→ GNSS DATA PROCESSING

Volume II: Laboratory Exercises

www.esa.int

European Space Agency

TM-23/2
May 2013

→ GNSS DATA PROCESSING

Volume II: Laboratory Exercises

J. Sanz Subirana, J.M. Juan Zornoza and M. Hernández-Pajares

European Space Agency

Acknowledgements

We would like to thank Dr Javier Ventura Traveset and Professor Dr Günter Hein for suggesting that we write this book and for their enthusiastic support and advice throughout the whole process. We also thank Carlos López de Echazarreta Martín for his help during the editing. We are grateful to Professor Dr Bernhard Hofmann-Wellenhof for his advice and encouragement. Special thanks go to Jordi Català Domínguez for redrawing most of the figures in this book.

Our thanks also go to Adrià Rovira García for his continuous help and the deep implications of his review of this material, to Dagoberto Salazar for initial reviews of the English and to Georgina Sanz for her help and support with the editing of this book. We are also grateful to ESA Communications for their supporting advice and cooperation.

Any comments regarding this book may be addressed to *jaume.sanz@upc.edu*.

Cover: Galileo is the European GNSS (ESA/J. Huart).

To Georgina Sanz Pons
This book is for you

An ESA Communications Production

Publication *GNSS Data Processing, Vol. II: Laboratory Exercises* (ESA TM-23/2, May 2013)

Production Editor K. Fletcher

Editing Contactivity bv, Leiden, the Netherlands

Publisher ESA Communications
ESTEC, PO Box 299, 2200 AG Noordwijk, the Netherlands
Tel: +31 71 565 3408 Fax: +31 71 565 5433 www.esa.int

ISBN 978-92-9221-886-7 (two volumes plus CD)

ISSN 1013-7076

Preface

This book is inspired by the welcome package that we offer to our doctoral students when they start their activities in the research group. The original package has been updated and compiled into two volumes which contain a self-learning course and software tools aimed at providing the necessary background to start work in an operative way in Global Navigation Satellite System (GNSS). The design and contents are focused on the instrumental use of the concepts and techniques involved in GNSS navigation and it is intended to include all the elements needed to understand how the system works and how to work with it. In this way, after working through the two volumes, the students should be able to develop their own tools for high-accuracy navigation, implementing the algorithms and expanding the skills learned.

The first volume is devoted to the theory, providing a summary of the GNSSs (GPS, Glonass, Galileo and Beidou) fundamentals and algorithms. The second volume is devoted to laboratory exercises, with a wide range of selected practical examples going further into the theoretical concepts and their practical implementation. The exercises have been developed with a specialised software package provided on a CD-ROM together with a set of selected data files for the laboratory sessions. This is an end-to-end GNSS course addressed to all those professionals and students who wish to undertake a deeper study of satellite navigation, targeting the GNSS data processing and analysis issues.

Starting from a review of the GNSS architecture, the contents of Volume I range from the analysis of basic observables (code pseudorange and carrier phase measurements) to setting up and solving the navigation equations for Standard Point Positioning (SPP) and Precise Point Positioning (PPP). It involves, in particular, an accurate modelling of GNSS measurements (up to the centimetre level of accuracy or better) as well as the required mathematical background to achieve the high-accuracy positioning goal. Parameter estimation techniques such as least squares and Kalman filtering are explained from a conceptual point of view, looking towards implementation at an algorithmic level.

For this self-contained educational package, we have tried not only to explain the theoretical concepts and provide the software tools, but also to illustrate the results and give the methodology on GNSS data processing. This is achieved in Volume II through guided examples developed in laboratory sessions using actual GNSS data files in standard formats. Moreover, some additional practical information on public domain servers with GNSS data products (precise orbits and clocks, ionospheric corrections, etc.) is included, among other items.

Most of the algorithms introduced in the theory are implemented in the GNSS-LABoratory (`gLAB`) tool suite. `gLAB` is an interactive and user-friendly educational multipurpose software package for processing and analysing GNSS data to centimetre-level positioning accuracy. The use and functionalities of this tool are thoroughly explained and illustrated through the different guided exercises in the laboratory sessions, together with self-explanatory templates, tool tips and warning messages included in the Graphical User Interface (GUI). `gLAB` has been developed under an European Space Agency (ESA) Education Office contract.

The `gLAB` tool suite is complemented with an additional software package of simple routines implementing different algorithms described in the theory.[1] These elementary routines are included as examples of basic implementations to help students build their own tools. As mentioned above, the target is to provide *effectiveness* in *instrumental use* of the concepts and techniques of GNSS data processing from scratch.

The didactic outline of this book is the result of more than 25 years of university teaching experience. In a similar way, the scientific/technological approach has been enhanced by our experience in developing different R&D projects in the GNSS field.

[1] The `gLAB` source code is also provided as an example of the full implementation of the algorithms in a self-contained tool with a wide range of capabilities.

Contents

How to use this book

This volume contains the practical part of the GNSS course developed in these books. The fundamentals and algorithms appear in Volume I.

The aim is to provide an experimental understanding of the theoretical basis and algorithms already introduced in the fundamentals and, along with this experimentation, to develop a methodology on GNSS data processing and analysis. This is achieved with a set of 15 practical sessions, distributed in six chapters, plus two appendices, and a specialised software package provided in a CD-ROM together with selected GNSS data files in standard format.

The first chapter introduces the GNSS-LABoratory (`gLAB`) tool suite, which is an interactive and user-friendly software tool allowing high-accuracy positioning. The use and functionalities of this tool are thoroughly explained and illustrated in different guided exercises. These exercises are motivated by selected examples illustrating GPS performances in different scenarios, including different geometries of satellites in view, high ionospheric activity conditions or satellite clock anomalies, among others.

The second chapter is devoted to describing the laboratory exercises environment and to providing an introduction to the GNSS standard data file formats. The UNIX (Linux) Operating System (OS) has been chosen as the baseline platform for the laboratory sessions, due to its high-performance and professional environment for dealing with these topics. For newcomers to this OS, a brief session on UNIX is included to introduce the basic commands and two additional software resources, `awk/gawk` and `graph.py`, widely used in the exercises.[2] Finally, the different GNSS standard file formats are explained in a user-friendly way through a set of HTML files with self-explanatory tool tips.

The third chapter focuses on GNSS satellite coordinates, clocks and reference frames. Satellite coordinates and clock synchronisation errors are analysed using both broadcast navigation message RINEX files and precise orbit and clock SP3 files. GPS satellite coordinates are computed using the broadcast pseudo-Keplerian orbital elements. Glonass orbits are integrated from the broadcast initial conditions, using a fourth-order Runge–Kutta method. Several related issues are studied, such as short- and long-term accuracy, integration step width and different perturbations on the satellite orbits. The variation of osculating elements due to Earth's oblateness or Sun–Moon acceleration is analysed from an experimental point of view. The accuracy of

[2] Note: `gLAB`, `awk` and `graph.py` are also available for the Windows OS.

the broadcast orbits and clocks is assessed against precise products, taking into account the different aspects involved, such as the reference frames and antenna phase centres used, coordinate interpolation, etc. Singular events, such as the GPS selective availability switch-off on 2 May 2000, or the Glonass reference frame transition from PZ-90 to PZ-90.02 on 20 September 2007, are also studied with actual data from files collected during such periods.

The fourth chapter targets GNSS measurements. Using actual measurements from RINEX files gathered from public domain data servers, the pseudorange (code) and carrier phase observables, as well as their different combinations (geometry-free, ionosphere-free, wide-lane, GRAPHIC), are analysed, and some of their characteristics directly from graphical results shown. Files collected under different ionospheric conditions (low solar activity, solar maximum, etc.), including special events such as the Halloween storm in October 2003, are processed to illustrate system performances in different scenarios, including extreme phenomena. The propagation of travelling ionospheric disturbances is also depicted from actual GPS measurements, in a very simple and straightforward way. Other exercises are devoted to the analysis of multipath and receiver noise in measurements and in combinations of measurements of two- and three-frequency signals. This study is carried out with several GNSS (GPS, Glonass and Galileo) data sources, including GPS measurements collected with anti-spoofing conditions both enabled and disabled. Different combinations of three-frequency signal measurements are analysed, assessing the noise and discussing their suitability for navigating or estimating ionospheric delays, as well. The feasibility of depicting the second-order ionospheric effect is also explored and discussed using combinations of carriers from three-frequency signals.

The fifth chapter deals with measurement modelling and positioning. The different terms involved in the code and carrier measurement modelling for SPP and PPP are studied in detail. Geometric ranges, relativistic correction, atmospheric effects (ionosphere and troposphere), instrumental delays, wind-up, solid tides, etc., are analysed and their magnitude and impact assessed on range and user domains. Using precise IGS products as reference values and GNSS measurements, the error budget in the SPP model is assessed and the transfer of range domain errors to the position domain is analysed.

As mentioned above, the use and functionalities of `gLAB` are explained using the exercises in the laboratory sessions. As an example, a deep study of the computation of the static PPP solution for an IGS permanent receiver site is undertaken. This covers checks on all the products used (SP3 orbits and clocks, and ANTEX files), models and parameter settings (in the Kalman filter) applied to achieve a centimetre level of agreement between the IGS solution for the receiver coordinates and clock (SINEX file), as well as zenith tropospheric delay estimates. Other examples illustrating `gLAB` capabilities to process trajectories are illustrated with the kinematic positioning of a low Earth orbit satellite and an aircraft flight.

To explain the implementation of the algorithms in detail, the GPS measurements are modelled by hand from scratch for the SPS, using simple routines for elemental functions (Klobuchar, troposphere, etc.), when needed. Afterwards, the navigation equations system is proposed and solved with Octave or MATLAB, using least squares and Kalman filtering.

The laboratory sessions end with a work of synthesis presented in the sixth chapter. It is inspired by the paper by [Springer and Dilssner, 2009] published in the *Inside GNSS* journal and motivated by the analysis of the GPS SVN49 anomaly on the L1 signal.

Two additional laboratory sessions are given in Appendix A, as complementary background, to help the reader develop his or her own GNSS software tools. A set of examples of elemental routines implementing basic functions (Klobuchar model, tropospheric model, satellite coordinates computation, among others) for end-to-end single-frequency GPS positioning are analysed in detail in the first session. Examples of dual-frequency-based cycle-slip detectors, among single-frequency detectors, are also given. The second session illustrates the use of `gLAB` as a library for developing GNSS software.

Although a basic knowledge of the Linux OS is desirable, it is not essential to follow the book. By means of the different 'guided' exercises, the reader is introduced in a natural way and by immersion to the syntax and possibilities of this environment. Our experience has shown that students with no previous knowledge of Linux, or the tools used in the exercises, do not experience any great difficulty in familiarising themselves with this environment; furthermore, they appreciate the fact that their training is applied to a real context where these problems are usually encountered. Nevertheless, and taking into account that the fundamental purpose of this publication is GNSS training, graphical results of the exercises have been included after each laboratory session, to develop many of the concepts of these sessions without the need to execute the programs.

1. The gLAB Tool Suite

The GNSS-LAB tool suite (gLAB) is an advanced interactive, educational, multipurpose package for processing and analysing GNSS data.

From an operational point of view, this tool was conceived as a software resource to support a practical GNSS course, where the fundamentals introduced in the theory are experimented upon by means of guided exercises. In this way, gLAB is used as the baseline for developing the exercises in this book. As well as gLAB, other additional programs are also used for some specific computations.

gLAB is a quite flexible software tool capable of high-accuracy positioning, and able to run under the Linux and Windows operating systems. It was developed by Research group of Astronomy and Geomatics (gAGE) at the Technical University of Catalonia (UPC) under ESA Education Office contract N. P1081434 and is free of charge from ESA to university enterprises and GNSS professionals.

Figure 1.1: gLAB Graphical User Interface: Main panel.

Figure 1.2: gLAB panels. First row *left*: Input panel; first row *right*: Preprocess panel. Second row *left* Modelling panel; second row *right*: Filter panel. Third row *left*: Output panel; third row *right*: Analysis section panel.

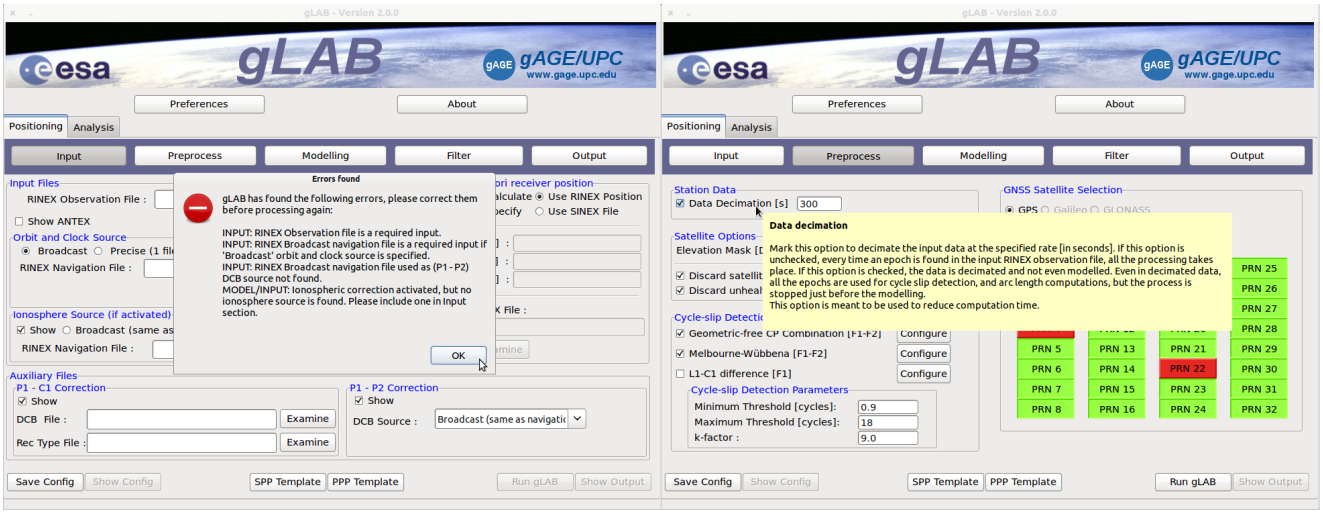


Figure 1.3: Some examples of tool tips and warning messages of gLAB. The pre-configured templates for Standard Point Positioning [SPP] and Precise Point Positioning [PPP] modes are indicated at the bottom of the panels.

Main gLAB Features

The gLAB tool features a wide range of characteristics only available in advanced GNSS data processing software. The more relevant ones are mentioned here, but a more detailed list may be found in the Software User Manual [Ramos-Bosch et al., 2010] and in [Sanz et al., 2012b]:

- **High-accuracy positioning capability:** This software tool implements a precise modelling of the GNSS measurements (code and carrier phase) at the centimetre level, allowing both standalone GPS positioning and PPP.

  The first release of this software package implements full processing capabilities for GPS data and it is designed to incorporate future module updates, such as an expansion to Galileo, Glonass and Beidou systems.

  It is capable of reading a variety of standard formats, such as RINEX-3.00, SP3, ANTEX or SINEX files, among others. Moreover, functionality is also included for GPS, Galileo and Glonass, allowing some data analysis to be performed with real multi-constellation data.

- **Fully configurable:** gLAB is driven by a configuration file, where the different internal parameters (input/output and data processing options) are set. This configuration file is generated by the Graphical User Interface (GUI)[1] as an American Standard Code for Information Interchange (ASCII) file that can be modified with a text editor by experienced users, as well.

  To help configure gLAB, the GUI is organised into different sections or panels with clickable buttons to set up the different options. Figure 1.2 shows examples of such panels.

[1] Actually, the main purpose of the GUI is to generate the configuration file, which is read by the gLAB core (an executable file coded in C).

The reader is directed to the gLAB Software User Manual for a wider explanation. On the other hand, gLAB's usage and capabilities are introduced and explained throughout the different laboratory exercises of this book.

- **Easy to use:** gLAB includes an intuitive GUI, with tool tips and explanations of the different options to select. Guidelines and several error and warning messages are also provided, as well as templates for pre-configured processing modes, see Fig. 1.3.

  The GUI is divided into two main sections: `Positioning` and `Analysis`. The `Positioning` section provides all the options for configuring the input data, observation model and navigation filter parameters, among other settings. The data `Analysis` tool provides a user-friendly environment for the data analysis and visualisation of the results.

  gLAB can also be executed in command-line form, which allows it to be included in scripts (batch processing) for automatic or massive data processing.

- **Access to internal computations:** Wide tracking of internal computations is provided by gLAB through different output messages. This written information is optional and these messages can be selected in the output panel, see Fig. 1.4.

> See Appendix B for the contents of the CD and for the software installation.

Note that the exercises in this book are based on gLAB version 2.0.0.

Figure 1.4: Output panel showing the different printable messages with information on the internal computations.

Session 1.1. Examples of Standard and Precise Positioning with gLAB

Objectives

To present some examples of Global Positioning System (GPS) data processing with gLAB and to illustrate the Standard Positioning Service (SPS) and the PPP performances.

Files to use

alrt3060.09o, kour3060.09o, madr3060.09o, brdc3060.09n, nacc080a.09o, brdc0800.09n, medi1230.00o, brdc1230.00n, upc31530.06o, brdc1530.06n, igs10602.sp3, igs15561.sp3, Postfix.kml, Prefix.kml, upc31530.06o, brdc1530.06n, amc230[1-5]0.03o, brdc30[1-6]0.03n, amc23050.03o, brdc3050.03n, igs_pre1400.atx, igs05_1552.atx, gage1580.05o, upc41580.05o, brdc1580.05n

Programs to use

gLAB_linux

Development

Session files:

Copy and uncompress these session files in the working directory[2]:

```
cp ~/GNSS/PROG/SES11/* .
cp ~/GNSS/FILES/SES11/* .
gzip -d *.gz *.Z
```

1. Standard Point Positioning (SPP)

This exercise will present a simple example of GPS single-frequency code-based positioning with broadcast orbits and clocks, that is a navigation message, using gLAB (SPP is explained in section 6.1, Volume I).

Using the Receiver INdependent EXchange format (RINEX) measurement and broadcast orbit and clock files, madr3060.09o and brdc3060.09n, compute the SPP solution with gLAB. Complete the following steps[3]:

(a) To run the gLAB GUI execute[4]

```
gLAB_GUI.py &
```

[2] According to the *software installation instructions* (see Appendix B), in this book we will assume that the CDROM content has been copied into the GNSS directory, located within the home directory.

[3] See the Notepad 'ntpd_ses11.scr'. This file contains equivalent command line sentences to execute these laboratory exercises.

[4] Programs gLAB_linux and graph.py, and files MainLogo.gif, Header.gif, gAGE.png and gAGE.ico must be available in the user installation directory (as is done when following the installation instructions in Appendix B.2), or in the directory /usr/share/gLAB, or in the directory where gLAB_GUI.py is executed.

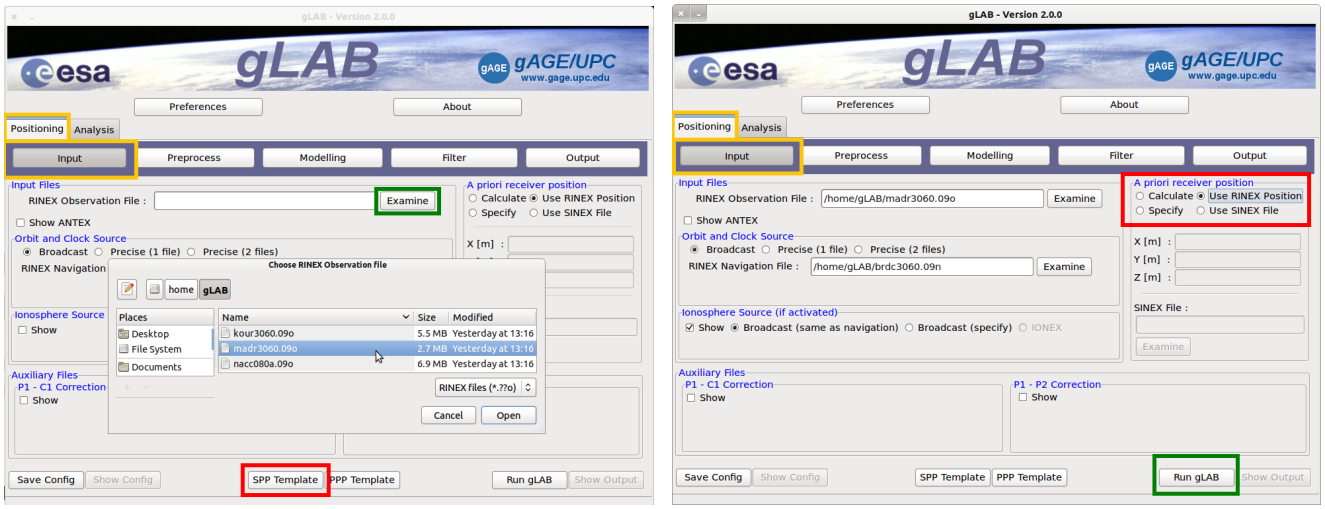


Figure 1.5: The [Input] section configuration.

The main panel of Fig. 1.1 is open. Click the `[Input]` button to open the *Input Section*. The panel shown in Fig. 1.5 will appear.

(b) Select the **default options for SPP processing**. This is done by clicking the button `SPP Template` in the `[Input]` section.

(c) In the `[Input]` section, click the `Examine` button for the `RINEX observation file` and select the file `madr3060.09o` in the `WORK` directory.[5] Do the same for the RINEX navigation file[6] `brdc3060.09n`

(d) Execute `Run gLAB` and wait for the processing to end.

(e) In the `[Analysis]` tab, click the `NEU positioning error` button and, then, click `Plot` to generate the plot (see Fig. 1.6, notice that vertical axes in this figure range from −10 to 10 m).

This plot shows the discrepancies, in North East Up (NEU) coordinates, between the computed solution and the '`A-priori Receiver Position`' defined in the RINEX file.[7]

Note that the measurements of file `madr3060.09o` were collected with a static receiver (i.e. with fixed coordinates). Thus the noise seen in the plot is due to the positioning error.

(f) Now click the `Horizontal positioning error` button and then click `Plot` to generate the figure. This plot shows the positioning error in the horizontal plane (see the plot on page 23, top right).

[5] By default, the browser selects the files '`*.??o`', with lower or upper case letters. If the required file does not appear in the browser, then select the option '`All files`' to see whatever is available in the directory.

[6] In SPP mode, the button `[⊙ Broadcast]` of the `Orbit and Clock Source` is selected by default.

[7] This is the default option. Four options are available in the `A-priori receiver position` box of the `[Input]` section: `[⊙ Use RINEX position]`, `[◯ Calculate]`, `[◯ Specify]` and `[◯ Use SINEX File]`, see Fig. 1.5, right.

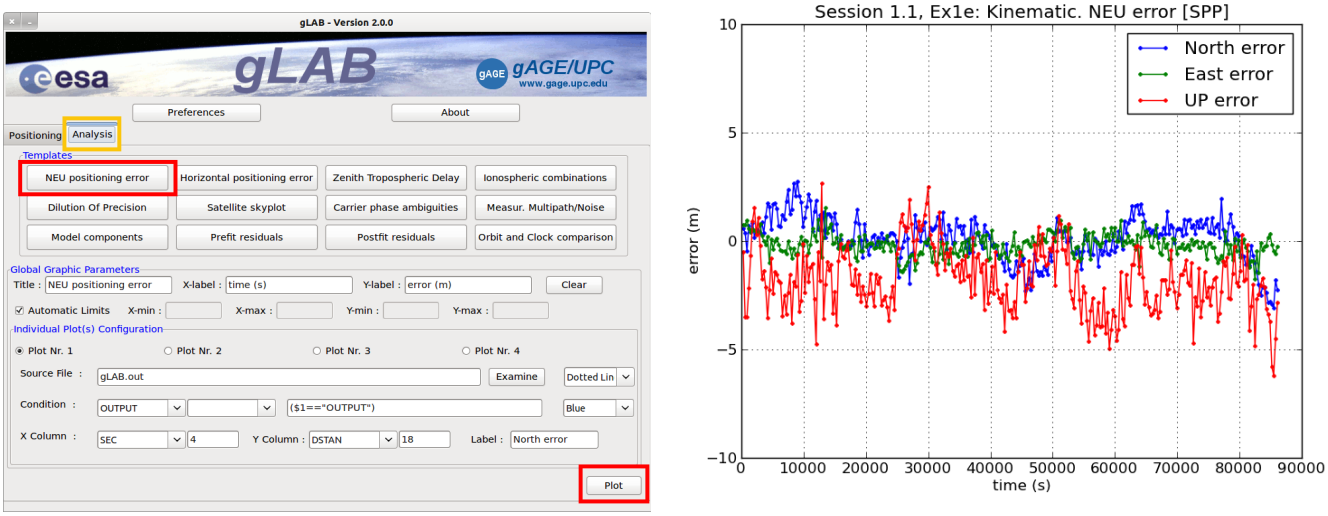


Figure 1.6: *Left*: the [Analysis] panel of the gLAB GUI. *Right*: the 'NEU positioning error plot'. Note that the vertical ranges have been adjusted for a better view of the plot.

2. SPP processing in static mode

As mentioned above, the measurements of file `madr3060.09o` were collected by a static receiver (i.e. with fixed coordinates). Thus the positioning accuracy can be improved by configuring the navigation filter to process the receiver coordinates as constants.[8] This can be done by selecting the option `[⊙ Static]` in the column on the right of the `[Filter]` section options in the `[Positioning]` tab (see Fig. 1.7).

Note that option `[⊙ Kinematic]` is activated by default for SPP positioning, as it corresponds to the GPS *SPS*.

After setting the static mode, click `Run gLAB` to process the data.

In the `[Analysis]` tab, repeat the same plots as in the previous case:

(a) Click the `NEU positioning error` button and then click `Plot`. See the plot in Figure 1.7 or on page 23, second row left.

(b) Now click the `Horizontal positioning error` button and then click `Plot`. See the plot on page 23, second row right.

Comment: The `[Filter]` section panel shows the configuration of the navigation filter used for the SPP mode. It allows the user to select the type of measurements to use (raw pseudorange, pseudorange smoothed, pseudorange and carrier phase) (see Chapter 4 in Volume I), the frequencies (single or dual frequency), the positioning mode (static or kinematic) and, in general, the configuration of the navigation filter (see Chapter 6 in Volume I). The configuration of the navigation filter is discussed in detail in session 5.1, exercise 2d, see for instance page 173.

[8] Roughly speaking, the filter will average the estimation of such coordinates over time.

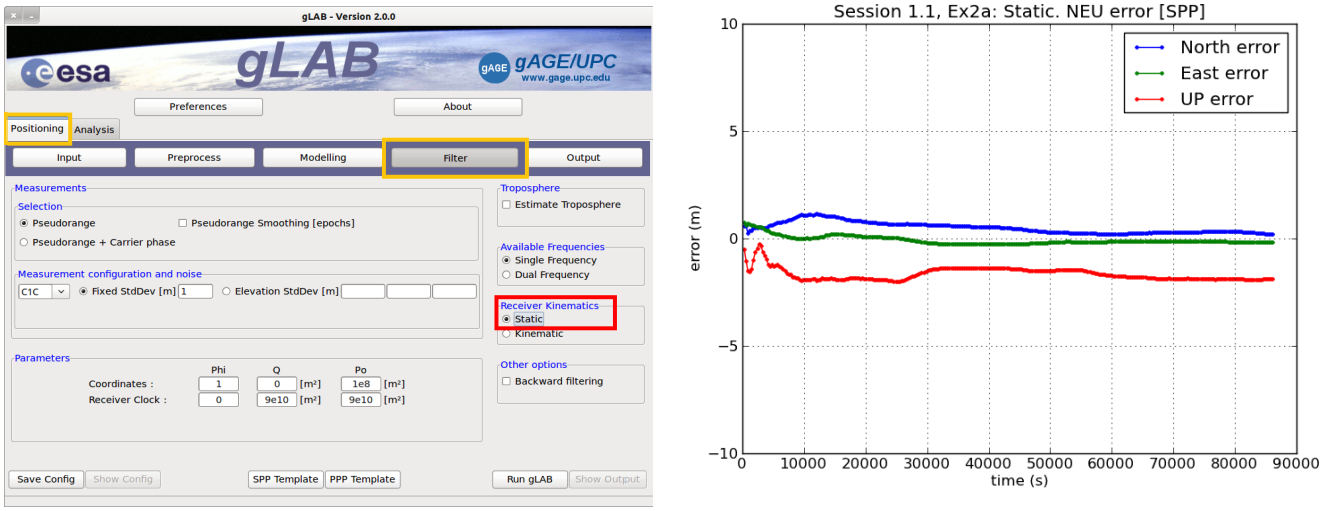


Figure 1.7: *Left*: the [Filter] section configuration of the gLAB GUI for SPP processing in Static mode. *Right*: the 'NEU positioning error' plot (in static mode).

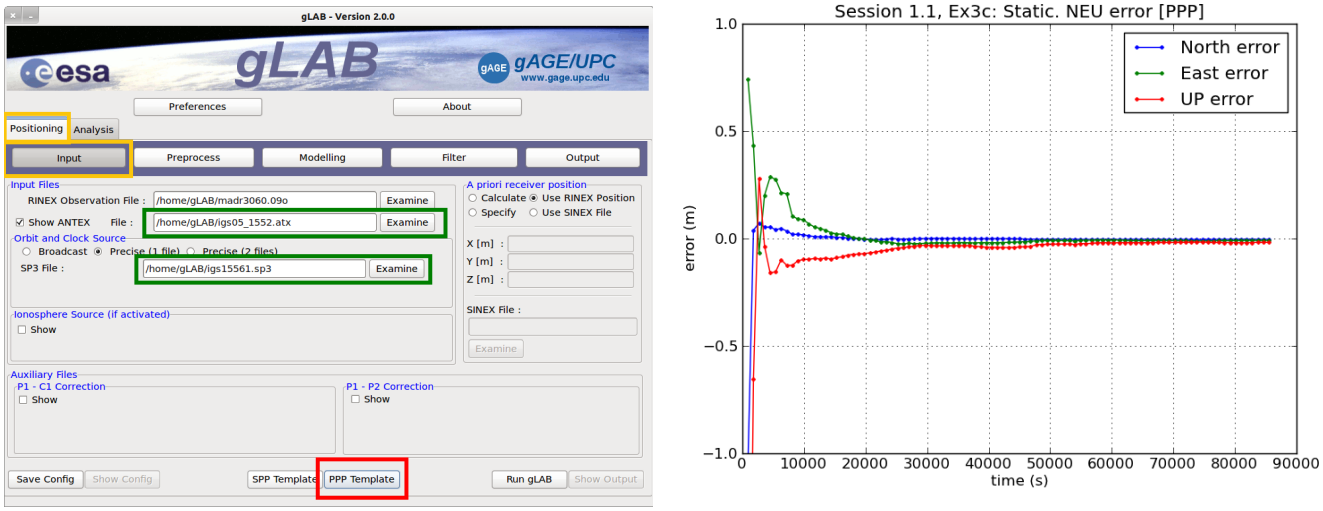


Figure 1.8: *Left*: the [Filter] section configuration of the gLAB GUI for PPP processing in Static mode. *Right*: the 'NEU position error' plot (in static mode).

3. Static Precise Point Positioning (PPP)

This exercise shows how to perform PPP (see section 6.2 in Volume I) with `gLAB`, in a very quick and easy way.

The approach relies on dual-frequency code and carrier GPS data. Moreover, postprocessed precise orbit and clock products from the International GNSS Service (IGS) are used, instead of broadcast values (see sections 3.2.3 and 3.3.3 in Volume I), to ensure a very accurate solution.

The measurements were collected with a receiver with fixed coordinates, and hence they are processed in `Static` mode to achieve the highest accuracy. This way provides an accuracy of a few centimetres over 24 h data.

(a) Configuration options for PPP:

i. Select the **default options for PPP processing**. This is done by clicking the button `PPP Template` in the `[Input]` section (see Fig. 1.8, left).

ii. In the `[Input]` section, click the `Examine` button for the `RINEX observation file` and select the file `madr3060.09o` in the working directory.[9]

iii. Select the ANTEX file `igs05_1552.atx` by clicking the corresponding `Examine` button.[10]

iv. The option `[⊙ Precise (1 file)]` is set by default. Then, click the button `Examine` and select the file `igs15561.sp3` with the precise orbit and clock data (see Fig. 1.8, left).

(b) Once the data files and the configuration options for PPP mode are selected, click `Run gLAB` to process the data.

(c) Then, go to the `[Analysis]` tab and produce the same plots as in the previous case. That is:

i. Click the `NEU positioning error` button and then `Plot` (see the plot in Fig. 1.8, on the right).

ii. Click the `Horizontal positioning error` button and then `Plot`. See the plot on page 23, third row right.

4. Kinematic PPP processing

In the previous exercise, the permanent receiver MADR has been processed assuming its coordinates are constants (i.e. static mode) in order to assess the higher positioning accuracy achievable for a permanent receiver over a 24 h data campaign. Here, the same data files (measurement and IGS precise orbits and clocks) will be processed in PPP kinematic mode to assess the expected navigation accuracy for a roving receiver.

This can be done in `gLAB` by selecting the option `[⊙ Kinematic]` in the right-hand options of the `[Filter]` section in the `[Positioning]` tab (see Fig. 1.9, left).

Notice in figure 1.9 that the option `[⊙ Dual Frequency]` is activated by default for the PPP positioning, and that code (pseudorange) and carrier phase measurements are used in the filter (see section 6.2 in Volume I).

To process the data, click `Run gLAB`, and then go to the `[Analysis]` tab to draw the plots.

Under this `[Analysis]` tab, repeat the same plots as in the previous exercise

(a) Click the `NEU positioning error` button and then click `Plot` (see the plot in Fig. 1.9, on the right).

(b) Click the `Horizontal positioning error` button and then click `Plot`. See plot on page 23, last row right.

[9] The file `madr3060.09o` will remain selected if the user has not exited the program.

[10] This file contains the receiver and satellite Antenna Phase Centre (APC) offsets, which are needed for this high-accuracy processing (see section 5.6 in Volume I).

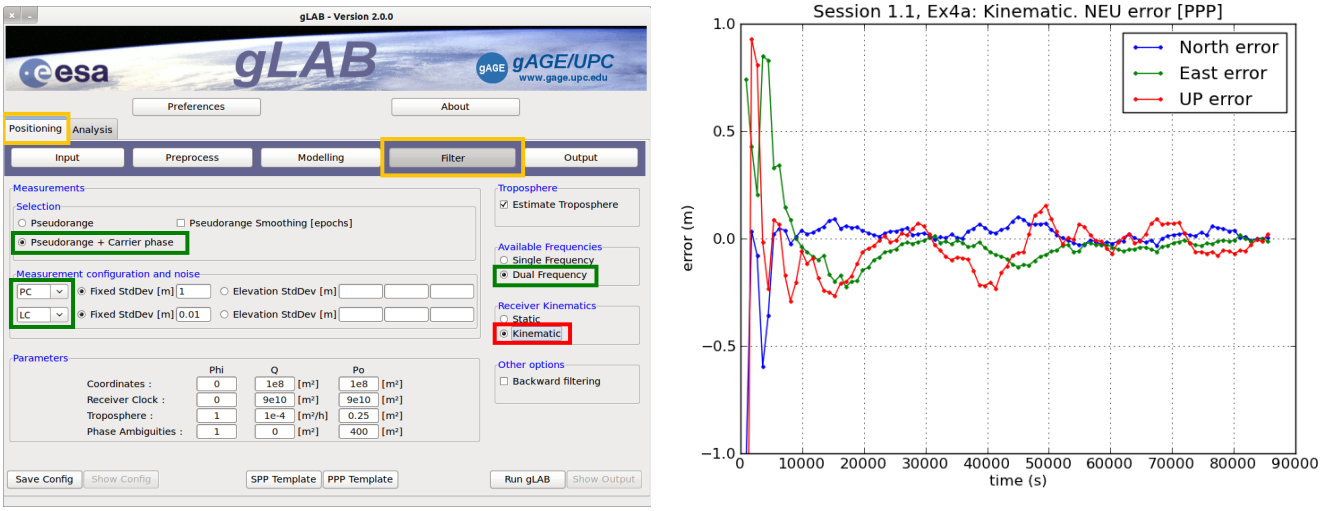


Figure 1.9: *Left*: the gLAB [Filter] for PPP processing in Kinematic mode. *Right*: the 'NEU error' plot (in kinematic mode).

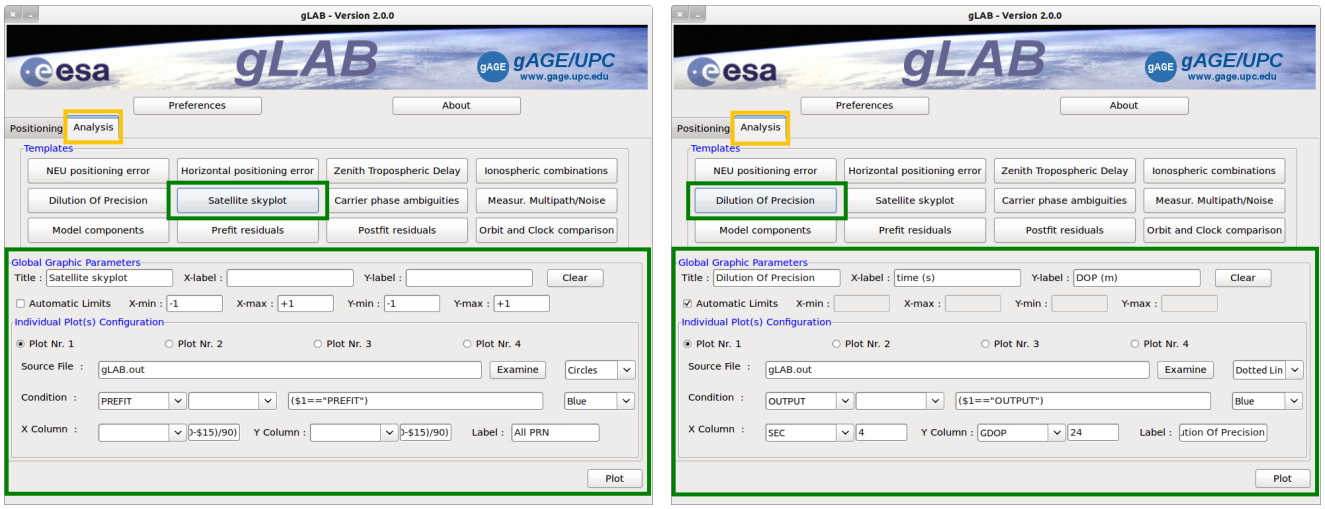


Figure 1.10: The gLAB [Analysis] tab configuration: satellite skyplot (*left*) and DOP plot (*right*).

5. SPP and receiver location

The geometry of the satellites in view and the positioning error are illustrated in this exercise for three different locations: close to the North Pole (ALRT ($\varphi = 82.5°$, $\lambda = 297.7°$), Canada), mid-latitude (MADR ($\varphi = 40.4°$, $\lambda = 355.7°$), Spain) and the equator (KOUR ($\varphi = 5.2°$, $\lambda = 307.2°$), French
Guyana).

The data files `alrt3060.09o`, `madr3060.09o` and `kour3060.09o` were collected on the same day (2 September 2009) by three permanent IGS receivers located at these locations. The associated broadcast orbits and clocks are given in the file `brdc3060.09n`.

For each station it is proposed to apply the following procedure:

(a) Process the data file in the SPP mode, as in the exercise (1).

(b) Produce the following plots:

i. A skyplot showing the tracks of the satellites in view on the local sky. This plot corresponds to the pre-configured option '`Satellite Skyplot`' in the `[Analysis]` tab (Fig. 1.10, left).

ii. The horizontal positioning error. This plot corresponds to the '`Horizontal positioning error`' plot in the `[Analysis]` tab.

iii. The NEU errors as a function of time. This plot corresponds to the '`NEU positioning error`' plot in the `[Analysis]` tab.

iv. The number of satellites used in the solution can be added to the plot with the button `[⊙ Plot 4]`, selecting the `Source file gLAB.out`, the `Condition EPOCHSAT`, the `X column` 4 and `Y column` 6.

v. The Dilution Of Precision (DOP) components GDOP, HDOP, VDOP and TDOP (see section 6.1.3.2 in Volume I) as a function of time. These plots corresponds to the pre-configured option `Dilution Of Precision` (see Figs 1.10 to 1.13).

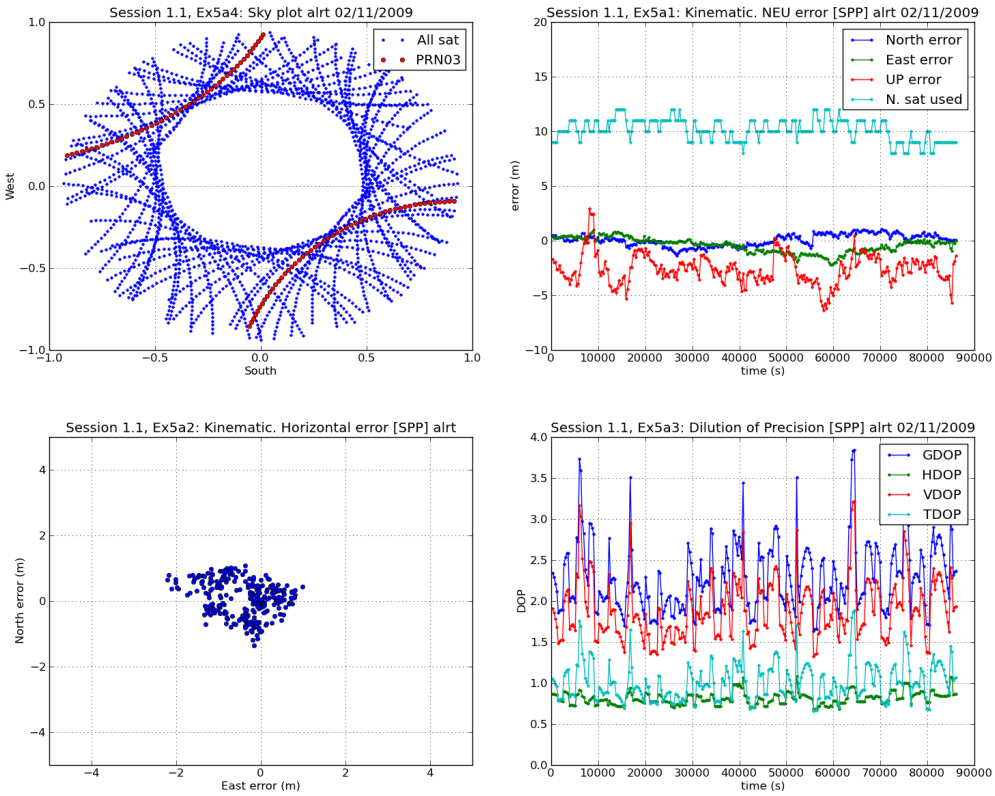
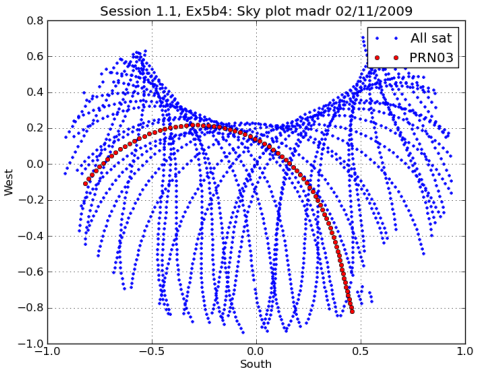
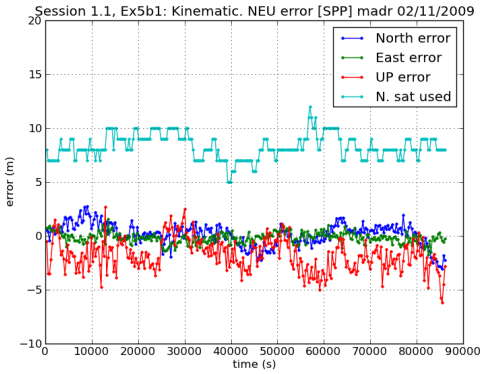
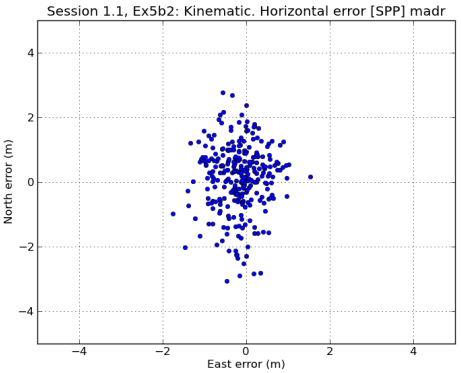


Figure 1.11: GPS SPS, 2 September 2009. IGS station ALRT (Ellesmere Island), Canada ($\varphi = 82.5°, \lambda = 297.7°$). Skyplot (first row left). ENU error and number of satellites used as a function of time (first row right). Horizontal positioning error (second row left). GDOP, HDOP, VDOP and TDOP (second row right). See the plots for the MADR and KOUR stations in Figs 1.12 and 1.13.







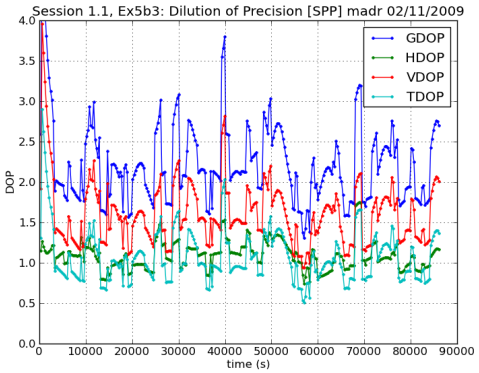
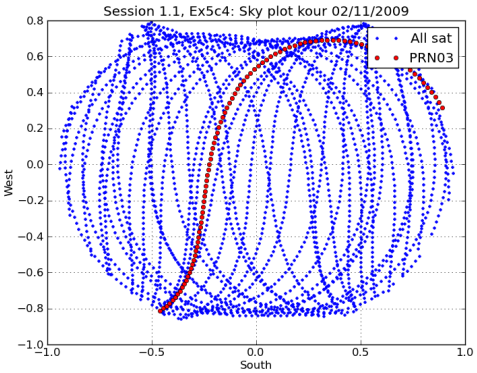
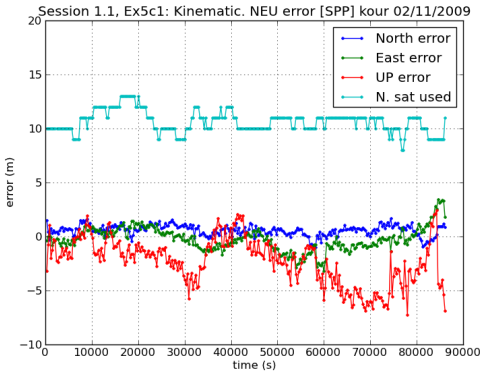
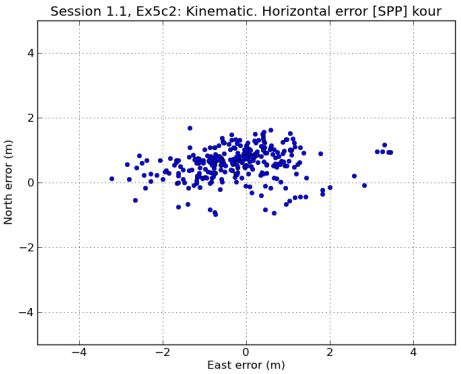


Figure 1.12: The same as 1.11 for MADR (Madrid), Spain ($\varphi = 40.4°, \lambda = 355.7°$).







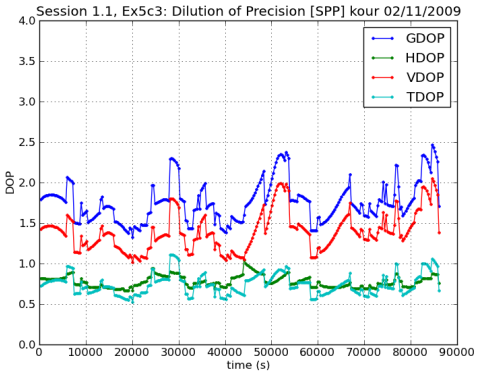


Figure 1.13: The same as 1.11 for KOUR (Kourou), French Guyana ($\varphi = 5.2°, \lambda = 307.2°$).

6. SPP processing and receiver quality

The single-frequency SPP positioning error of a low-cost receiver and a high-performance receiver will be compared in this exercise.

The data file `gage1580.05o` was collected with the inexpensive single-frequency Trimble Lassen SK-II receiver, with an active microstrip antenna. A second data file `upc41580.05o` was simultaneously collected in the same environment[11] with a high-performance dual-frequency receiver, namely the Septentrio Polar Rx2 with a GPS-600 Pinwheel Antenna.

Applying the same procedure as in the previous exercise 1, but setting `Data Decimation` to 30 s in the `[Preprocess]` section, process both data files and compare the results. The broadcast orbits and clocks are provided in the `brdc1580.05n` file.

The plots in Figs 1.14 and 1.15 show the user domain accuracy found after processing such data files with `gLAB`.

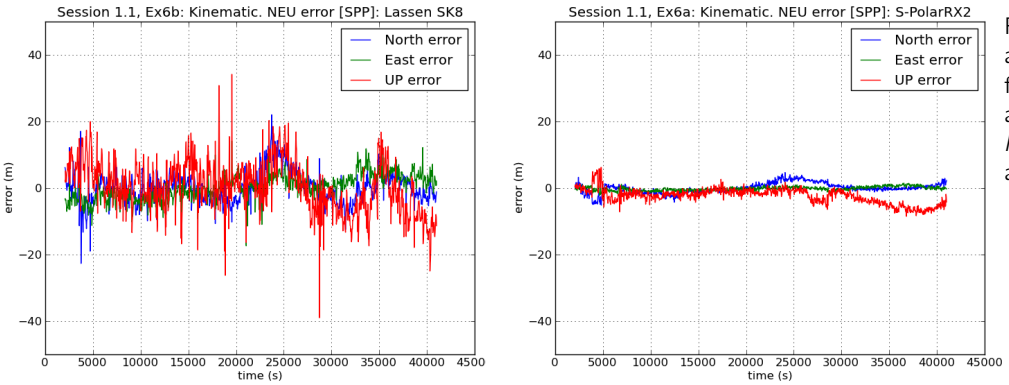


Figure 1.14: The NEU accuracy as a function of time for the *low-cost* receiver and antenna (*left*) and the *high-performance* receiver and antenna (*right*).

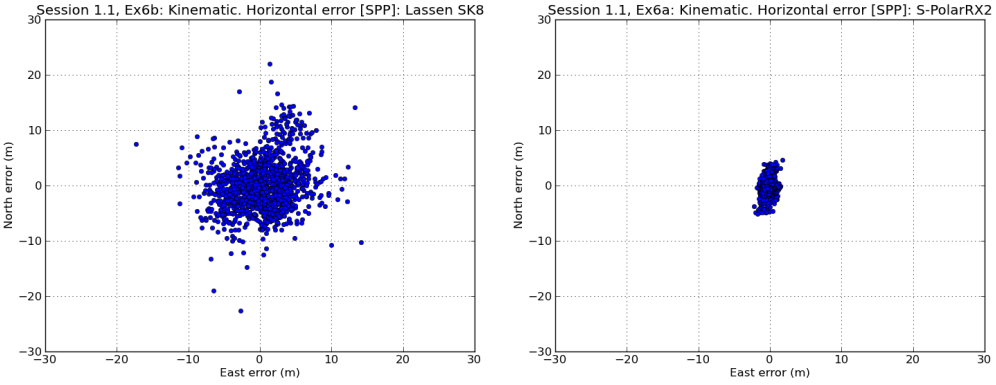


Figure 1.15: The horizontal accuracy for the *low-cost* receiver and antenna (*left*) and the *high-performance* receiver and antenna (*right*).

[11] They were located on the roof of the gAGE/UPC building in Barcelona on 7 June 2006, under an open sky.

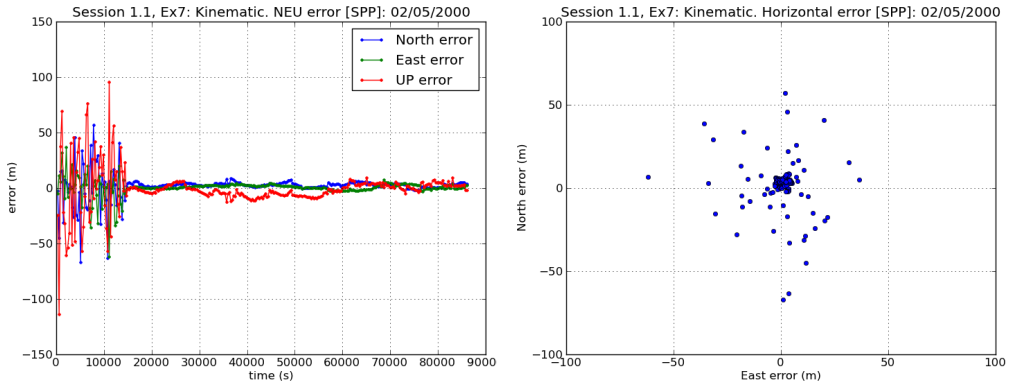


Figure 1.16: SPS accuracy before and after S/A switch-off.

7. 2 May 2000: Selective Availability (S/A) switched off

S/A was an intentional degradation of public GPS signals (mostly in the satellite clocks, see exercise on page 81) implemented for US national security reasons. US President Bill Clinton ordered S/A turned off at midnight on 1 May 2001.

The data file `medi1230.00o` was collected on 2 May 2000 by a permanent receiver in Italy at coordinates $\varphi = +44°31'$, $\lambda = 11°39'$. Process this data file applying the same procedure as in the previous exercise 1, using the navigation file `brdc1230.00n`. Assess the improvement in user domain accuracy after removing the S/A (see the results in Fig. 1.16).

8. Error model component assessment

As explained in Chapter 5 of Volume I, the GNSS signals are affected by several error sources that must be taken into account to allow GNSS positioning and/or to improve user domain accuracy. `gLAB` implements measurement modelling up to the centimetre level to achieve high-accuracy positioning as has been shown in the previous exercise 7.

The impact on user domain accuracy of neglecting the different model terms in the GNSS positioning will be analysed in this exercise. This will be done using the data set of the previous exercise 7 in order to assess the worsening performance in the S/A on and S/A off scenarios. A deeper analysis of the error model components can be found in the laboratory exercises on page 193.

The list of different model components implemented in `gLAB` is given in the `[Modelling]` section of the `[Positioning]` tab. These model components can be switched `on/off` by clicking the corresponding button.[12]

Thus, the contribution of the different model terms on user domain accuracy can be assessed from the `[Modelling]` section as follows.

[12]The model terms used in the pre-configured mode `SPP` or `PPP` are not the same. On the one hand, the `PPP` uses the ionosphere-free combination of measurements (see section 5.4.1.1 in Volume I), hence the ionospheric model or the P1−P2 Differential Code Biass (DCBs) are not required (see sections 5.4.1.2 and 5.3 in Volume I). On the other hand, the `SPP` is based on the code measurements, with error noise at the level of 0.5 m (when smoothed with the carrier) or more, so it does not make sense to apply model corrections of the order of a few centimetres.

(a) *Relativistic clock correction effect* (see the modelling of this effect in section 5.2.1, Volume I):

i. Set the `[SPP Template]` and uncheck the `[☐ Relativistic clock correction]` button in the section `[Modelling]` (see Fig. 1.17, left).

ii. Click the `Run gLAB` button to reprocess the data without this model component being taken into account.

iii. Produce the `NEU positioning error` plot and compare results with previous plots. It is recommended that the vertical range in the plot is set to ±150 m to better see the results (i.e. `Y-min=-150` and `Y-max=150` in the plot limits). See the plot results in Fig. 1.18.

(b) *Ionospheric correction effect* (see section 5.4.1.2.1, Volume I). Again check the `[☒ Relativistic clock correction]` button and then uncheck the `[☐ Ionospheric correction]`. Repeat the same procedure as in the previous case. See plot on page 26.

(c) *Tropospheric correction effect* (section 5.4.2.1, Volume I). The same as before. See plot on page 26.

(d) *P1−P2 correction effect* (section 5.3, Volume I). The same as before.

(e) *Earth rotation correction effect* (section 5.1.1, Volume I). The same as before. See plot on page 26.

(f) *Satellite movement effect* (section 5.1.1.2, Volume I). The same as before (see the plot in Fig. 1.18, right).

(g) *Satellite clock offset effect* (section 5.2, Volume I). The same as before. See plot on page 27.

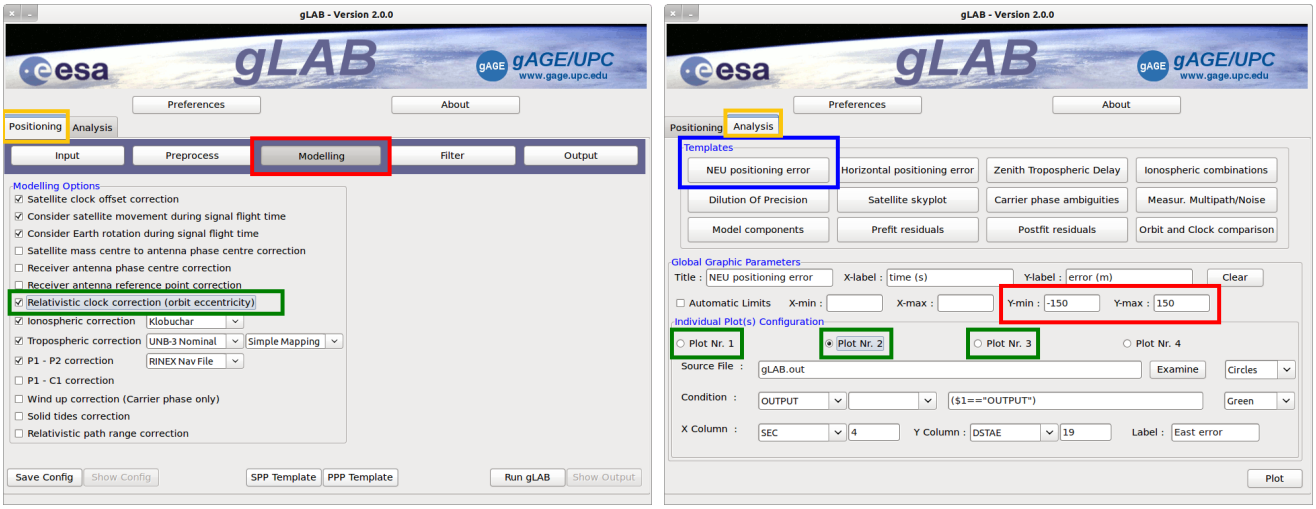


Figure 1.17: Configuration panels: model components (*left*); plotting (*right*).

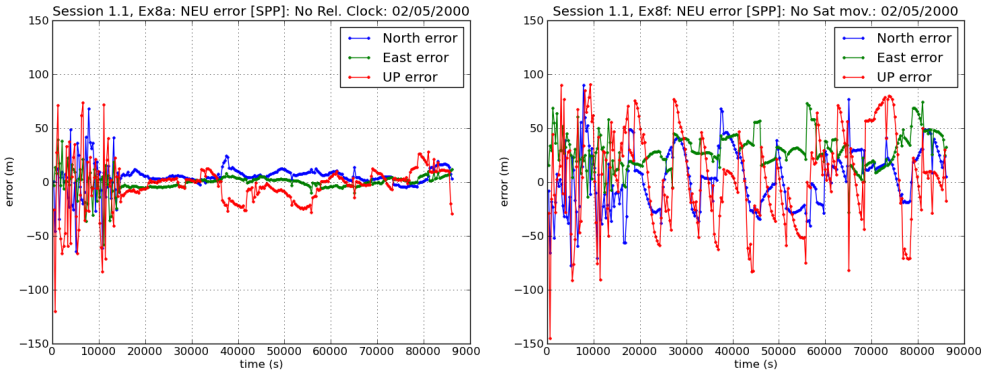


Figure 1.18: NEU position error as a function of time. Comparison of the effect of neglecting error model components with S/A on (time before 14 000 s) and S/A off (time after 14 000 s). Left-hand plot shows the accuracy without applying the relativistic correction. Right-hand plot shows the results without applying the correction due to satellite motion during the signal travel.

9. SPP performance during the Halloween ionospheric storm

A severe ionospheric storm occurred on 29–31 October 2003, producing an increase in the electron density which led to large ionospheric refraction values on the GPS signals (for more details see the paper [HJS et al., 2005]). These ionospheric conditions were beyond the capability of the GPS ionospheric model (see the Klobuchar model in section 5.4.1.2.1, Volume I), broadcast for single-frequency users, producing large errors in the SPS.

Dual-frequency users, navigating with the ionospheric-free combination of GPS signals, were not affected by these errors. As shown in section 5.4.1.1 in Volume I, the ionospheric refraction can be removed by up to 99.9% using dual-frequency signals.

The effect of this storm on single-frequency users will be assessed using a set of measurements collected by the IGS receiver AMC2 in Colorado Springs (USA) ($\varphi = 38.8°, \lambda = 255.5°$), between 28 October and 1 November. The positioning accuracy with two-frequency signals (with the code ionosphere-free combination) is also computed for comparison.

(a) Using the data files `amc23010.03o` and `brdc3010.03n` (28 October 2003, the day before the storm) follow the next steps:

i. Compute the SPS navigation solution (i.e. process the data using the default `SPP Template`).

ii. Produce the `NEU positioning error` plot.

iii. Plot the ionospheric refraction for code measurements (see Volume I, section 4.1). The `[Analysis]` tab includes the button `Ionospheric combinations` with the default configuration to produce this plot (see Fig. 1.19, left). See plot on page 27.

iv. Compute the navigation solution for a dual-frequency user, using the ionosphere-free combination of codes. This can be done by selecting the option `[⊙ Dual frequency]` in the `[filter]` section, and unchecking `[□ Ionospheric correction]` and `[□ P1-P2 correction]` in `[Modelling]` of the `[Positioning]` tab.

v. After completing these settings, click the `Run gLAB` button.

vi. Produce the `NEU positioning error` plot.

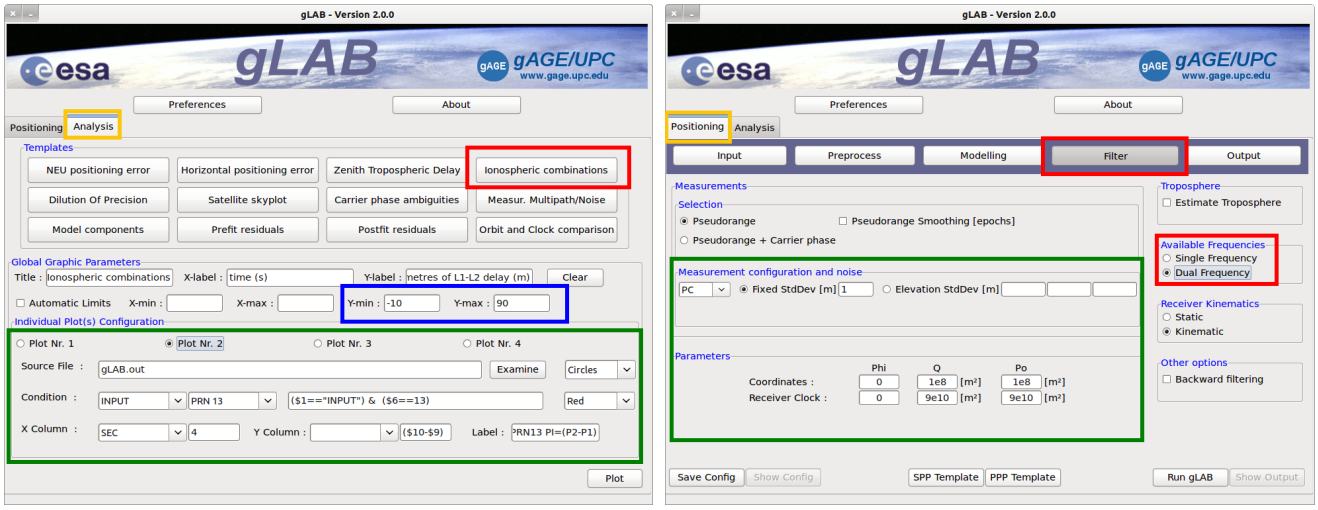


Figure 1.19: gLAB panels: default configuration to plot the code ionospheric delays (*left*); filter configuration for dual-frequency positioning (*right*).

(b) Repeat the previous process for 29 October 2003: use the data files `amc23020.03o` and `brdc3020.03n`. See plot on page 27.

(c) Repeat the previous process for 30 October 2003: use the data files `amc23030.03o` and `brdc3030.03n`. See plot in Fig. 1.20.

(d) Repeat the previous process for 31 October 2003: use the data files `amc23040.03o` and `brdc3040.03n`. See plot in Fig. 1.20.

(e) Repeat the previous process for 1 November 2003: sue the data files `amc23050.03o` and `brdc3050.03n`. See plot on page 28.

10. GPS clock anomaly

A GPS clock anomaly occurred on satellite PRN30 on 2 June 2006. An important clock drift of about 0.5 ns/s (i.e. a 15 cm/s drift on pseudorange measurements) was suddenly experienced around 20:00 UTC.

This event is analysed in the paper by [Marechal et al., 2007], which explains that: 'after 12 minutes of clock drift, the clock offset was equal to 550 ns creating errors in user pseudorange of at least 165 metres[...] After 20:14:06 and for 40 minutes, most of the receivers lost track of the satellite[...] At 20:55, clock drift was equal to 4.5 ns/s, equivalent to 1.35 m/s in pseudorange, leading to potential errors of several kilometres'. NANU[13] 2006052 was issued at 20:17, setting the satellite to unusable.

Analyse the effect of this clock anomaly on the position domain accuracy using the measurements data file `upc31530.06o` collected by the permanent receiver UPC3 in Barcelona ($\varphi = 41.4°, \lambda = 2.1°$), Spain, and the broadcast orbit and clock file `brdc1530.06n`.

[13] Notice Advisory to NAVSTAR Users (NANU), *http://cgls.uscg.mil/pipermail/nanu/*.

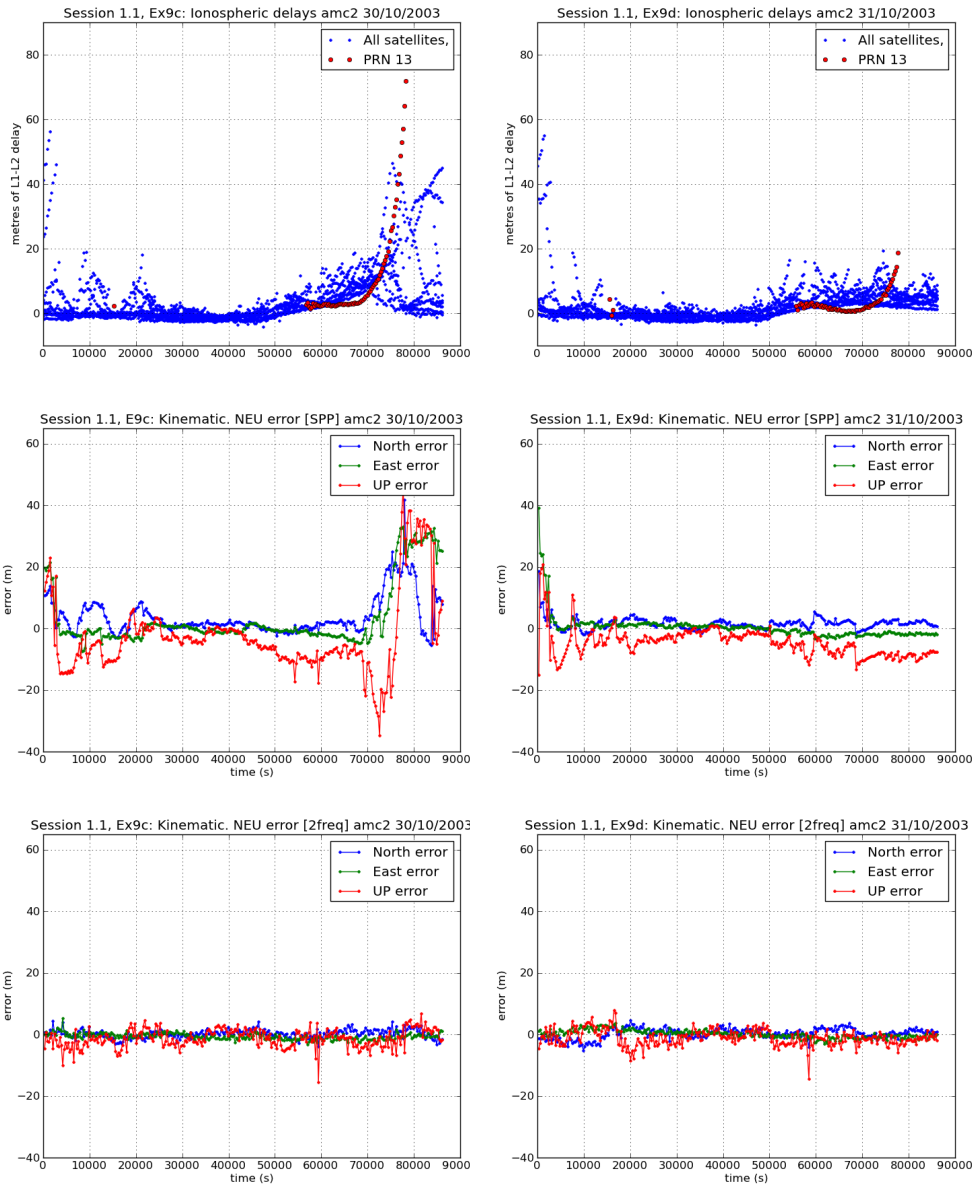


Figure 1.20: Plots of 30 (left column) and 31 (right column) October 2003: code ionospheric delays (first row); single-frequency positioning accuracy (second row); dual-frequency positioning accuracy (third row).

(a) Process the data file in SPP mode, with data decimation of 30 s (to better see the effect).[14] Then:

i. Set gLAB in the SPP processing mode by clicking `SPP Template`.

ii. In the `[Preprocess]` section, set the `[☒ Data Decimation]` to `30` seconds (instead of the 300 seconds default value). See Fig. 1.21, first row on the right.

iii. In the `[Output]` section, uncheck all the message printings, except the last one (i.e. `[☒ Print OUTPUT messages]`). This is done in order to save time in writing the output file. See Fig. 1.21, second row on the left.

iv. Produce a NEU plot for the time interval $71\,500 < t < 73\,500$ GPS seconds of day (i.e. 19:51:40 to 20:25:00 GPS time).

[14] The data file `upc31530.06o` was collected at a sampling rate of 1 Hz.

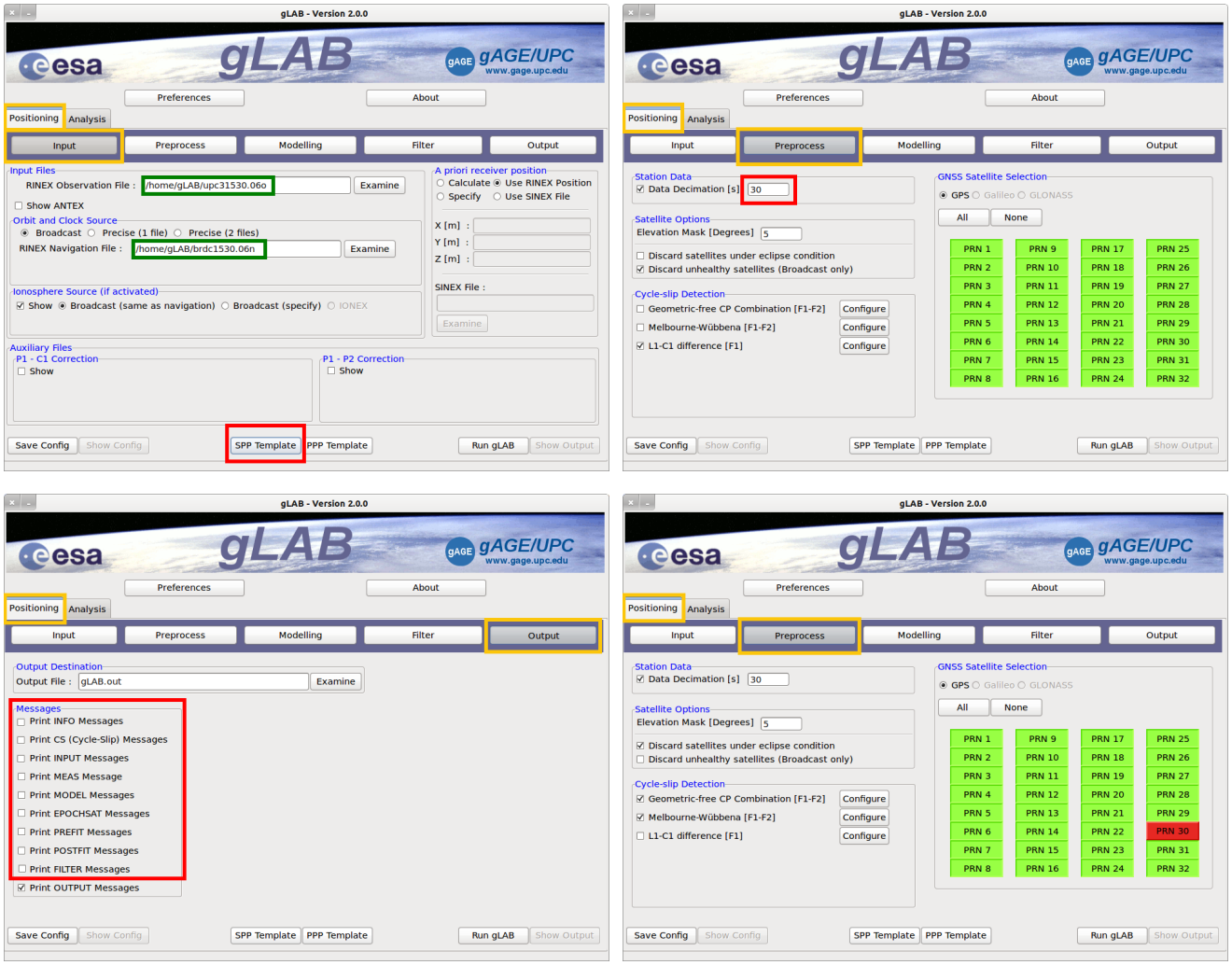


Figure 1.21: gLAB panels. First row left: setting the SPP Template mode and selecting the input files upc31530.06o and brdc1530.06n. First row right: setting the [☒ data Decimation] to 30 seconds. Second row left: unchecking all output messages, except [☒ Print OUTPUT messages]. Second row right: unchecking PRN30.

v. Produce a NEU plot for the time interval $72\,000 < t < 76\,000$ GPS seconds of day.

Note that a second anomaly on clock PRN30 was experienced around 75 400 s (see Fig. 1.22, first row on the right).

(b) Repeat the previous process, unchecking satellite PRN30 in the `[Preprocess]` section. See Fig. 1.21, second row on the right.

11. Processing roving receivers (flight trial)

The data file `nacc080a.09o` contains measurements collected by a receiver onboard an aircraft flying over the Pyrenees, close to the Mediterranean Sea, along the border between Spain and France.

Unlike the previous exercises, where the data files were collected with a permanent receiver with fixed coordinates, in this case the receiver is onboard an aircraft in flight. Thus, the navigation solution will provide a trajectory (i.e. the aircraft's flight path).

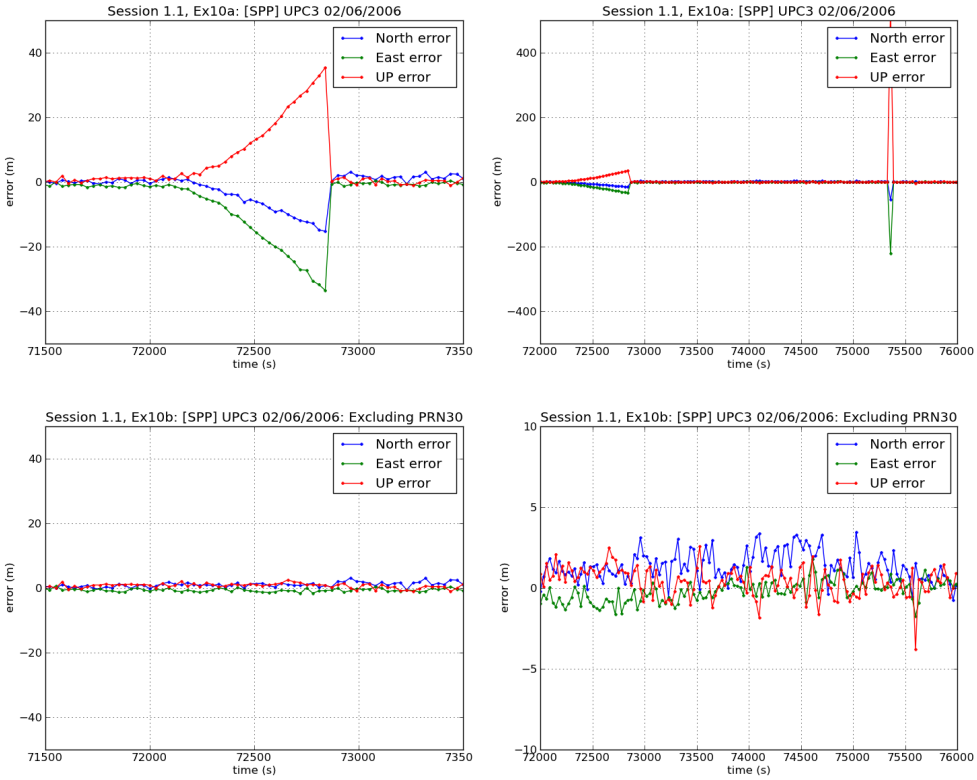


Figure 1.22: PRN30 clock anomaly, effect on the NEU positioning error (the left-hand plots are magnifications of right-hand plots). The first row shows the results using PRN30 and the second row excluding PRN30.

Figure 1.23: gLAB panels. Configuration to process trajectories with the option [⊙ Calculate] in the 'A priori receiver position' box set.

Consequently, the '`A priori receiver position`', applied by default in the previous process (with static receiver data), cannot be used. In this case, `gLAB` has to compute a trajectory by continuously updating the a priori receiver position.[15] This is done in `gLAB` by selecting the option `[⊙ Calculate]` in the `A priori receiver position` box of the `[Input]` section (see Fig. 1.23).

Then, following the same procedure as in the previous exercises, but setting `gLAB` to compute trajectories as indicated before, process the data file `nacc080a.09o` in `SPP Template` mode. The broadcast orbits and clocks are in file `brdc0800.09n`.

Note: Uncheck `[□ Data decimation]` *in the* `[Preprocess]` *section to have the solutions at one second rate.*

After processing the measurements, the aircraft's height can be plotted as a function of time as follows:

In the `[Analysis]` tab, click the `Clear` button, select `gLAB.out` as the `Source File` and set the following options:

Condition:	OUTPUT	▼		
X Column:	SEC	▼	4	▼
Y Column:	STAHGT	▼	17	▼

Finally, click the button `Plot`. See plot on page 29, last row on the left.

Moreover, the trajectory can be viewed, for instance, with `Google Earth` as follows (see Fig. 1.24):

(a) Select fields 16 (longitude), 15 (latitude) and 17 (height) from the message labelled as `OUTPUT` in the output file `gLAB.out` and create a data file (for instance, `track.kml`).

(b) Add a 'header file' `Prefix.kml` and an 'ending file' `Postfix.kml` before and after the coordinate file `track.kml`.

(c) Open the resulting file with `Google Earth`.

The previous process can be done (in a script file in a Linux environment, for instance) by executing the following instructions (see the computing tools and skills in laboratory session 2.1):

```
cat Prefix.kml > track.kml

grep OUTPUT gLAB.out | grep -v INFO |
 gawk 'BEGIN{OFS=","}{print $16,$15,$17}' >> track.kml

cat  Postfix.kml >> track.kml
```

[15] As explained in section 6.1, Volume I, these 'a priori' coordinates are used to linearise the navigation equations. For moving receivers, this point is iteratively computed starting from an arbitrary solution (e.g., Earth's centre) and continuously updated along the trajectory. Of course, usage of 'a priori' values in the case of permanent receivers allows the computation load to be reduced.

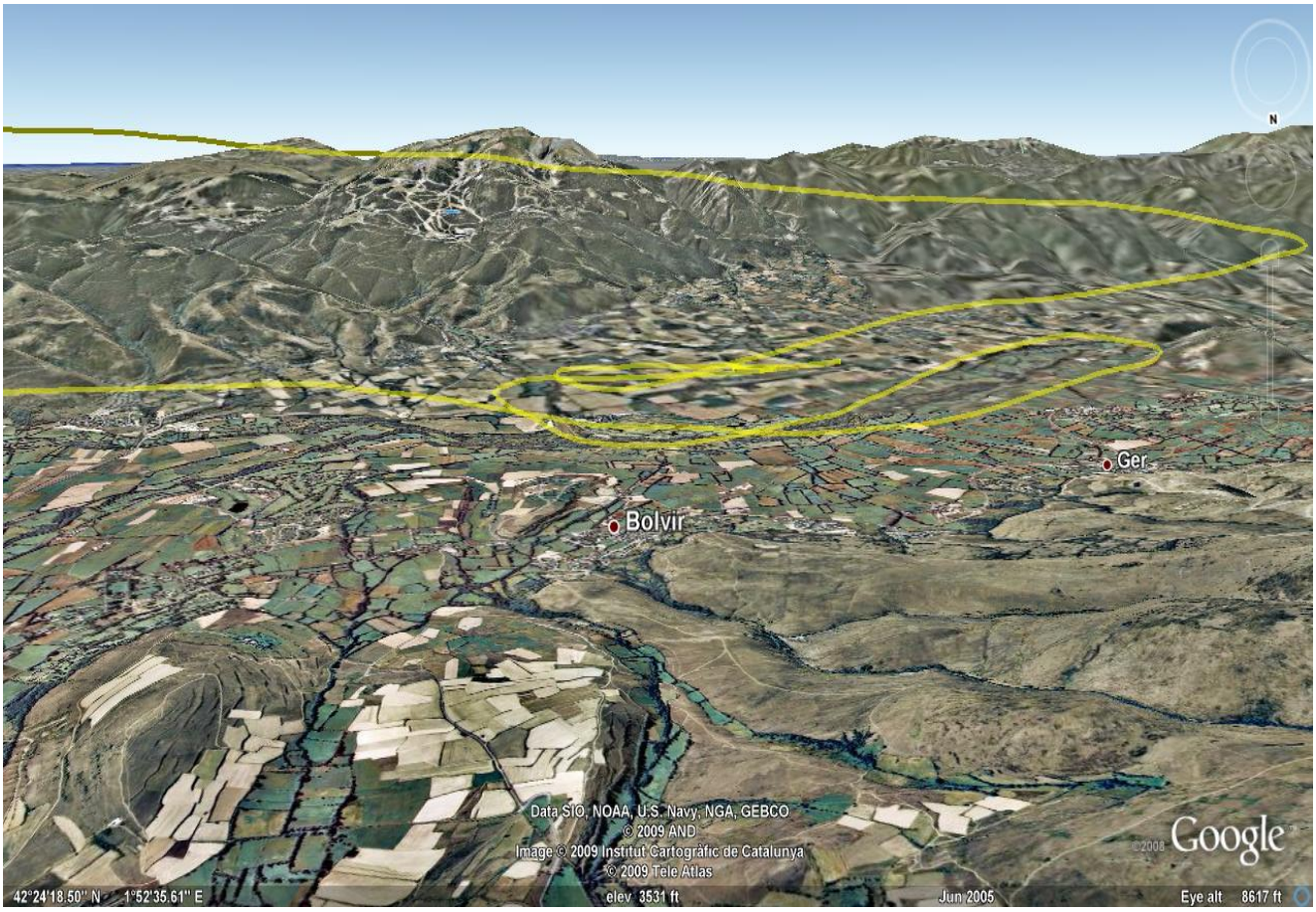
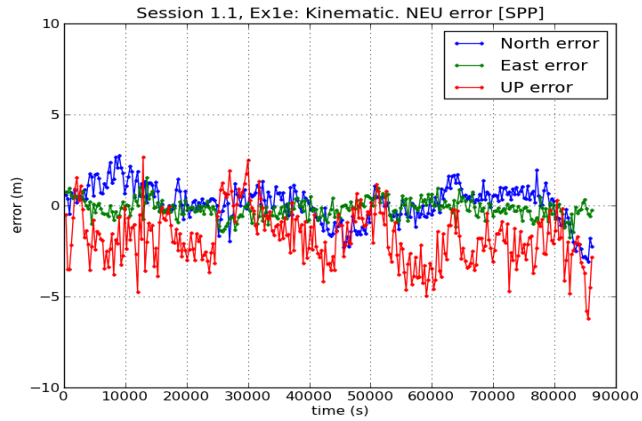
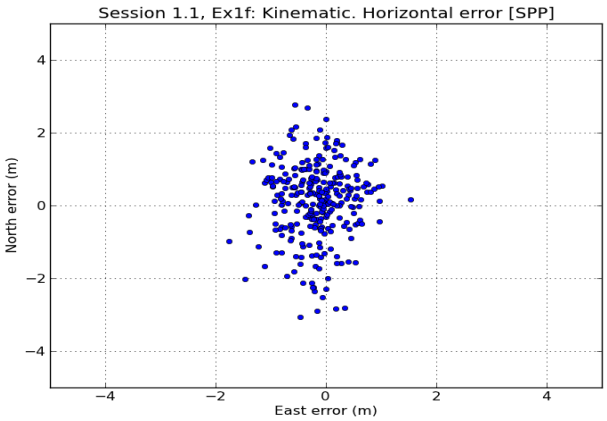
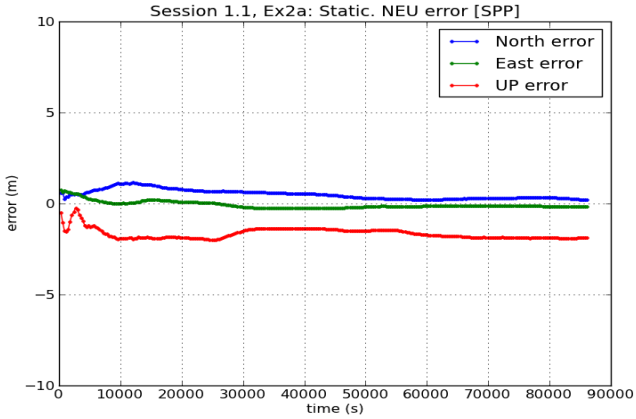


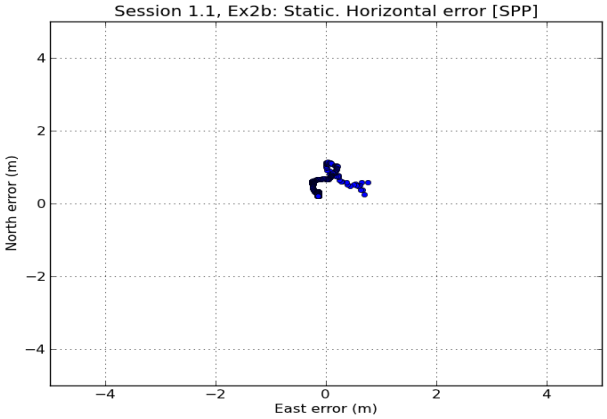
Figure 1.24: Solution trajectory of the flight trial of the measurements file nacc080a.09o.

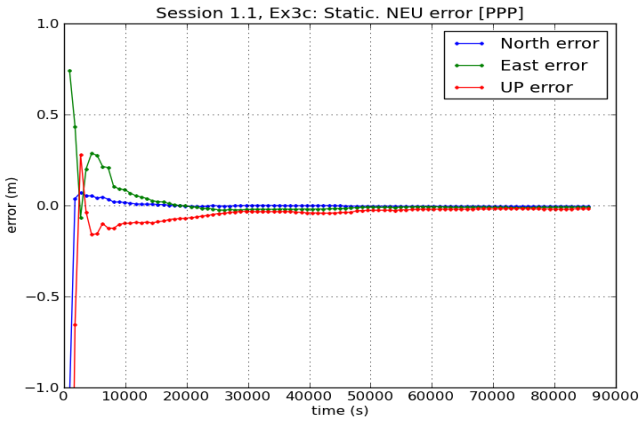
Graphs Session 1.1

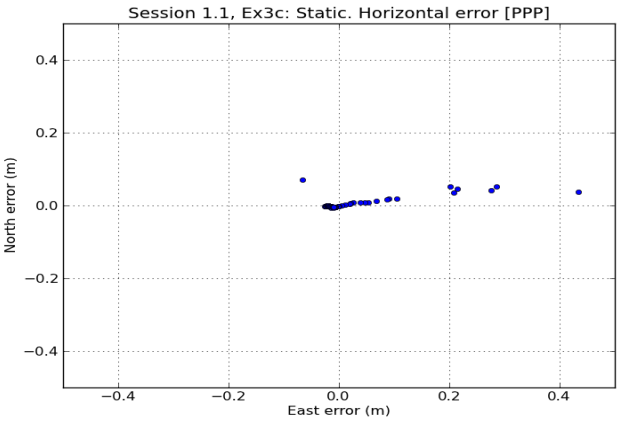


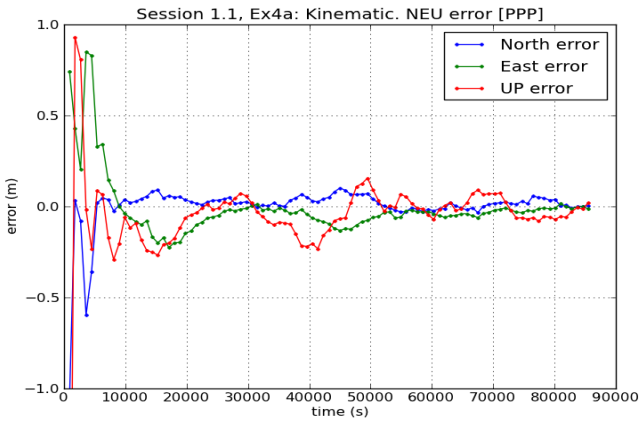


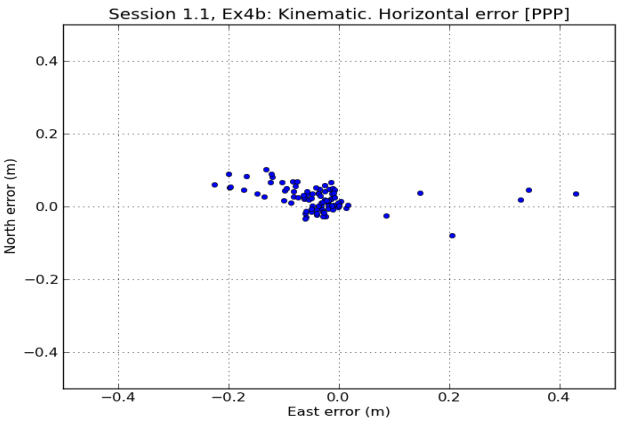


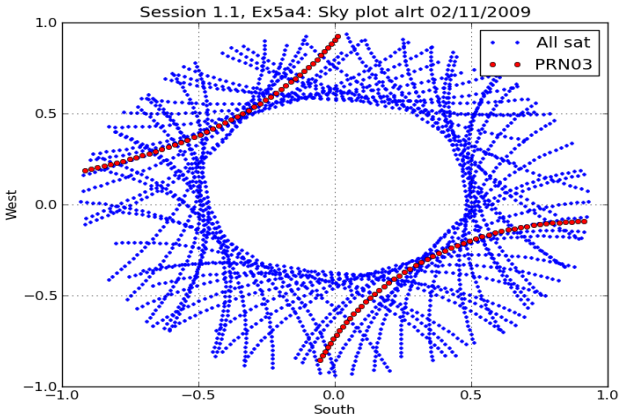


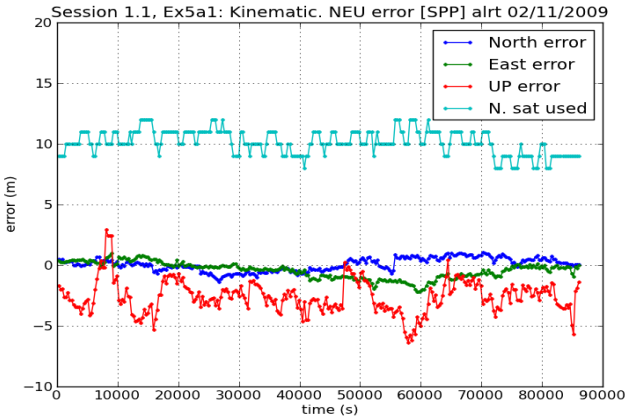


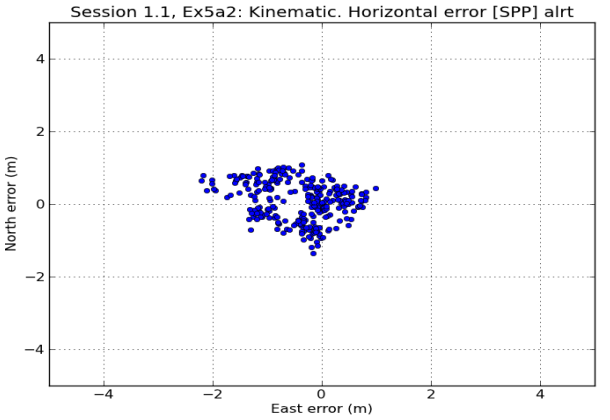


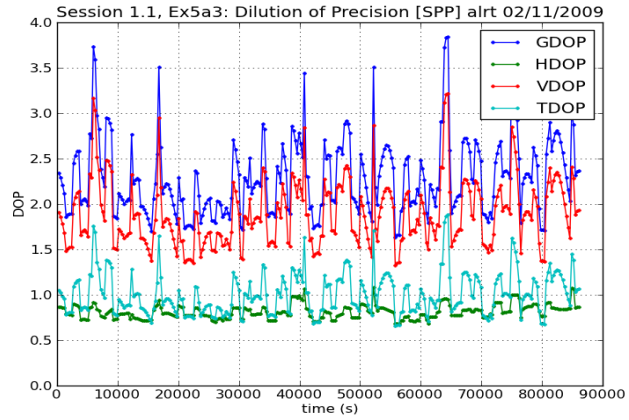


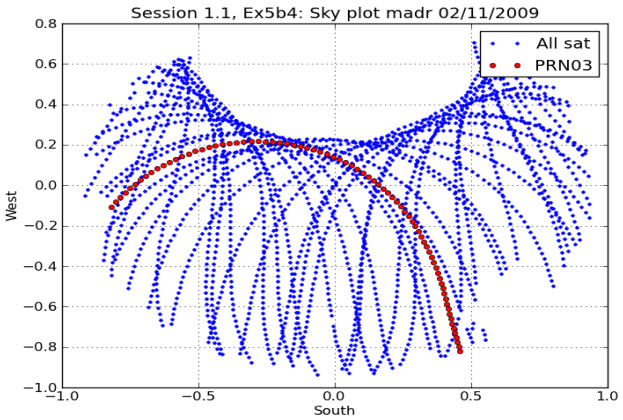


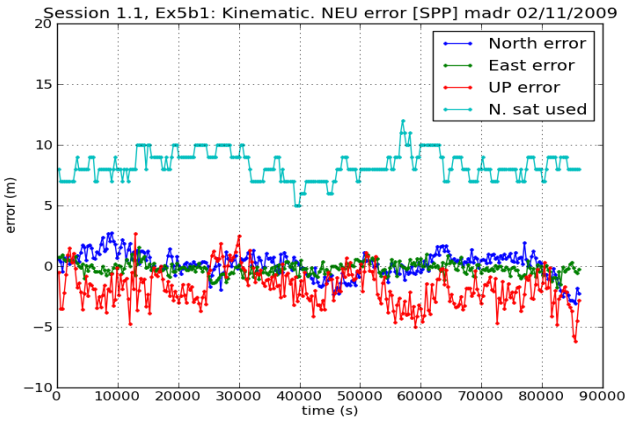

Session 1.1, Ex5a4: Sky plot alrt 02/11/2009
All sat
PRN03
West
South

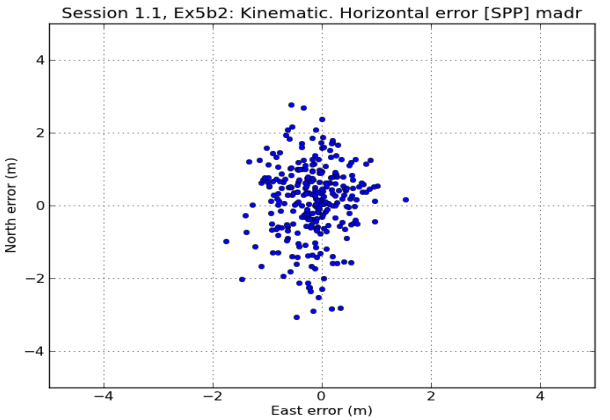

Session 1.1, Ex5a1: Kinematic. NEU error [SPP] alrt 02/11/2009
North error
East error
UP error
N. sat used
error (m)
time (s)

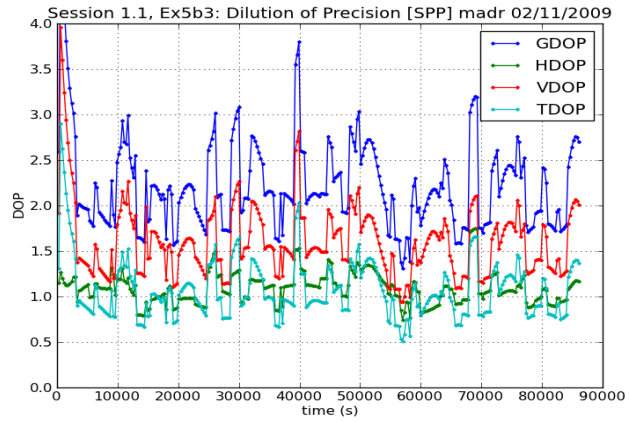

Session 1.1, Ex5a2: Kinematic. Horizontal error [SPP] alrt
North error (m)
East error (m)

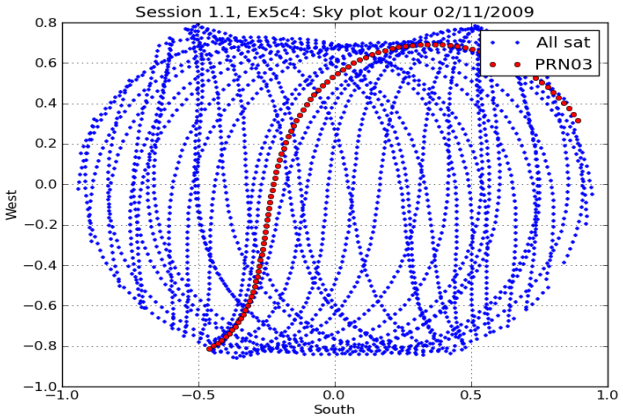

Session 1.1, Ex5a3: Dilution of Precision [SPP] alrt 02/11/2009
GDOP
HDOP
VDOP
TDOP
DOP
time (s)

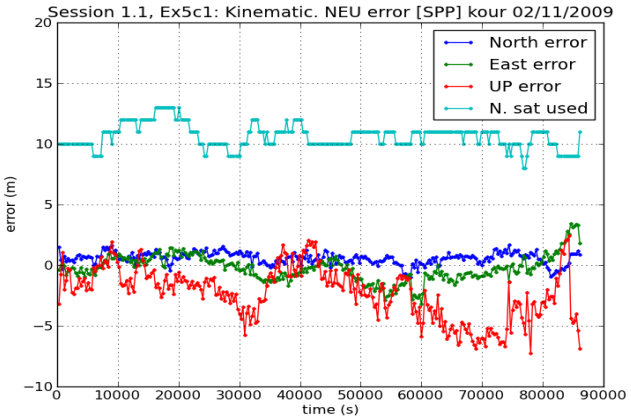

Session 1.1, Ex5b4: Sky plot madr 02/11/2009
All sat
PRN03
West
South

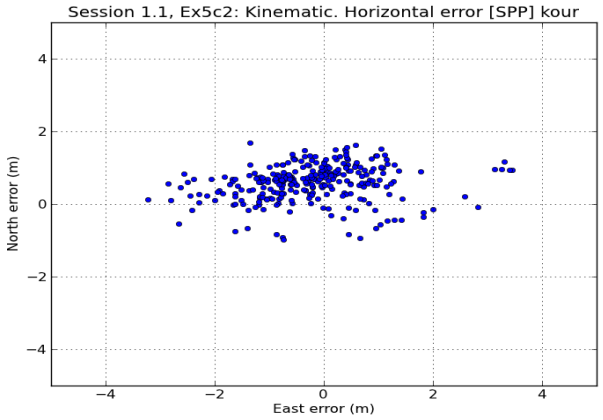

Session 1.1, Ex5b1: Kinematic. NEU error [SPP] madr 02/11/2009
North error
East error
UP error
N. sat used
error (m)
time (s)

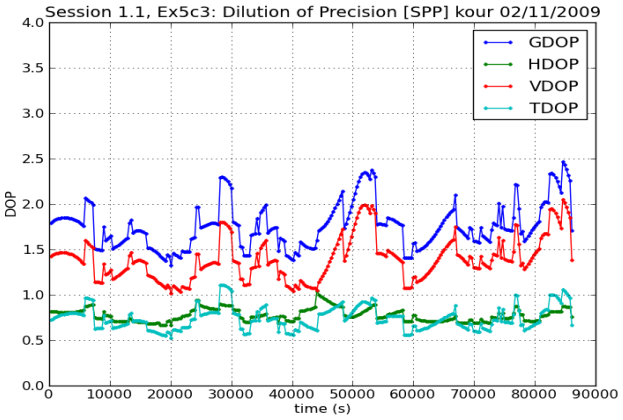

Session 1.1, Ex5b2: Kinematic. Horizontal error [SPP] madr
North error (m)
East error (m)

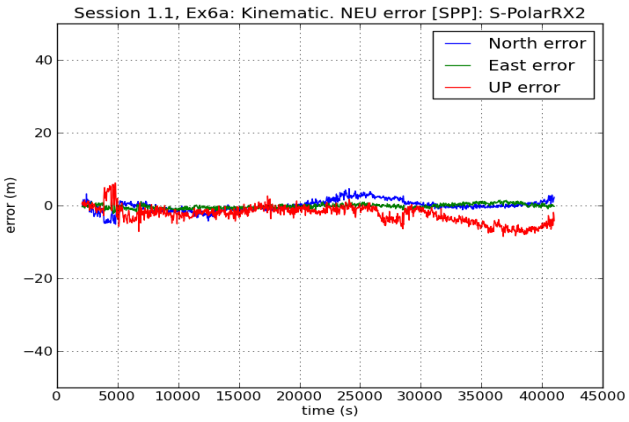

Session 1.1, Ex5b3: Dilution of Precision [SPP] madr 02/11/2009
GDOP
HDOP
VDOP
TDOP
DOP
time (s)

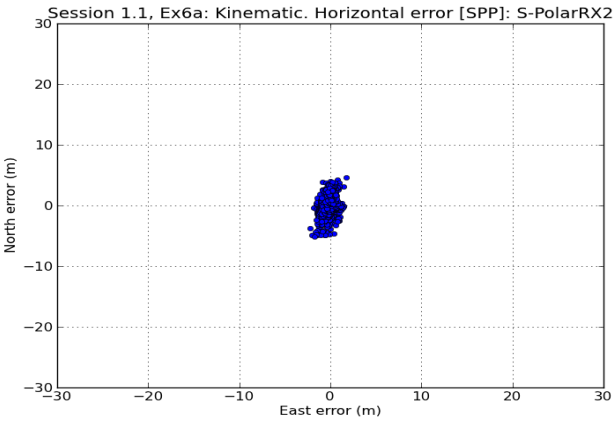

Session 1.1, Ex5c4: Sky plot kour 02/11/2009
All sat
PRN03
West
South

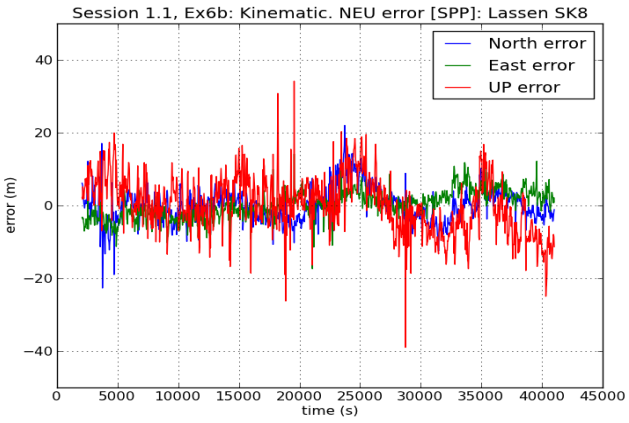

Session 1.1, Ex5c1: Kinematic. NEU error [SPP] kour 02/11/2009
North error
East error
UP error
N. sat used
error (m)
time (s)

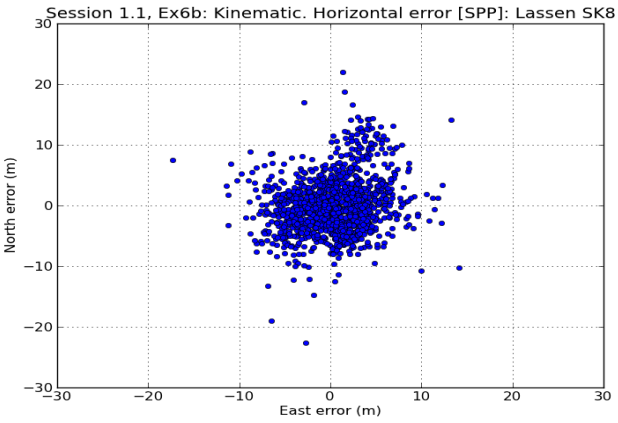

Session 1.1, Ex5c2: Kinematic. Horizontal error [SPP] kour
North error (m)
East error (m)

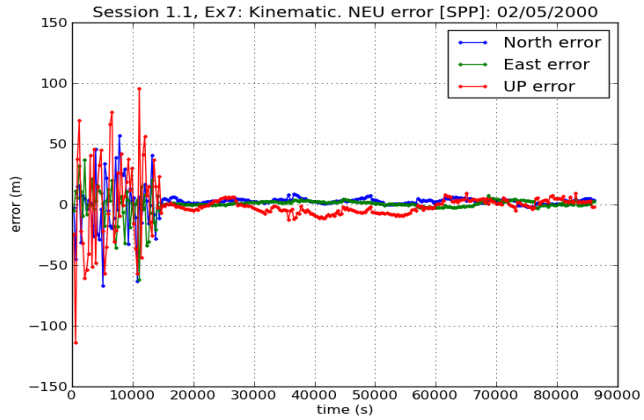

Session 1.1, Ex5c3: Dilution of Precision [SPP] kour 02/11/2009
GDOP
HDOP
VDOP
TDOP
DOP
time (s)

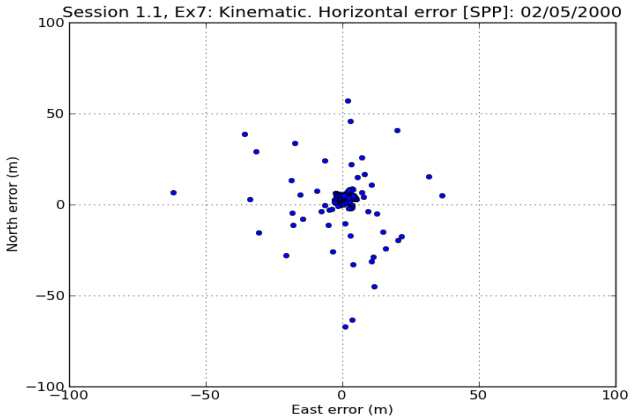

Session 1.1, Ex6a: Kinematic. NEU error [SPP]: S-PolarRX2
North error
East error
UP error
error (m)
time (s)

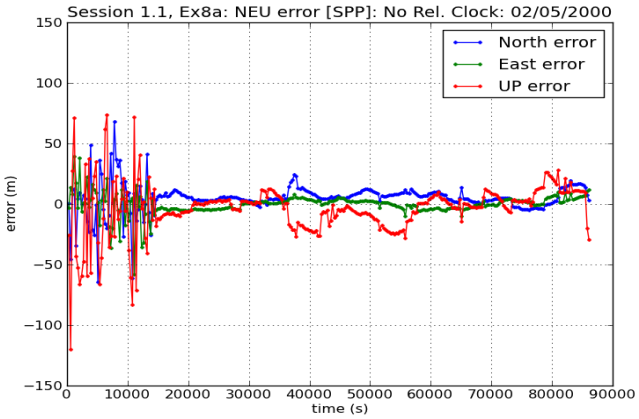

Session 1.1, Ex6a: Kinematic. Horizontal error [SPP]: S-PolarRX2
North error (m)
East error (m)

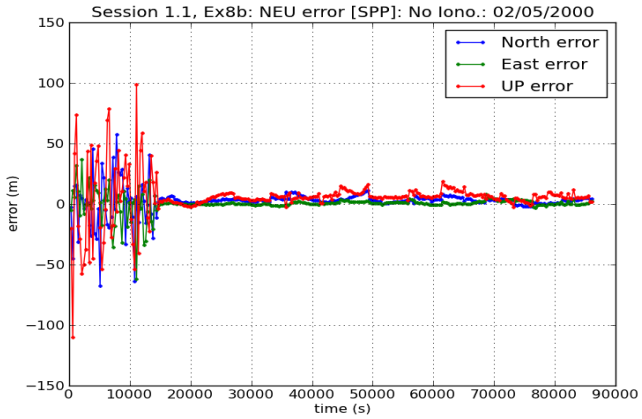

Session 1.1, Ex6b: Kinematic. NEU error [SPP]: Lassen SK8
North error
East error
UP error
error (m)
time (s)

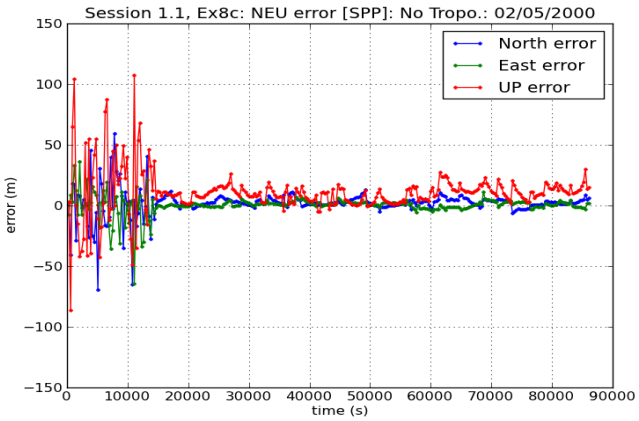

Session 1.1, Ex6b: Kinematic. Horizontal error [SPP]: Lassen SK8
North error (m)
East error (m)

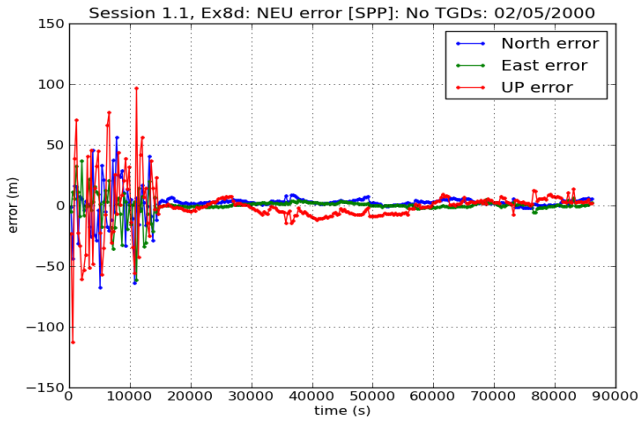

Session 1.1, Ex7: Kinematic. NEU error [SPP]: 02/05/2000
North error
East error
UP error
error (m)
time (s)

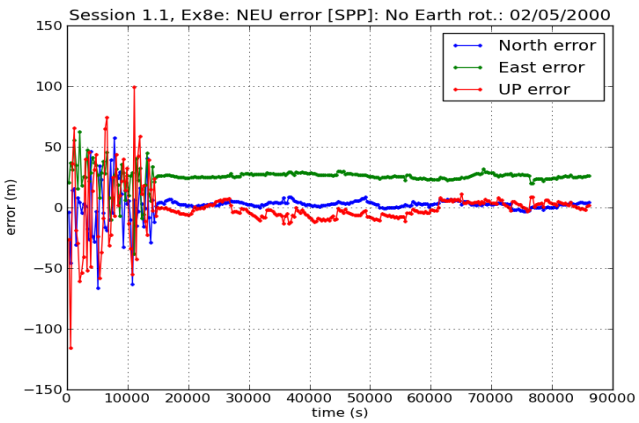

Session 1.1, Ex7: Kinematic. Horizontal error [SPP]: 02/05/2000
North error (m)
East error (m)

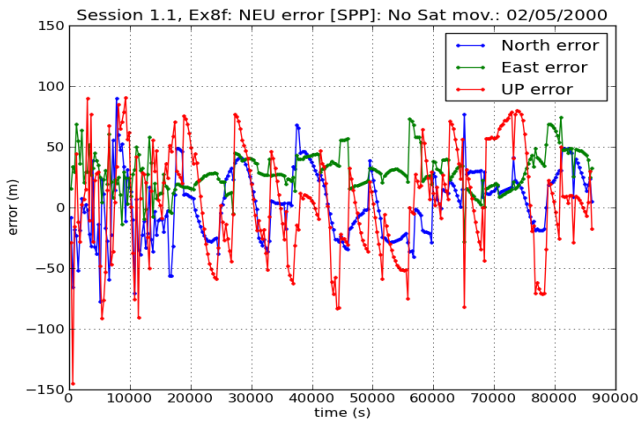

Session 1.1, Ex8a: NEU error [SPP]: No Rel. Clock: 02/05/2000
North error
East error
UP error
error (m)
time (s)

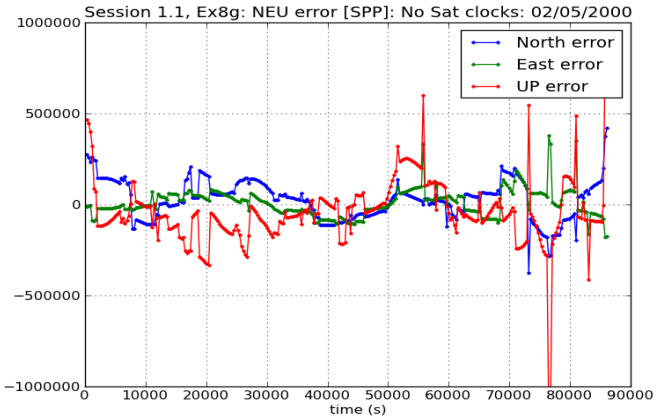

Session 1.1, Ex8b: NEU error [SPP]: No Iono.: 02/05/2000
North error
East error
UP error
error (m)
time (s)

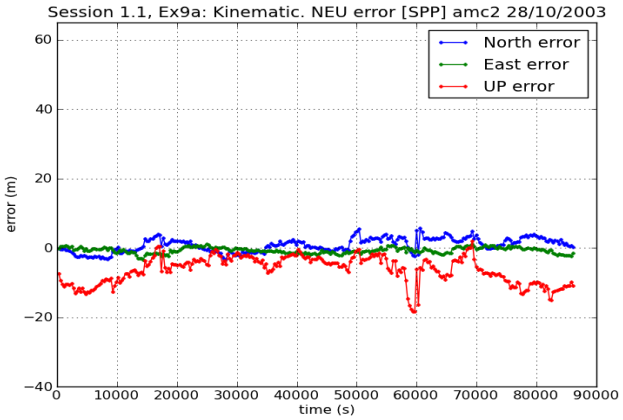

Session 1.1, Ex8c: NEU error [SPP]: No Tropo.: 02/05/2000
North error
East error
UP error
error (m)
time (s)

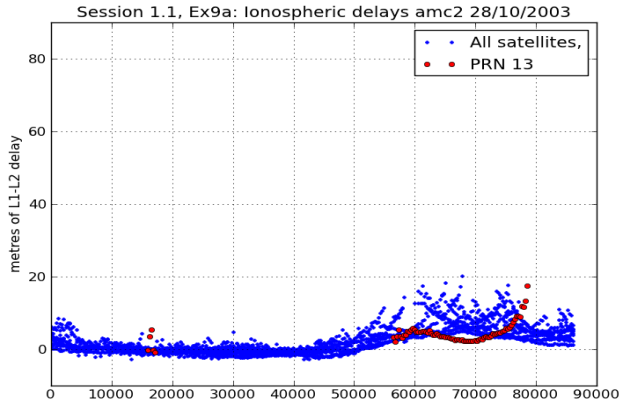

Session 1.1, Ex8d: NEU error [SPP]: No TGDs: 02/05/2000
North error
East error
UP error
error (m)
time (s)

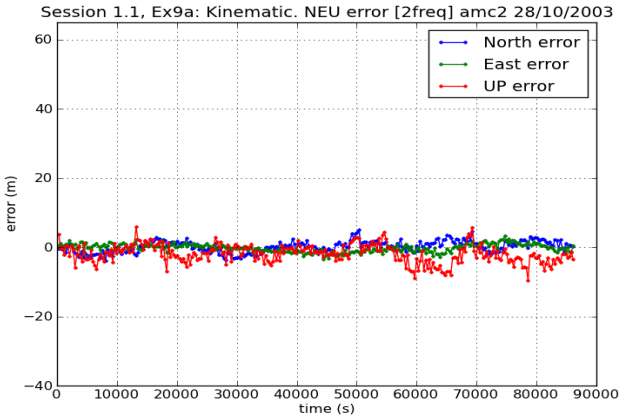

Session 1.1, Ex8e: NEU error [SPP]: No Earth rot.: 02/05/2000
North error
East error
UP error
error (m)
time (s)

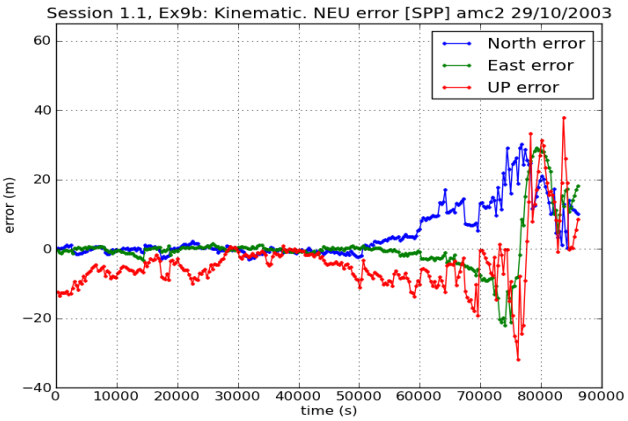

Session 1.1, Ex8f: NEU error [SPP]: No Sat mov.: 02/05/2000
North error
East error
UP error
error (m)
time (s)

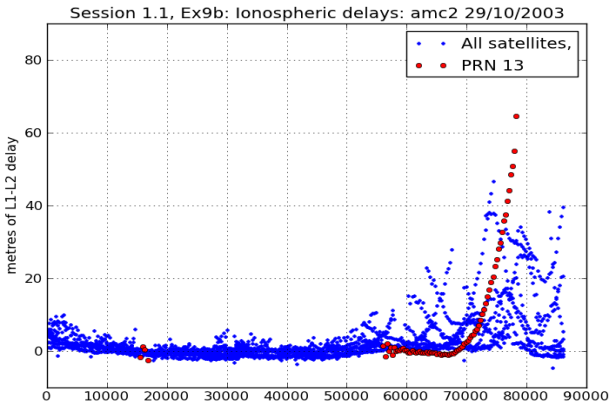

Session 1.1, Ex8g: NEU error [SPP]: No Sat clocks: 02/05/2000
North error
East error
UP error
time (s)

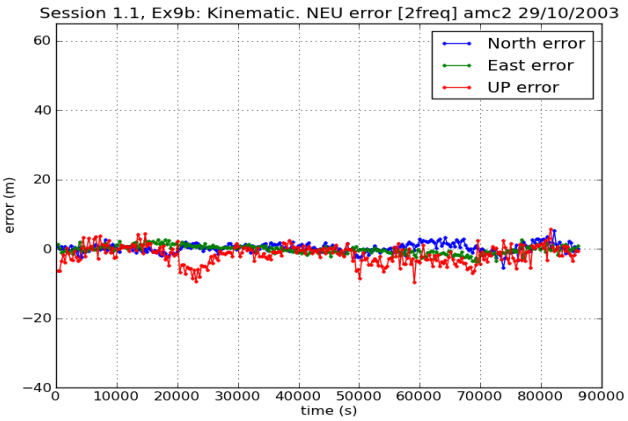

Session 1.1, Ex9a: Kinematic. NEU error [SPP] amc2 28/10/2003
North error
East error
UP error
error (m)
time (s)

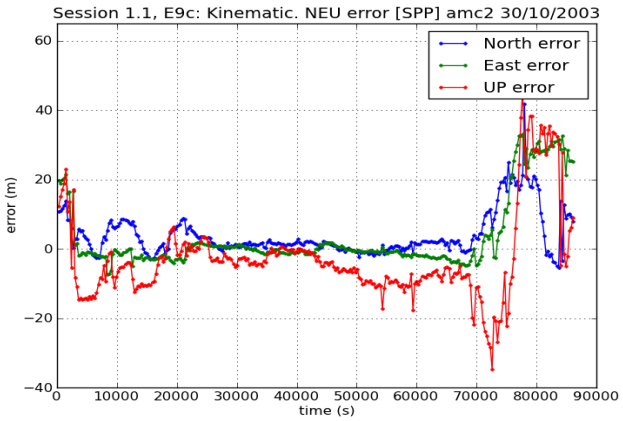

Session 1.1, Ex9a: Ionospheric delays amc2 28/10/2003
All satellites,
PRN 13
metres of L1-L2 delay

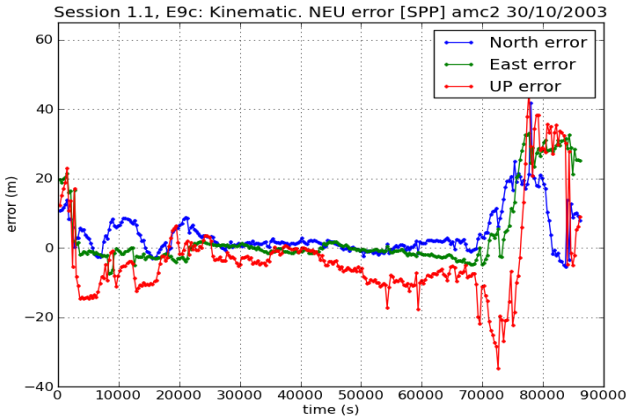

Session 1.1, Ex9a: Kinematic. NEU error [2freq] amc2 28/10/2003
North error
East error
UP error
error (m)
time (s)

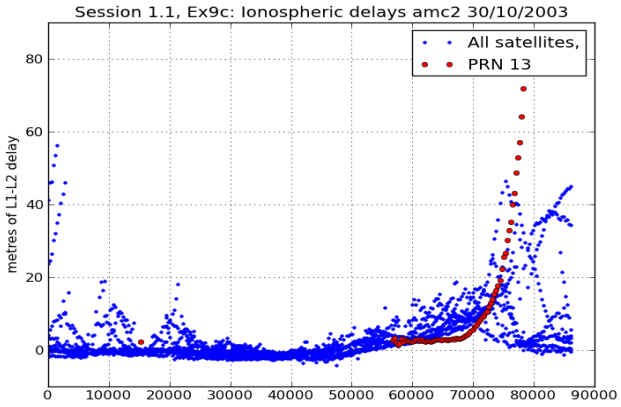

Session 1.1, Ex9b: Kinematic. NEU error [SPP] amc2 29/10/2003
North error
East error
UP error
error (m)
time (s)

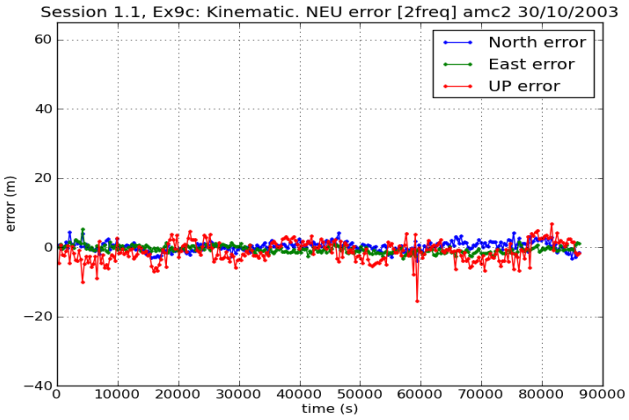

Session 1.1, Ex9b: Ionospheric delays: amc2 29/10/2003
All satellites,
PRN 13
metres of L1-L2 delay

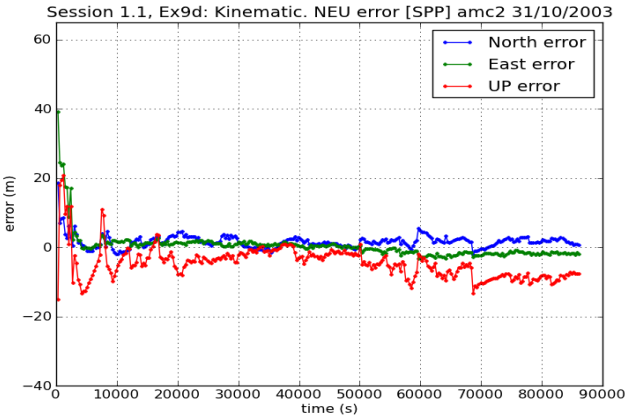

Session 1.1, Ex9b: Kinematic. NEU error [2freq] amc2 29/10/2003
North error
East error
UP error
error (m)
time (s)

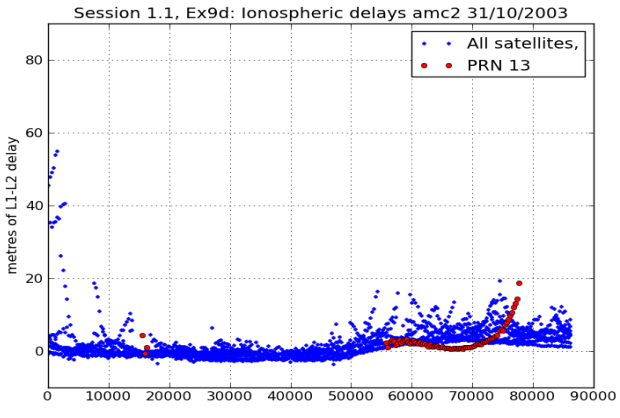

Session 1.1, E9c: Kinematic. NEU error [SPP] amc2 30/10/2003
North error
East error
UP error
error (m)
time (s)

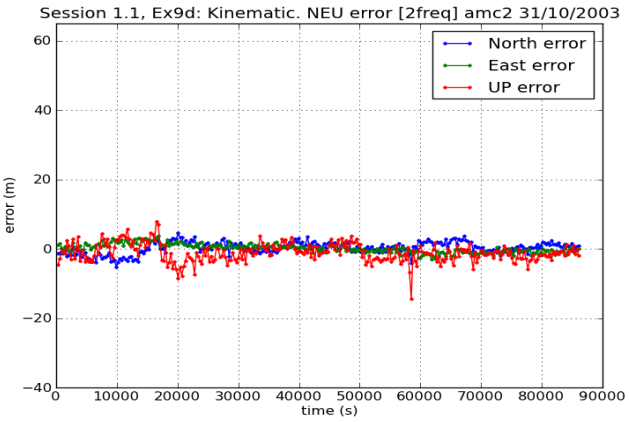

Session 1.1, E9c: Kinematic. NEU error [SPP] amc2 30/10/2003
North error
East error
UP error
error (m)
time (s)

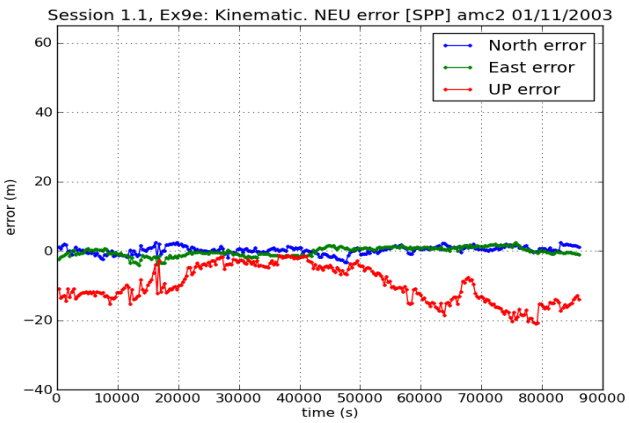

Session 1.1, Ex9c: Ionospheric delays amc2 30/10/2003
All satellites,
PRN 13
metres of L1-L2 delay

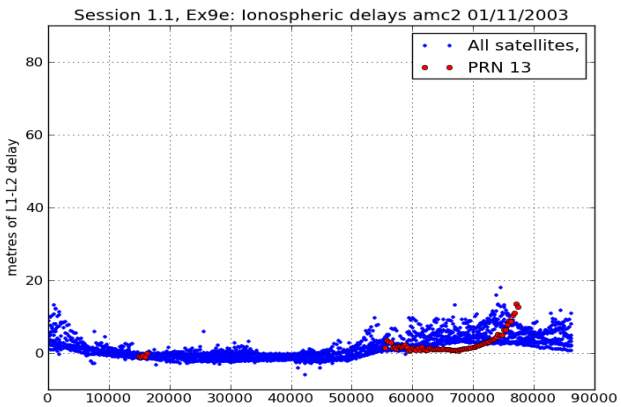

Session 1.1, Ex9c: Kinematic. NEU error [2freq] amc2 30/10/2003
North error
East error
UP error
error (m)
time (s)

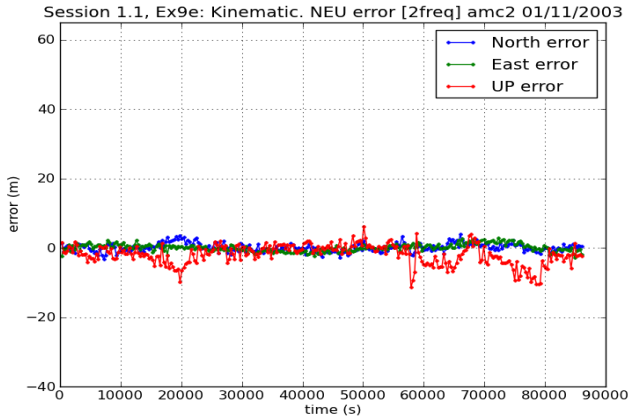

Session 1.1, Ex9d: Kinematic. NEU error [SPP] amc2 31/10/2003
North error
East error
UP error
error (m)
time (s)

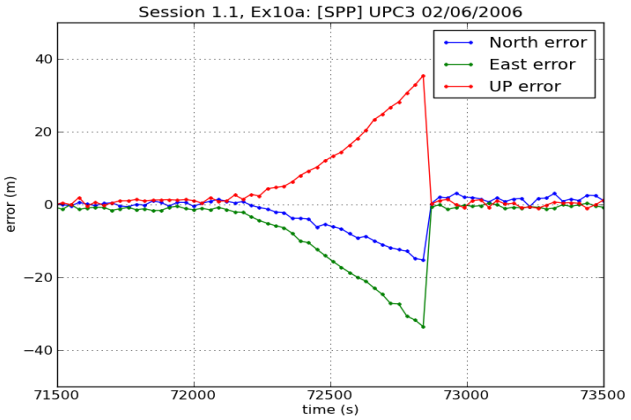

Session 1.1, Ex9d: Ionospheric delays amc2 31/10/2003
All satellites,
PRN 13
metres of L1-L2 delay

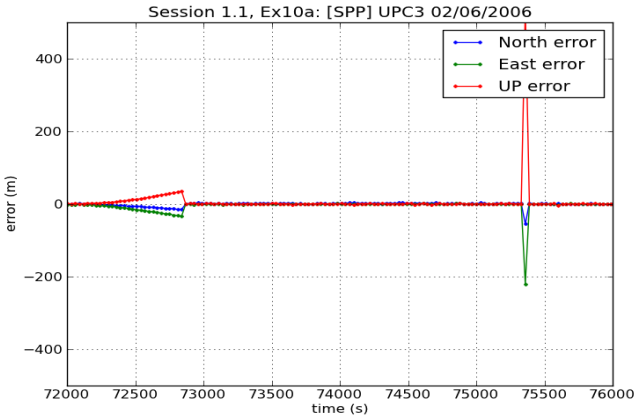

Session 1.1, Ex9d: Kinematic. NEU error [2freq] amc2 31/10/2003
North error
East error
UP error
error (m)
time (s)

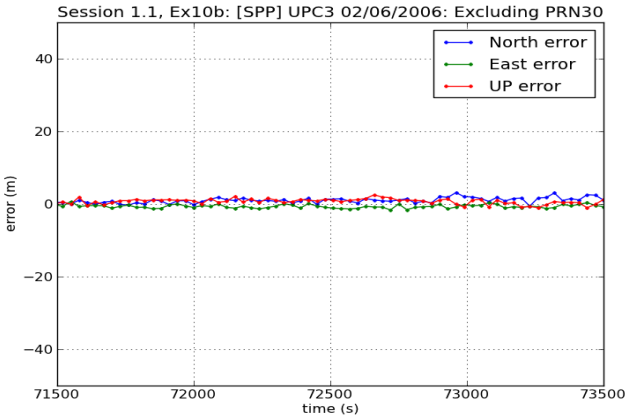

Session 1.1, Ex9e: Kinematic. NEU error [SPP] amc2 01/11/2003
North error
East error
UP error
error (m)
time (s)

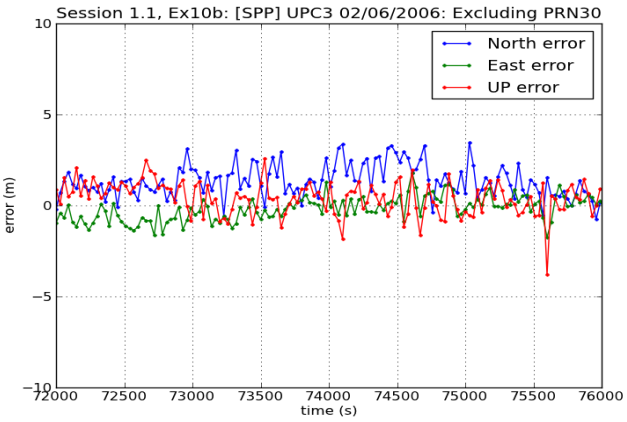

Session 1.1, Ex9e: Ionospheric delays amc2 01/11/2003
All satellites,
PRN 13
metres of L1-L2 delay

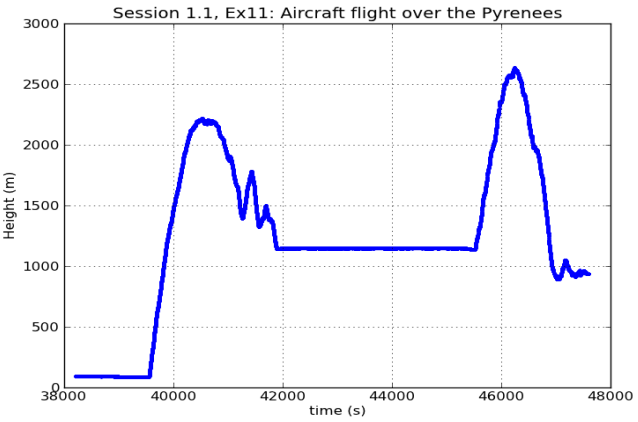

Session 1.1, Ex9e: Kinematic. NEU error [2freq] amc2 01/11/2003
North error
East error
UP error
error (m)
time (s)


Session 1.1, Ex10a: [SPP] UPC3 02/06/2006
North error
East error
UP error
error (m)
time (s)


Session 1.1, Ex10a: [SPP] UPC3 02/06/2006
North error
East error
UP error
error (m)
time (s)


Session 1.1, Ex10b: [SPP] UPC3 02/06/2006: Excluding PRN30
North error
East error
UP error
error (m)
time (s)


Session 1.1, Ex10b: [SPP] UPC3 02/06/2006: Excluding PRN30
North error
East error
UP error
error (m)
time (s)


Session 1.1, Ex11: Aircraft flight over the Pyrenees
Height (m)
time (s)

2. Laboratory Environment and Data Files

As explained in the Preface, the laboratory sessions of this book are developed on a UNIX (Linux) Operating System (OS).[1] Therefore, an introductory chapter on the UNIX environment and software tools has been considered useful to give the reader a basis for the development of the exercises. Among the tools, additional background on the GNSS standard file formats is also provided as complementary information. Readers with some knowledge of UNIX (Linux) OS and GNSS data file formats can skip sessions 2.1 and 2.2.

The first session introduces the user to the Linux environment. It provides a reduced set of commands to start work in this OS. Moreover, it explains the use and some capabilities of the `awk/gawk` programming environment, which is one of the basic tools used in the laboratory exercises.

Plotting tools are also given in this session. The `graph.py` utility is introduced as the baseline tool for the laboratory exercises. This tool is based on Python libraries and is distributed jointly with `gLAB`.

The second session is devoted to a description of the standard formats used in the GNSS data files. The aim is to familiarise the reader with such standards in a very friendly way. This is done with the help of a set of HyperText Markup Language (HTML) files and explanatory tool tips. A wide variety of questions are posed to the reader to guide the learning of the standards format and to draw attention to different specific issues.

[1] Nevertheless, the Windows OS can be used as well, if preferred. In this case, it is necessary to install the `awk` (or `gawk`) tool and to compile the programs in the Windows environment. A `gLAB` installable version for Windows is readily available from *http://www.gage.upc.edu/glab*.

Session 2.1. UNIX Environment, Tools and Skills

Objectives

To present a (very limited) set of UNIX instructions in order to manage files and directories, as well as some basic elements of `awk/gawk` programming and the graphical plotting environment `graph.py`. The aim is not to teach UNIX or programming languages, but to provide some basic tools needed to develop the practical sessions.

Note: The following exercises are very elementary and can be omitted if the reader already has some basic knowledge of UNIX and `gawk`.

Files to use
`sxyz.eci`

Programs to use
`graph.py`

Development

This session has been organised as a series of guided exercises to be done in the established order, which introduces the main instructions for use in the following sessions.

1. First instructions

(a) Show the name of the directory where you are located.
Execute: `pwd`

(b) Examine the directory content.
Execute: `ls -lt`

(c) Go to the personal directory or `home` directory ('∼').
Execute: `cd` or `cd ∼`

(d) Go to the `GNSS` directory (inside the `home` directory)[2]
Execute: `cd GNSS`

(e) Access the `HTML` directory and view its contents.
Execute:

```
cd HTML
ls -lt
```

(f) Go back to the `home` directory.
Execute: `cd ∼`

[2] If the installation has been done properly according to instructions in the installation guide, the following three directories will be found: `FILES`, `PROG` and `HTML`. These directories, together with the Notepad files, will appear in the `home/GNSS` directory.

(g) Show a text line on the screen:
Execute:

```
echo "Have a nice day"
```

(h) Direct the contents to a file:
Execute:

```
echo "Have a nice day" > test
ls -ltr
echo "you too" >> test
```

(i) Show the contents of the file on the screen:
Execute:

```
cat test
```

Try to execute this too: `echo test`. What happens?

(j) Edit a file
Execute:[3]

```
gedit test
```

2. Directory management

(a) From any directory where you are located, go to the `GNSS` directory and check that you are in it. Create the directory `working` inside the `GNSS` directory. Access it. Go back to the directory immediately above (in this case the `GNSS` directory) by executing '`..`').
Execute:

```
cd ~/GNSS
pwd
mkdir working
cd working
pwd
cd ..
pwd
```

3. File management

(a) Go to the `working` directory (which is inside the `GNSS` directory). Copy the file `test` from the `home` directory ('~') to your current directory (symbolised by '`.`').
Execute:

```
cd ~/GNSS/working
cp ~/test .
ls -lt
```

(b) Copy the file `test` to the file `file1`.[4] Check the contents of `file1`.
Execute:

```
cp test file1
ls -lt
more file1
```

[3] Any text editor can also be used.

[4] As file `file1` does not exist, a new file will be created with this name and with the same contents as file `test`.

(c) Create a 'link'[5] from file `file2` to file `test`. Check the file contents. Check the contents of `file2`.
Execute:

```
ln -s test file2
ls -lt
more file2
```

(d) Using a text editor (e.g. `gedit` or similar), edit `file2` and change the word *day* to the word *YEAR*. Save the changes and exit `gedit`. Next, check the contents of file `test` and its link `file2`. Have the contents of the original file `test` been modified through its link `file2`?
Execute:

```
gedit file2
more test
more file1
```

(e) Remove file `file1` and its link `file2`. Check if they have been deleted. Create the directory `other`. Remove the directory `other`.
Execute:

```
ls -lt
rm file1 file2
ls -lt
mkdir other
ls -lt
rm -r other
ls -lt
```

(f) Find information on the '`mkdir`' and '`rm`' commands in the help pages (i.e. in the UNIX manual).
Execute:

```
man mkdir
man rm
```

4. Programming environment `gawk`[6]

(a) From within the working directory, create a link from file `sxyz.eci` (which is placed in the directory `GNSS/FILES/SES21`) to a file with the same name in the working directory.
Execute:

```
cd ~/GNSS/working
ln -s ~/GNSS/FILES/SES21/sxyz.eci .
ls -lt
```

The file `sxyz.eci` contains three columns of coordinates, in a geocentric inertial system, of a set of satellites at different epochs. It contains the following fields:

```
SATELLITE time(sec) X(km) Y(km) Z(km)
```

[5] This differs from the previous case because `file2` is not a new file, just a pointer to file `test`. Thus, the 'link' `file2` represents the minimum space expense, independent of the file `test` size. Execute `man ln` to see the meaning of different types of links.

[6] `gawk` is the GNU implementation of `awk` (from the Free Software Foundation).

(b) Execute the instructions `cat`, `more` and `less` in order to display the file contents `sxyz.eci`. What differences can be seen among these instructions?[7]
Execute:

```
cat sxyz.eci
more sxyz.eci
less sxyz.eci
cat sxyz.eci | less
```

(c) Using the programming language `gawk`, print (on screen) the first and third fields of file `sxyz.eci`.
Execute:

```
gawk '{print $1,$3}' sxyz.eci |more
or
cat sxyz.eci |gawk '{print $1,$3}' |more
```

(d) Now print all the fields at the same time.
Execute:

```
cat sxyz.eci | gawk '{print $0}' | more
```

(e) The following instruction generates the file `prb1` which contains data from a single satellite. Which satellite is selected?
Execute:

```
cat sxyz.eci | gawk '{if ($1==5) print $0 }' > prb1
more prb1
```

(f) What is the meaning of the values in the second column of file `prb2` generated by the next instruction?
Execute (in a single line):[8]

```
cat sxyz.eci | gawk '{if ($1==5)
           print $2,sqrt($3**2+$4**2+$5**2)}'> prb2
more prb2
```

(g) Discuss the structure of the following instruction that makes a 'print' with a particular format (where `%i`=integer, `%f`= float, `%s`= string).
Execute:

```
cat sxyz.eci |gawk '{printf
  "%2i %02i %11.3f %i %s \n",$1,$1,$3,$3, $1}'|more
```

(h) Access the manual pages of `gawk`.
Execute:

```
man gawk
```

[7] The command '`|`' allows us to connect the output of a process with the input of another. For example, the output of `cat` can be sent to `more`.

[8] The sentence line is a single line, although it may appear as two lines because of typesetting constraints.

5. Graphics environment `graph.py`

(a) Program `graph.py` is in directory ∼`/GNSS/PROG/src/gLAB_src`. Link this program to the current directory `"."` (i.e. directory `working`).
Execute:

```
ln -s ∼/GNSS/PROG/src/gLAB_src/graph.py .
```

(b) For the previously generated file `prb1`, plot the third field (x coordinate) as a function of the second one (time in seconds).
Execute:[9]

```
graph.py -f prb1 -x2 -y3
```

(c) Using file `sxyz.eci`, plot satellites #5 and #9.
Execute:[10]

```
graph.py -f sxyz.eci -x2 -y3 -c '($1 == 5)'
         -f sxyz.eci -x2 -y3 -c '($1 == 9)'
```

(d) Repeat the previous plot for the interval [20 000 : 30 000] on the x-axis.
Execute:

```
graph.py -f sxyz.eci -x2 -y3 -c '($1 == 5)'
         -f sxyz.eci -x2 -y3 -c '($1 == 9)'
         --xn 20000 --xx 30000
```

(e) Using file `prb1`, plot on the same graph the x (third field), y (fourth field) and z (fifth field) coordinate as a function of time (second field).
Execute:

```
graph.py -f prb1 -x2 -y3 -f prb1 -x2 -y4 -f prb1 -x2 -y5
```

(f) Using file `prb1`, plot on the same graph the distance of the satellites from Earth's mass centre ($r = \sqrt{x^2 + y^2 + z^2}$) as a function of time.
Execute:

```
graph.py -f prb1 -x2 -y'math.sqrt($3*$3+$4*$4+$5*$5)'
```

(g) For the same file used in the previous cases (`prb1`), write the title `"orbit"`, x label `"sec"` and y label `"m"`.
Execute:

```
graph.py -f prb1 -x2 -y3 -t "orbit" --xl "sec" --yl "m"
```

[9] Depending on the `PATH` configuration, it would be necessary to execute '`./graph.py`' instead of '`graph.py`'. Another possibility is to include the current directory '`.`' in the `PATH` variable of the current terminal. This is done in the `bash` environment by executing `export PATH="./:$PATH"`. On the other hand, if we want to make this `PATH` update permanent, then the sentence `export PATH="./:$PATH"` must be included at the end of file '`.bashrc`' in the home directory. When working in `tcsh` instead of `bash`, the equivalent sentences are as follows: execute `set PATH=./:${PATH}` in the current directory (for a non-permanent change), or add the sentence `set path=(./ $path)` to the end of file `/etc/csh.cshrc` to make the `PATH` update permanent. Note that, in this last case, administrator privileges are needed to edit the `/etc/csh.cshrc` file.

[10] Note that, in the Windows OS, instructions like `'($1=="5")'` must be written as `"($1=='5')"`; that is, replacing (") by (').

(h) Visualise the different forms of graphic representations in the following instructions (with points, lines and other symbols or colours).

Execute:

```
graph.py -f prb1 -x2 -y3 -s.
graph.py -f prb1 -x2 -y3 -s-
graph.py -f prb1 -x2 -y3 -s-.
graph.py -f prb1 -x2 -y3 -s--
graph.py -f prb1 -x2 -y3 -so
graph.py -f prb1 -x2 -y3 -s+
graph.py -f prb1 -x2 -y3 -sp
graph.py -f prb1 -x2 -y3 -so --cl r
graph.py -f prb1 -x2 -y3 -s. --cl g
```

(i) Save the plot in a `POSTSCRIPT` file `"orbit.ps"` and in a `Portable Network Graphics (PNG)` file `"orbit.png"`.
Execute:

```
graph.py -f prb1 -x2 -y3 --sv orbit.ps
graph.py -f prb1 -x2 -y3 --sv orbit.png
```

(j) Check `"help"` in `graph.py`.
Execute:

```
graph.py -help
```

Session 2.2. GNSS Standard File Format

By Adrià Rovira García
gAGE/UPC & gAGE-NAV, S.L.

Objectives

To become familiar with the different format standard(s) involved in the GNSS data sets. To provide the user with a reliable and powerful tool to learn the standard formats in a very easy and friendly way, aided by explanatory tool tips. These tips will be triggered automatically when the mouse is hovered over a field. The explanations include a description of the field, the format in which it is written and, if applicable, its units.

Files to use
`LaunchHTML.html`, `GLONASS_Navigation_Rinex_v2.11.html`, `SP3_Version_C.html`, `Observation_Rinex_v3.01.html`, `ANTEX_v1.3.html`, `Observation_Rinex_v2.11.html`, `IONEX_v1.0.html`, `Observation_Rinex_v2.10.html`, `SP3_Version_C.html`, `RINEX_CLOCKS_v3.00.html`, `SBAS_Navigation_Rinex_v3.01.html`, `GPS_Navigation_Rinex_v2.11.html`

Programs to use
Any Web browser (`Firefox` can be used, as well).

Development

1. RINEX measurement files: v2.10

This standard gathers the GNSS observations collected by a receiver. The file is divided clearly into two different sections: the header section and the observables section. While in the header global information is given for the entire file, the observables section contains the code and carrier measurements, among others, stored by epoch.

Open file `Observation_Rinex v2.10.html` with a Web browser and answer the following questions (the answer is provided at the end of each question).

(a) Hover the mouse over the title '`Observation RINEX 2.10 Format`', where some general information is given. Which was the first institution to develop this format?

→ *The Astronomical Institute of the University of Berne.*

(b) What was the reason for developing such a standard?

→ *The exchange of GPS data from several different GPS receivers.*

(c) Which type of optimisation has been applied to this standard?

→ *Minimum space requirements, keeping the observation records as short as possible.*

(d) What is the maximum record (i.e. line) length?

→ *Files are kept to 81-character lines.*

(e) The header section contains a set of labels. Where are they located?

→ *The header labels are located between the* `61st` *and the* `80st` *column of each line.*

(f) Where can a collection of currently used formats be found?

→ *On the IGS website:*
http://igscb.jpl.nasa.gov/components/formats.html.

(g) The header section of this file is dual coloured in order to distinguish header information from header labels. What is the first and the last header label of the header section?

→ *The first label is '*`RINEX VERSION / TYPE`*' while the last label is '*`END OF HEADER`*'.*

(h) What is the difference between the '`RINEX VERSION / TYPE`' and the '`COMMENT `' labels?

→ *While the first one is a mandatory label, the second is optional and may not appear in some RINEX files.*

(i) Hover over the first field in the '`RINEX VERSION / TYPE`' line; what is the main advance in this RINEX version?

→ *Version 2 has been prepared to contain GLObal NAvigation Satellite System (Glonass) or other satellite system observations apart from GPS satellites.*

(j) Where is the antenna located approximately? In which reference system?

→ *The approximate antenna position is (478 902 8.4701, 176 610.0133, 419 501 7.0310) in the World Geodetic System 84 (WGS-84) system.*

(k) Hover over the first field in the '`LEAP SECONDS`' line; what is the difference between the various satellite systems, when counting leap seconds?

→ *The Glonass time system is tied to UTC, so leap seconds are needed, for instance, to mix Galileo or GPS with Glonass data.*

(l) How many satellites are present in the current file? To which satellite system do they belong?

→ *There are 14 different satellites:*
14 GNSS satellites = 10 GPS + 3 Glonass + 1 Satellite-Based Augmentation System (SBAS).

(m) Line '`PRN / # OF OBS` ' has to be consistent with the observables listed in line '`# / TYPES OF OBSERV`'. What are these observations? Do all satellites contain the same observables in this example?

→ *The observables are:* `L1, L2, P1, P2, C1, S1, S2`.
Glonass and SBAS satellites only have: `L1, C1, S1`.

(n) How does the standard indicate the end of the header?

→ *The last field of the header is a record of 60 empty characters.*

(o) The data section of this file is again dual coloured in order to distinguish blocks of data. According two the different colours used, how many epochs are included in this file?

→ *There are two different epochs, each with a different colour.*

(p) The first line of an epoch contains information on a whole block of measurements. When is the first epoch of the file?

→ *The first epoch corresponds to* `5/3/2010` *at* `00:00:00`.

(q) How many satellites are present in the first epoch? Is there any particular feature occurring due to this number?

→ *The first epoch contains* `14` *different satellites. Because there are more than* `12` *satellites, the list of epoch satellites is divided into two different lines.*

(r) Measurements follow the order stated in line '`# / TYPES OF OBSERV`' line. The satellite order is given in the first line of the epoch. What are the units for the observables?

→ `L1, L2`*: cycles of the carrier.*
`P1, P2, C1`*: metres.*
`S1, S2`*: receiver dependent.*

(s) Hover over the first L1 measure of satellite G13. The measure is given together with two indicators. What are these indicators? What do they represent?

→ *Loss of Lock Indicator: Depending on its value, it can show a cycle slip, an opposite wavelength factor, or an anti-spoofing measurement. Signal Strength: Projected into the* `[1-9]` *interval.*

2. RINEX measurement files: v2.11

Open file `Observation_Rinex_v2.11.html` with a Web browser.

(a) Hover over the '`WAVELENGTH FACT L1/2`' line records. The first line states the default values for the wavelength measures for the L1 and L2 frequencies. Which satellite system do the default values apply to? Interpret the second line of the '`WAVELENGTH FACT L1/2`' records.

→ *The Default Wavelength Factor applies for GPS only.*
Three satellites `(G14, G18, G19)` *have a full cycle factor in the first frequency (L1) and half a cycle factor in the second frequency (L2).*

(b) Hover over the first record of line '`# / TYPES OF OBSERV`'. What observation types are defined in RINEX version 2.11? In what units are these observations measured?

→ *Pseudorange measures:* `C` *and* `P` *code* `[m]`
Carrier phase: `L [cycles of the carrier]`
Doppler frequency: `D [Hz]`
Signal-to-noise ratio: `S [receiver-dependent]`.

(c) What value is set in the '`RCV CLOCK OFFS APPL`' record? Where can the Receiver Clock Offset be found later in the RINEX file?

→ *It has a value of* `1`*, which means that an offset is applied in the receiver clock. The value of this offset is reported in the last record of the first line of each epoch.*

3. RINEX Measurement files: v3.01

Open file `Observation_Rinex_v3.01.html` with a Web browser.

(a) This RINEX version is newer than version 2. At first glance the header section is larger and the observation data records are reorganised to be column readable. Which satellite system does this file belong to? What type of file is it?

→ *This file contains GPS, Glonass and SBAS observations, so it is a mixed file. It is an observation data file. Note that the possible values may have changed from the previous version.*

(b) The date record in '`PGM / RUN BY / DATE`' also has been redesigned. What is the date format now? What has changed in the versions?

→ *In version 2.11, there was no a strict rule stating date file creation. In version 3.01, the date has to follow the structure* `Year/Month/Day - Hour/Minute/Second - Time zone`.

(c) A '`MARKER TYPE` ' record has been added to the header. What type of marker type would report a station at the North Pole?

→ *The North Pole is located in the middle of the Arctic Ocean, almost permanently covered with constantly shifting sea ice. It suits the '*`FLOATING_ICE`*' marker type.*

(d) An extensive description of the ANTENNA can now be obtained from this new standard. What type of antenna information is available?

→ *Antenna Number. Antenna Type + Radome Identifier.*
Antenna Reference Point (ARP): (Height, East Eccentricity, North Eccentricity).
ARP: (X,Y,Z) when mounted on a vehicle.
Antenna Phase Centre: (North, East, North) w.r.t. the ARP.
Antenna Bore-sight: (North, East, North) or (X,Y,Z).
Antenna Zero Direction: Azimuth or Vector.

(e) A new '`SYS / # / OBS TYPES`' record has been added in this file. What observations are present for the SBAS satellites?
Hover over the observation descriptors. How many pseudorange code descriptors are present for the sixth Beidou (Compass) frequency?

→ *Glonass and SBAS satellites have the same four observation types:* `L1C S1C C1C S1C`. *RINEX v3.01 defines* `C6I, C6Q, C6X` *pseudorange code descriptors for Beidou.*

(f) A new '`SIGNAL STRENGTH UNIT` ' record has been added to standardise RINEX Signal Strength Indicators (SSIs). What indicator would have a signal-to-noise ratio of 33 dBHz at the output of the correlator?

→ *From the RINEX table, it would present a* `5 SSI` *value.*

(g) Records '`SYS / DCBS APPLIED`' and '`SYS / PCVS APPLIED`' inform if DCB or Phase Centre Variation (PCV) has been applied. What programs/files have been used in these files?

→ *DCBs are corrected using the* `CC2NONCC` *program*[11] *with the input file* `p1c1bias.hist`. *PCVs are corrected using the* `PAGES` *program with input file* `igs05.atx`.

(h) Hover over the '`SYS / SCALE FACTOR`' data records. What is the purpose of implementing such a scale factor?

→ *The purpose of the scale factor is to increase resolution of the phase observations.*

(i) Find the week number in the '`LEAP SECONDS`' line. What is the reason for the week roll-over? When it did happen first?

→ *Because it is a 10-bit number.*
It happened on `22/08/1999` *at* `00:00:00` *GPS time.*

(j) Hover over any observation data record. Why are there lines of 80 characters, and lines that are longer?

→ *While the epoch headers retain a record length of 80 characters, the observation record length limitation of RINEX versions 1 and 2 has been removed.*

4. RINEX navigation files: v2.11 (GPS)

The standard navigation messages broadcast by the GNSS satellites discussed here can vary slightly from one satellite system to another. For example, while the GPS navigation RINEX contains pseudo-Keplerian elements that permit the calculation of the satellite's position, Glonass navigation RINEX contains the satellite's position, velocity and Sun and Moon acceleration in order to integrate the satellite orbits using the Runge–Kutta numerical method.

Open file `GPS_Navigation_Rinex_v2.11.html` with a Web browser.

(a) As usual, this file is divided into a header section and a data record section. The header section is shorter than in the observation RINEX. How short can a GPS navigation message RINEX header be?

→ *The shortest header contains just three lines: '*`RINEX VERSION / TYPE`*', '*`PGM / RUN BY / DATE`*' and '*`END OF HEADER`*'.*

(b) What ionospheric corrections are given?
The `alpha0`, `alpha1`, `alpha2`, `alpha3` *and the* `beta0`, `beta1`, `beta2`, `beta3` *coefficients for the Klobuchar model.*

(c) How many leap seconds are used in the file? When is it recommended to state the number of leap seconds?

→ *Fifteen leap seconds. It is recommended for mixed GPS/Glonass files.*

[11] Program `CC2NONCC` is available from *ftp://dgn6.esoc.esa.int/CC2NONCC*.

(d) This file contains the ephemerides for two different satellites, each block of ephemeris having a separate colour. Where is the satellite identifier found? What satellites are present in the file?

→ *The satellite identifier is found in the first record of the ephemeris block. The Pseudo-Random Noise (PRN) identifiers present in the file are* `PRN 6` *and* `PRN 13`*.*

(e) The first line of each ephemeris block contains three records to compute the satellite clock offset to the GPS time. What are these three coefficients?

→ *They are the polynomial coefficients to compute the satellite clock offset from the GPS system time. These coefficients are: bias, drift and drift rate from the GPS time.*

(f) GPS orbits are nearly circular. In the third line can be found the broadcast orbit eccentricity. What are the orbit eccentricities for the file satellites?

→ `PRN 6` *has an eccentricity of* 0.006 267 404 183 75*.*
`PRN 13` *has an eccentricity of* 0.002 002 393 477 60*.*

(g) The last non-blank field (second record in the eighth row) states the validity period for each ephemeris block. What is the value if it is not known? In this case, what is the fitting interval?

→ *The value is zero when the fitting interval is unknown. In this case, the fitting interval is* $[t_0 - 2\,h, t_0 + 2\,h]$.

5. RINEX navigation files: v2.11 (Glonass)

Open file `GLONASS_Navigation_RINEX_v2.11.html` with a Web browser.

(a) How do you identify this file as a Glonass navigation message?

→ *The RINEX file type value is a '*`G`*'.*

(b) Does this file include any ionospheric information?

→ *The Glonass navigation message does not broadcast ionospheric corrections.*

(c) What correction has to be applied to correct Glonass system time to UTC for the Soviet Union (SU) time zone?

→ `Tutc = Tsv + TauN - GammaN*(Tsv-Tb) + TauC.`

(d) Each of the other lines of Glonass navigation files contain three records with satellite position, velocity and Sun–Moon acceleration. In which units are they given? What information gives the fourth record of each line?

→ *Units are: km, km/s and km/s*^{2}*.*
Message Frame Time, Satellite Health, Frequency Number, Information Age.

(e) Frequency information is found in the third row of each ephemeris block. What are the channel numbers of the two satellites present in the file? Which are the associated frequencies?

→ `R3` *used the* `21st frequency slot`*, so*

$$1602 + 0.5625 * 21 = 1613.8125\, MHz.$$

`R11` *used the* `4th frequency slot`*, so*

$$1602 + 0.5625 * 4 = 1604.2500\, MHz.$$

6. RINEX navigation files: v3.01 (SBAS)

Open file `SBAS_Navigation_RINEX_v3.01.html` with a Web browser.

(a) The SBAS broadcast navigation message is given in a new version (v3.01). The first change is observed in the first header line, where the file type and the satellite system are now clearly divided. What is the difference between the older version 2.11 in representing the navigation messages?

→ *In the older navigation message version, the satellite system record was unused. In version 2.11, file type* `N` *or* `G` *would directly represent a GPS or Glonass navigation message.*
In version 3.01, file type `N` *represents a navigation message, where the satellite system record specifies which of the* `G`*,* `R`*,* `S`*,* `E`*,* `M` *satellite system(s) is involved.*

(b) The '`TIME SYSTEM CORR`' line allows the satellite system time to be transformed to UTC time through a correction. What are the coefficients of the formula? Which is the augmentation system of this navigation message?

→ *The formula is:* `CORR(t) = a0 + a1*DELTAT`
Particularised:
`CORR(t)= 0.1331791282E-06 + 0.107469589E-12*(t-552960)`
The augmentation system is EGNOS.

(c) This file contains ephemerides for a geostationary satellite. What is the difference in time between the two records? Where can it be found?

→ *The epoch time is located in the first line of each data epoch.*
The first epoch corresponds to `18/12/2010` *at* `00:01:04`*.*
The second epoch corresponds to `18/12/2010` *at* `00:05:20`*.*
So, between them, the elapsed time is 4 minutes and 16 seconds.

(d) SBAS navigation files are similar to Glonass ones, in that both contain records of satellite position, velocity and accelerations. What is the flag for a healthy satellite? Which Issue Of Data Navigation (IODN) corresponds to the ephemeris present in the file?

→ *A healthy satellite contains a* `0.0 Health flag`*.*
The first ephemeris block has `IODN: 23`*.*
The second ephemeris block has `IODN: 24`*.*

7. Global Ionospheric Map files: IONEX v1.0

Using a GNSS tracking network it is possible to extract information about the Total Electron Content (TEC) of the ionosphere on a global scale. The IONosphere map EXchange format (IONEX) format is a well-defined standard used to exchange ionospheric maps. It follows the same philosophy as the RINEX format, even when the files are organised into a header and a data section where the maps are allocated.

Open file `IONEX_v1.0.html` with a Web browser.

(a) Hover over the third record of the first line. How many sources can contribute to produce an IONEX map? And how many models?

→ *Several satellite sources can be used:* `Envisat`, `Geostationary`, `Glonass`, `GPS`, `TOPEX/POSEIDON`, `Navy Navigation Satellite (NNS)` *or International Reference Ionosphere (IRI). Two different models are possible: BENt or ERS.*

(b) This file contain two types of information lines: COMMENT lines and DESCRIPTION lines. What is the difference between the two records? Which model is used in this IONEX map? When is this map valid?

→ *Description lines give a brief description of the technique and model used, while comment lines can contain any kind of information. The IONEX map model of this file is based on spherical harmonics. The ionospheric map is for* `Day of Year (DoY) 288 of 1995`.

(c) What information would you expect to appear in the '`OBSERVABLES USED`' line when using a theoretical model?

→ *A blank line is given when a theoretical model is used.*

(d) After COMMENT line(s) the IONEX Map Grid is described. Give details about: the number and dimension of the maps present in the current file with its mapping function; the number of stations and satellites used for the TEC computations; the grid size and the satellite elevation cut-off.

→ *The IONEX file contains five 3D* `TEC/RMS/HGT` *maps with a* $1/\cos(z)$ *mapping function.*
80 *stations and* 24 *satellites have been used to produce this map.*
The grid extends from 200 *to* 800 *km in height with an equidistant increment of* 50 *km.*
The grid extends from 85° *to* −85° *in latitude and from* 0° *to* 355° *in longitude in increments of* 5°.

(e) There are some auxiliary data in this file. What type of information is given, and in which units?

→ *The IONEX map gives the DCBs and their Root Mean Square (RMS) for the satellites used in the map.*
The DCBs are given between the GPS P1 and P2 codes in units of nanoseconds (of L1−L2 delay).

(f) Once the header section ends, the ionospheric maps are detailed. How many ionospheric maps are presented in the current file?

→ *There are three different maps: a TEC; an RMS error map of the associated TEC map; and a final map containing the heights at which the TEC values are obtained.*

(g) Hover over the first TEC values. What is the time for this map?

→ *The first map corresponds to* `15/10/1995` *at* `00:00:00`.

(h) How can the exponent values be interpreted?

→ *The exponent values indicate the* 10 *exponent to apply to the TEC values:*
Exponent: `-3` *TEC field;* `1000` *TEC value:* $1000 \cdot 10^{-3} = 1$ *TECU. Total Electron Content Unit (TECU) is the given unit for TEC, where 1 TECU corresponds to* $10^{16} e^{-}/m^2$.

(i) How would an non-available TEC value be seen?

→ *Non-available TEC values are given as* `99999` *values.*

(j) How many TEC values fit in a single line and in which units are these values given?

→ *TEC rows are given in 16 fields per line, up to the grid limits.*

(k) Hover over any of the RMS values. What is the default exponent for stating the RMS values of the TEC maps?

→ *The default exponent is* `-1`*:* $10^{-1} = 0.1$ *TECU, where 1 TECU corresponds to* $10^{16} e^{-}/m^2$.

8. RINEX clock files: v3.00

The standard files provide station and satellite clock data. Four types of information are given in this format: data analysis results for receiver and satellite clocks derived from a set of network receivers and satellites with respect to a reference clock; broadcast satellite clock monitoring;, discontinuity measurements; and calibration(s) of single GNSS receivers.

Open file `RINEX_CLOCKS_v3.00.html` with a Web browser.

(a) This clock data file starts with a header section as usual. Hover over the first and second comment records. What is the difference between the comments? What kind of information does the second comment give?

→ *The first comment is generic, while the second is a Timescale Re-Alignment comment. This comment is required if '*`Ax`*' data are given (*`AR` *or* `AS`*). In case clock values have been time-scale shifted, the method applied to all receiver and satellite clocks should be noted.*

(b) Hover over the '`SYS / # / OBS TYPES`' records. Which observation descriptors are present in this RINEX clock file?

→ *This file contains four different descriptors from the GPS system:* `C1W`*,* `L1W`*,* `C2W`*,* `L2W`.

(c) Records '`SYS / DCBS APPLIED`' and '`SYS / PCVS APPLIED`' describe the DCB and PCV. What programs are used to apply different corrections?

→ *DCB corrections are applied using the* `CC2NONCC` *program, while PCV corrections are applied using the* `PAGES` *program.*

(d) Records '`STATION NAME / NUM`' and '`STATION CLK REF`' give information about the station. Which receiver identifier is present in the file? What external clock of this station is used for calibration?

→ *The station name is the United States Naval Observatory (USNO) with a receiver identifier* `40451S003`*. The external clock is the USNO, connected via a continuous cable.*

(e) What is the analysis centre of this file?

→ *The analysis centre is the USNO, using the GPS Inferred Positioning SYstem (GIPSY)/OASIS software.*

(f) This file uses two different groups of '`# OF CLK REF`' and '`ANALYSIS CLK REF`'. When does each clock reference apply?

→ *The data set is for the date* `14/07/1994`*. USNO clock reference applies from* `00:00:00` *to* `20:59:59`*. Afterwards, the clock reference used is TIDB from* `21:00:00` *to* `23:59:59`*.*

(g) How many receivers are included in the data file? Which reference frame is used to give station coordinates?

→ Five different stations are used in the file: `GOLD` (USA), `AREQ` (Peru), `TIDB` (Australia), `HARK` (South Africa), `USNO` (USA).

(h) The data record section of this file comes right after the '`END OF HEADER`' record. Each clock data record starts with a data type identifier. Which ones are present?

→ *The following clock data records are present:* `AR` *(Receiver Analysis),* `AS` *(Satellite Analysis),* `CR` *(Receiver Calibration),* `DR` *(Receiver Discontinuities). The only missing record is the* `MS` *(Satellite Monitor) record.*

(i) What data records are present for all clock data types? Which data records are sometimes present?

→ *Records '*`Clock Bias`*' and '*`Clock Bias Sigma`*' are always present. The '*`Clock Rate`*', '*`Clock Rate Sigma`*', '*`Clock Acceleration`*' and '*`Clock Acceleration Sigma`*' are only present for the* `AR` *clock data type.*

9. APC files: ANTEX v1.3

These standard files in ANTenna EXchange format (ANTEX) contain satellite and receiver antenna corrections. Satellite data include satellite and block-specific Phase Centre Offset (PCO). Receiver data include elevation and azimuth-dependent corrections for combinations of antennas and radomes.

Open file `ANTEX_v1.3.html` with a Web browser.

(a) As usual, this file contains a header section. Hover over the record present in line '`PCV TYPE / REFANT`'. Which PCV is used in this file? Which antenna is used as a reference?

→ *PCVs are relative (i.e. not absolute, see section 5.6.1, Volume I). This ANTEX file uses the* `AOAD/M_T` *antenna as a reference.*

(b) The different blocks of information are clearly identified using colours. What are the opening and closing records of each block? Which antennas are described in the file?

→ *The antenna blocks start with record '*`START OF ANTENNA`*' and end with '*`END OF ANTENNA`*'. This ANTEX file contains three different*

antenna descriptions. The first one is from the `GLONASS R08` *satellite, the second from the GPS satellite* `G12` *(modernised block 2) and the third corresponds to the* `AOAD/M_B` *antenna.*

(c) Analyse Glonass antenna record '`TYPE / SERIAL NO`'. How is the satellite code (CNNN) interpreted? When was the satellite launched? How many satellites were launched that year before this one? Repeat the exercise with the GPS satellite.

→ *The Glonass antenna corresponds to a* `GLONASS (R)` *satellite, numbered* `729`*. It was launched during the year 2008. According to the ANTEX record, it is the 67th launched satellite. The GPS antenna is onboard the* `58th GPS Space Vehicle (SV)`*. It is the* `52nd` *satellite launched during 2006.*

(d) Line '`DAZI`' shows the azimuth increment used to characterise the antenna's azimuth phase pattern. For satellite R08, is the PCV azimuth dependent? What about the GPS one?

→ *Because a* `DAZI` *value of* 0.0 *is given, the PCVs of* `R08` *and* `G12` *are not azimuth dependent.*

(e) '`ZEN1 / ZEN2 / DZEN`' gives information on both `AZI` and `NOAZI` phase patterns. What satellite grids are used to study the antennas? Is there any difference between receiver and satellite antennas?

→ *Both satellite grids start at* 0.0° *and end at* 14.0° *in* 1° *steps. Receiver antennas use zenith degrees, satellite ones use nadir degrees.*

(f) '`# OF FREQUENCIES`' determines the number of frequencies for each antenna block. What frequencies are described in the block for each satellite phase pattern?

→ *Two different frequencies are described. The Glonass satellite describes* `G1` *and* `G2` *frequencies, while the GPS satellites describe* `L1` *and* `L2`*.*

(g) The frequency section extends from '`START OF FREQUENCY`' to '`END OF FREQUENCY`' records. What information is given? Give the eccentricity vector for both satellites. What is the origin of this vector?

→ *The eccentricities of the APC and the phase pattern values are* `R08=(-545.00,0.00,2300.00)` *and* `G12=(-10.16,5.87,-93.55)`*. The vector points from the satellite centre of mass to the satellite APC.*

(h) How many non-azimuth-dependent (NOAZI) phase pattern values are specified? Why?

→ *Fourteen different* `NOAZI` *values are specified, due to the '*`ZEN1 / ZEN2 / DZEN`*' definition. Because* `DAZI` *is set to* `0`*, NOAZI-dependent phase patterns are specified.*

(i) The third antenna corresponds to a receiver antenna. What is the calibration method? Which agency has created such corrections?

→ *Calibrations have been converted from file* `igs_01.pcv`*. Technische Universität München (TUM) is the creator of the file.*

(j) This antenna has a non-zero DAZI record for the PCVs. What is the value of this increment? Where else can we see this `DAZI`?

→ *This antenna block has a* 30° *azimuth increment for the PCVs. So, this value will be given from 0 to 360 in increments of* 30°.
Apart from giving the `NOAZI` *values, this data set includes the* `DAZI` *values, both of them using a* 0.0 *to* 90 *grid with a* 5° *step.*

10. Precise orbit and clock files: SP3 version C

Precise orbital data (satellite position and velocity), the associated satellite clock corrections, orbit accuracy exponents, correlation information between satellite coordinates and satellite clock are available in this format.

Open file `SP3_Version_C.html` with a Web browser.

(a) The structure of this file is different from the RINEX ones seen up to now. Hover the mouse over the '`Extended Standard Product 3 Orbit Format`' title, where some general information is given. What additional information is included regarding to the `SP3-c` version?

→ *Satellite clock corrections, orbit accuracy exponents, comment lines, the GPS week and the first epoch seconds of a week.*

(b) What was the main advance in the SP3-b version? Did this change some previous characteristics?

→ *Version* `B` *of the* `SP3` *files accommodated Glonass orbits.* `SP3-b` *modifications were backwards compatible with* `SP3-a`*, with the exception of the satellite identification records, from an* `I3` *field to an* `A1,I2` *one.*

(c) What is the main advance in the `SP3-c` version? How is this achieved?

→ *Files now include not only clock accuracy information, but also information on the accuracy of* (X, Y, Z) *satellite coordinates, which is added using columns 61 to 80 in each position and clock record.*

(d) How is this format organised? Is there any optional record?

→ *The format of an SP3 file is a header, followed by a series of epoch times, each with a set of position and clock records listed for each satellite. Optional records are: satellite velocities, clock correction rate of change, position and clock correlation record* `(EP record)` *and velocity and clock rate-of-change correlation record* `(EV record)`.

(e) The first line of the SP3 file contains information about the entire file. What is the version of the current file? What might be the next SP3 version? Which flag mode is set in this file? What time is the first epoch? How many epochs are included in this file? What type of orbit is used in the file? Which agency has created the file?

→ *The version of the file is* `C-version`. `SP3` *versions follow the alphabet, so next one might be the* `SP3-D` *version. The velocity mode flag is set. This means the records will follow the order* `P,V, P,V, ...` *in case there are additional records, (*`EP` *or* `EV`*), which would be in the middle.*
The first epoch dates from `08/08/2001` *at* `00:00:00`.
There are 192 epochs up to a maximum of 10 million.

The orbit type is HLM, which means it has been fitted by applying a Helmert transformation.
IGS is the agency that created the file.

(f) What is the symbol identifying the start of the second line? Which GPS week number corresponds to this file? Which interval is used in this file?

→ *The* `##` *symbols.*
This file corresponds to the GPS week `1126`*.*
A `900` *s interval is used.*

(g) The next block (lines 3–7) states the satellites used in the file. How many have contributed? What does a zero value mean? How many different satellites from different satellite systems are identified?

→ *There are* `31` *different satellites.*
The `0` *value indicates that all identifiers have been listed.*
Each identifier consists of the satellite system indicator followed by a two-digit integer.

(h) From lines 8 to 12, orbit accuracy of the satellites is listed following the previous block order (lines 3–7). How is this accuracy exponent interpreted?

→ *For example, if the accuracy exponent is* $13 \rightarrow 2^{13}\,\mathrm{mm} \simeq 8\,\mathrm{m}$*. This accuracy represents one standard deviation of the entire file (per satellite).*

(i) The next block comprises lines 13 and 14. Although it is mainly unused, what information is given in the file?

→ *This block states the file type and the system time used in the file.*

(j) The next block comprises lines 15 and 16. What information is given in the file? What is the reason for using such base numbers?

→ *This block contains the floating-point base number used for computing the standard deviations of satellite position, velocity, clock correction and rate of change of the clock correction. Better resolution can be obtained using a floating-point number different from 2.*

(k) After the 'comment' section, the data records start with the previously seen time of first epoch. What symbols mark the start of comments and the new epoch?

→ *Comments start with symbols '*`/*`*', while the new epoch starts with symbol '*`*`*'.*

(l) The second epoch line contains the position and clock data record. What parameters are given? Which units are used? Explain the different flags applied to the first satellite.

→ *Satellite (X,Y,Z) positions, the clock correction with its corresponding standard deviation exponents. There are four final flags.*
Satellite positions are given in kilometres, clock correction in milliseconds. The exponents generate corrections in millimetres and picoseconds.
The first satellite has set: a discontinuity flag between the previous epoch and the current epoch; an orbit position and clock prediction flag; and a manoeuvre flag.

(m) The third epoch line contains the extended position and clock correlation record. What information is given in the first four records? What is the difference with the previous line? Which correlation coefficients are given next?

→ *The standard deviations of the satellite position and clock correction are given with greater resolution than the approximate values given in the position and clock record.*
The correlations between the `X/Y`*,* `X/Z`*,* `Y/Z` *satellite coordinates, and the correlations between the* `X/C`*,* `Y/C`*,* `Z/C` *satellite coordinates and clock.*

(n) The fourth epoch line contains the velocity and clock rate-of-change record. In which units are satellite velocities, clock rate of change and their deviations expressed? What happens if the deviations are unknown or too big to represent?

→ *Velocities are given in decimetres/second and clock rate of change in microseconds/second. Velocity deviation is in millimetres/second and clock rate-of-change deviation in picoseconds/second.*
A value of 99 means the standard deviation was too large to represent, while a blank means it is unknown.

(o) The fourth epoch line contains the extended velocity and clock rate-of-change record. What correlation coefficients are given? Why are the values given greater than one?

→ *The correlations between the* `Vx/Vy`*,* `Vx/Vz`*,* `Vy/Vz` *satellite velocities, and correlations between the* `Vx/Clock`*,* `Vy/Clock`*,* `Vz/Clock` *satellite velocities and clock.*
Because they have to be divided by 10^7.

3. Satellite Orbits and Clocks

This chapter focuses on the analysis of GNSS satellite orbits and clocks.

The first session contains introductory exercises on coordinate frames and satellite orbits. Using broadcast and precise orbit files of GPS, Glonass and Galileo satellites, the different orbits are compared and the tracks on an Earth-Centred, Earth-Fixed (ECEF) coordinate frame are shown. The osculating orbital elements are computed from the satellite coordinates and velocities, and its variation due to the different accelerations acting on the satellites (J_2, Sun+Moon) is depicted. Results obtained by integrating the satellite orbit from initial conditions and neglecting different terms in the force model are compared to assess their effect on the computed satellite coordinates. Finally, the magnitude of the different acceleration terms is assessed and plotted as a function of time.

Session 2 is devoted to the GPS broadcast and precise orbits and clocks. Using a specific function of `gLAB`, the accuracy of broadcast orbits and clocks (from RINEX files) is assessed against the precise IGS products (SP3 and CLK files), which are taken as the truth. This assessment is done in both conditions under S/A activated and deactivated, including the exercises on a broadcast navigation file of 2 May 2000, when the S/A was switched off. Using the capability of `gLAB` to compare two SP3 files, the discrepancies between different IGS products (final, rapid and ultra-rapid) are assessed, as well as the agreement between the IGS combined product and the products from the different associated centres. The target of these exercises is not only to assess the quality of such orbit and clock products, but also to provide a procedure to perform this kind of analysis. In this study it should be noted that broadcast orbits and clocks and IGS products are referred to different APCs, as illustrated in detail within the exercises.

Another issue analysed in this session is the interpolation of high-rate IGS precise clocks. The interpolation error is assessed using recent IGS precise products (S/A off) at different sampling rates. A data file under S/A on is also analysed to illustrate the situation when S/A was activated.

The last session is devoted to Glonass satellite orbit integration. Using an orbit integrator implementing the Glonass Interface Control Document (ICD), the short- and long-term accuracy as well as the step width effect are assessed. Using the `gLAB` functionality to compare orbits and clock files (SP3, CLK), the accuracy of different precise products from the Russian Mission Control Centre (MCC) and IGS, among other centres, is compared.

The effect of Earth's oblateness and Sun+Moon accelerations over the Glonass orbit integration is again analysed in this session.

The exercises include coordinate transformations from Glonass frames PZ-90 to PZ-90.02. In particular the broadcast coordinates on 20 September 2007, when the reference frame was switched to PZ-90.2, are analysed.

Session 3.1. Coordinate Frames and Satellite Orbits

Objectives

To become familiar with satellite (GPS, Glonass, Galileo) coordinates, reference frames and orbital elements. To depict the variations of orbital osculating elements due to different perturbations.

Files to use

`iac15454.sp3`, `brdc1360.00n`, `igs05_1545`, `jpl10621.sp3`, `2000-05-15.eci`, `PRN_GPS`, `gal15591.sp3`, `brdc2320.09g`, `brdc2320.09n`

Programs to use

`gLAB_linux`, `graph.py`, `GPSbrd2rvorb.f`, `car2esf.f`, `rv2osc.f`, `trs2crs.f`, `GLOeph2sp3.f`, `GLOeph2sp3.nml`, `trs2crs.nml`

Development

Session files.

Copy and uncompress these session files in the working directory:

```
cp ~/GNSS/PROG/SES31/* .
cp ~/GNSS/FILES/SES31/* .
gzip -d *.gz *.Z
```

1. Satellite coordinates

The `gLAB` tool has a specific function to compute satellite coordinates and clocks using broadcast RINEX navigation files or SP3 files of precise orbits and clocks. This function is not available in the `gLAB` GUI and must be executed in console mode.

As an example, the following sentence computes satellite coordinates and clock offsets from `brdc2320.09n` and `jpl10621.sp3` files:[1]

```
gLAB_linux -input:nav brdc2320.09n |grep SATPVT> xyzt.brd

gLAB_linux -input:SP3 jpl10621.sp3
          -input:ant igs05_1545.atx|grep SATPVT > xyzt.sp3
```

[1] Depending on the `PATH` configuration, it would be necessary to execute '`./gLAB_linux`' instead of '`gLAB_linux`'. Another possibility is to include the current directory '`.`' in the `PATH` variable of the current terminal. This is done in the `bash` environment by executing `export PATH="./:$PATH"`. On the other hand, if we want to make this `PATH` update permanent, then the sentence `export PATH="./:$PATH"` must be included at the end of file "`.bashrc`" in the home directory.
Note: See the instructions for the `tcsh` *environment in footnote 9 of session 2.1.*

It must be pointed out that, by default, the satellite coordinates output by `gLAB` are referred to the satellite APC. Thus, an ANTEX file[2] is required to transform the coordinates of the `jpl10621.sp3` file, which are relative to the mass centre of the satellite, to the APC.[3]

The previous sentence generates a file with the satellite coordinates and clock offsets, whose contents are given in the following table:[4]

Table 3.1: SATPVT data block description.

Field	Content
1	'SATPVT'
2	Year
3	Doy (Day-of-Year)
4	Seconds of day (seconds)
5	GNSS System (GPS, GAL, GLO or GEO)
6	PRN satellite identifier
7	X (TRF) (m)
8	Y (TRF) (m)
9	Z (TRF) (m)
10	VX (TRF) (m/s)
11	VY (TRF) (m/s)
12	VZ (TRF) (m/s)
13	dt (m)

2. Plotting satellite geocentric distance

Using, for instance, the coordinates computed from the broadcast file `brdc0820.99n`, calculate the geocentric distance (in km) of the satellite PRN15 as a function of time.

Execute for instance:[5]

```
cat xyzt.brd| awk '{if ($6==15) print $4,sqrt($7^2+$8^2
        +$9^2)/1000}' > dist.b

graph.py -f dist.b -so -t "Distance"
```

Note: As indicated above, depending on the `PATH` *configuration, it would be necessary to execute '*`./graph.py`*' instead of '*`graph.py`*'.*

(a) What is the range variation observed? Calculate the relative variation $(r_{max} - r_{min})/r_{min} \times 100$.

[2]The file `igs05_1545.atx` (or later) can be used, as indicated in the `jpl10621.sp3` file header.

[3]In the case of broadcast orbits, they are always referred to the APC (see section 5.6.3, Volume I). In the case of the SP3 files, the coordinates can also be computed relative to the satellite's centre of mass using the instruction `--model:satphasecenter` (notice the double '-') to disable the APC correction (applied by default).

[4]More details can be found by executing: `gLAB_linux -messages`.

[5]As indicated in the previous footnote, depending on the `PATH` configuration, it would be necessary to execute '`./graph.py`' instead of '`graph.py`'.

(b) Calculate the orbital period from the semi-major axis value (see equations (3.12) in Volume I). The satellite TOPEX/Poseidon orbits at a height of 1400 km above Earth's surface. What is its orbital period? (Earth's radius $\simeq 6400\,km$)

3. Satellite tracks: Terrestrial Reference Frame (TRF)

The program `car2esf.f` transforms the Cartesian satellite coordinates (x, y, z) to spherical coordinates (r, λ, ϕ), according to the input/output scheme (see the `car2esf.f` file header):

```
[PRN,YEAR,DoY,sec,x,y,z] ---> |car2esf| --->
                         ---> [PRN,YEAR,DoY,r,λ,φ]
```

By default, coordinates are in metres and angles in degrees.

(a) Using this program, transform the coordinates computed in the previous exercise to (r, λ, ϕ) and plot the tracks over Earth's surface for all GPS satellites. Repeat the plot for a single satellite (for instance, PRN11). Execute for instance:

```
cat xyzt.brd | gawk '{print $6,$2,$3,$4,$7,$8,$9}'
                                  | car2esf > gps.rfl

graph.py -f gps.rfl -x6 -y7 -s.  -l"All GPS"

      -f gps.rfl -x6 -y7 -so -l"PRN11" -c'($1==11)'
```

(b) Repeat the previous exercise with the GPS, Glonass and Galileo SP3 files `jpl10621.sp3`, `iac15454.sp3` and `gal15591.sp3`

Complete the following steps:
1. Computations (uncheck the APC correction to avoid using the ANTEX file):[6]

Satellite coordinates computation from the SP3 files:

```
gLAB_linux -input:SP3 jpl10621.sp3
     --model:satphasecenter |grep SATPVT > gps.xyzt

gLAB_linux -input:SP3 iac15454.sp3
      --model:satphasecenter|grep SATPVT > glo.xyzt

gLAB_linux -input:SP3 gal15591.sp3
      --model:satphasecenter|grep SATPVT > gal.xyzt
```

[6]Note: Turn off the APC corrections by setting the option '`--model:satphasecenter`' (with double '`-`'). Under this flag, the ANTEX file is not required.

Transforming coordinates:

```
cat gps.xyzt | gawk '{print $6,$2,$3,$4,$7,$8,$9}'
                                  | car2esf > gps.rfl

cat glo.xyzt | gawk '{print $6,$2,$3,$4,$7,$8,$9}'
                                  | car2esf > glo.rfl

cat gal.xyzt | gawk '{print $6,$2,$3,$4,$7,$8,$9}'
                                  | car2esf > gal.rfl
```

2. Plotting results:

```
graph.py -f gps.rfl -x6 -y7 -s.  -l "GPS"
          -f glo.rfl -x6 -y7 -s.  -l "Glonass"
             -f gal.rfl -x6 -y7 -so  -l "Galileo"
```

4. Satellite coordinates: Celestial Reference Frame (CRF)

The program `trs2crs.f` performs a rotation around the Z-axis with a magnitude of sidereal time '$-\theta$' (i.e. $R_3[-\theta]$) on the TRF coordinates (x, y, z) to 'roughly' align with the CRF system:[7]

```
[PRN,YEAR,DoY,sec,x,y,z,vx,vy,vz] ---> |trs2crs| --->
---> [PRN,YEAR,DoY,sec,x',y',z',vx',vy',vz']
```

By default, coordinates are in metres and velocities in metres per second.[8]

(a) Using this program, transform the TRF coordinates computed in the previous exercise to the CRF, and plot the orbits in space; that is, using (r, λ, ϕ) coordinates.

Complete the following steps:

1. Computations:

Transform TRF coordinates to CRF and, afterwards,
to spherical (r, λ, ϕ) coordinates:

```
cat gps.xyzt | gawk '{print $6,$2,$3,$4,$7,$8,$9,
           $10,$11,$12}'|trs2crs|car2esf > gps.rfl

cat glo.xyzt | gawk '{print $6,$2,$3,$4,$7,$8,
         $9,$10,$11,$12}'|trs2crs|car2esf > glo.rfl

cat gal.xyzt | gawk '{print $6,$2,$3,$4,$7,$8,
         $9,$10,$11,$12}'|trs2crs|car2esf > gal.rfl
```

[7] It is enough, for instance, to visualise the osculating elements variations, as shown in the next exercise.

[8] The input/output coordinates and velocity units can be given in km and km/s by setting [`input_r="km"`], [`input_v="km/s"`], [`output_r="km"`] and [`output_v="km/s"`] in the namelist `trs2crs.nml`. More information can be found in the header of the `car2esf.f` file.

2. Plot the results:

```
graph.py -f gps.rfl -x6 -y7 -s.  -l "GPS"
           -f glo.rfl -x6 -y7 -s.  -l "Glonass"
              -f gal.rfl -x6 -y7 -so  -l "Galileo"
```

5. Osculating elements variation

The program `rv2osc.f` computes the *osculating* orbital elements (see Appendix C, Volume I) of a satellite, starting from its position and velocity in an inertial reference frame, (CRF)[9] that is 'non-tied' to Earth's rotation.

`[PRN,YEAR,DoY,sec,x,y,z,vx,vy,vz] ---> |rv2osc| --->`
`---> [PRN,YEAR,DoY,sec,`a`,`e`,`i`,`Ω`,`ω`,`M`]`

By default, input coordinates are in km and the velocities in km/s. The output is in km and degrees. More information is in the `rv2osc.f` header.

File `2000-05-15.eci` contains GPS satellite mass centre positions and velocities in the 'pseudo-inertial' CRF (see section 3.1.2, Volume I).

The data contained in this file are organised in the following fields:

`[SV,YYY,MM,DD,hh,mm,ss.ss,x,y,z,vx,vy,vz,flag]`

where time is in UTC, coordinates are in km and velocities in km/s.

It must be pointed out that the program `rv2osc.f` requires the Day of Year (DoY) and the seconds of day as input, instead of `MM,DD,hh,mm,ss`. The update between the file `2000-05-15.eci` fields and the required input of `rv2osc.f` can be done easily using a `gawk` command as illustrated next.

(a) Using program `rv2osc`, compute the orbital osculating elements for satellites[10] and epochs recorded in the file `2000-05-15.eci`. Select the satellite SV32 (that corresponds to PRN01) and plot its osculating elements as a function of time.
Complete the following steps:

1. Computations:

Update file `2000-05-15.eci` fields to required input of `rv2osc`:

i. Selecting day `15` of 2000-05-15 (DoY=136), write the DoY and compute the seconds of day:

```
cat 2000-05-15.eci | gawk '{if($4==15)print $1,$2,"136",
      $5*3600+$6*60+$7,$8,$9,$10,$11,$12,$13}' > tmp.utc
```

ii. Add the leap seconds to transform the UTC to GPS time: (on 2000-05-15 there were 13)

```
cat tmp.utc | gawk '{$4=$4+13; print $0}' > tmp.gps
```

[9] Actually, 'pseudo-inertial', because it is affected by the acceleration due to Earth's motion around the Sun (annual revolution).

[10] The satellite identifier corresponds to the space vehicle number (SV), not the PRN. The relation between both indicators is given in file `PRN_GPS`. This file can be downloaded from *ftp://sideshow.jpl.nasa.gov/pub/gipsy_products/IERSB/PRN_GPS.gz*.

iii. Compute osculating elements:

```
cat tmp.gps | rv2osc > eci.orb
```

iv. Select satellite SV32 (i.e. PRN01):

```
cat eci.orb|gawk '{if ($1==32)print $0}' >eci.orb01
```

2. Plot the results:

```
graph.py -f eci.orb01 -x4 -y5 -l "Semi-major axis"
graph.py -f eci.orb01 -x4 -y6 -l "Eccentricity"
graph.py -f eci.orb01 -x4 -y7 -l "Inclination"
graph.py -f eci.orb01 -x4 -y8 -l "Ascending Node"
graph.py -f eci.orb01 -x4 -y9 -l "Arg.  Perigee"
graph.py -f eci.orb01 -x4 -y10 -l "Mean anom."
```

(b) File `jpl10621.sp3` contains the same satellite coordinates as file `2000-05-15.eci` but in the TRF (tied to Earth). Using the program `trs2crs` rotate the coordinates by the sidereal time '$-\theta$' to 'roughly' align with the CRF. Afterwards, use the program `rv2osc` to compute the orbital osculating elements. Compare the results for satellite PRN01 with those computed from the `2000-05-15.eci` file.

Note: Turn off the APC corrections by setting the option '`--model:satphasecenter`' (with double '-'). Under this flag, the ANTEX file is not required.

Complete the following steps:

1. Computations:

i. Compute satellite coordinates from the SP3 file
(note that the APC is turned off by setting
'`--model:satphasecenter`'):

```
gLAB_linux -input:SP3 jpl10621.sp3
     --model:satphasecenter |grep SATPVT > sp3.trs
```

ii. Transform coordinates from TRF to (roughly) CRF.
Set `input_r="m"`, `input_v="m/s"`, `output_r="km"`
and `output_v="km/s"` in the `trs2crs.nml` file:

```
cat sp3.trs| gawk '{print $6,$2,$3,$4,
        $7,$8,$9,$10,$11,$12}' | trs2crs > sp3.crs
```

iii. Compute osculating elements from file `sp3.crs`:

```
cat sp3.crs|rv2osc >sp3.orb
```

iv. Select satellite PRN01:

```
cat sp3.orb|gawk '{if ($1==1) print $0}' >sp3.orb01
```

2. Plot the results:

Plotting: Compare the results from ECI and SP3 files:

```
graph.py -f sp3.orb01 -x4 -y5 -so -l "SP3"
  -f eci.orb01 -x4 -y5 -so -l "ECI" -t "Semi-major"

graph.py -f sp3.orb01 -x4 -y6 -so -l "SP3"
  -f eci.orb01 -x4 -y6 -so -l "ECI" -t "Eccent."
```

Produce similar plots for the Inclination, Argument of Ascending Node, Argument of Perigee and Mean Anomaly.

(c) Are the variations of the osculating elements computed from both approaches (from true CRF and from raw transformed TRF to CRF coordinates) to the same order? Discuss such variations.

6. GPS broadcast orbital elements

Program `GPSbrd2rvorb.f` computes the GPS satellite coordinates (x, y, z) from the broadcast navigation message of a RINEX file and writes the pseudo-Keplerian elements used to compute such coordinates[11] and the satellite clock offset polynomial coefficients. Results are given at 300 s sampling rate.

`[GPS RINEX navigation file] ---> |GPSbrd2rvorb| --->`
`---> [PRN,YEAR,DoY,sec,`a`,`e`,`i`,`Ω`,`ω`,`M`,`x`,`y`,`z`,`a_0`,`a_1`,`a_2`]`

Output units are km, km/s or degrees.

See more details in the `GPSbrd2rvorb.f` file header.

(a) Using the program `GPSbrd2rvorb.f` and the RINEX navigation file `brdc1360.00n`, compute the GPS pseudo-Keplerian elements from the broadcast navigation message and compare them with the osculating elements obtained in the previous exercise for satellite PRN01 (use, for instance, the `sp3.orb01` file).

Complete the following steps:

1. Computations:

Compute pseudo-Keplerian elements from file `brdc1360.00n`:

```
GPSbrd2rvorb brdc1360.00n > brd.orb
```

Select satellite PRN01:

```
cat brd.orb|gawk '{if ($1==1) print $0}' >brd.orb01
```

[11] The FORTRAN subroutine `orb.f` is used to compute the satellite coordinates.

2. Plot the results:

```
graph.py -f brd.orb01 -x4 -y5 -s.  -l "BRD" --cl r
 -f sp3.orb01 -x4 -y5 -s.  -l "SP3" -t "Semi-major"

graph.py -f brd.orb01 -x4 -y6 -s.  -l "BRD" --cl r
 -f sp3.orb01 -x4 -y6 -s.  -l "SP3" -t "Eccent."
```

Produce similar plots for the Inclination, Argument of Ascending Node, Argument of Perigee and Mean Anomaly.

Discuss the observed discrepancies.

7. Comparison of GPS, Glonass and Galileo orbits

(a) Applying the same procedure as in the previous exercise, compute the osculating elements for the GPS, Glonass and Galileo orbits. Use the files `jpl10621.sp3` (GPS), `iac15454.sp3` (Glonass) and `gal15591.sp3` (Galileo).

Complete the following steps:

1. Computations:

Compute satellite coordinates from the SP3 files (note that the APC is turned off by setting '`--model:satphasecenter`', with double '`-`'):

```
gLAB_linux -input:SP3 jpl10621.sp3
       --model:satphasecenter |grep SATPVT> gps.trs

gLAB_linux -input:SP3 iac15454.sp3
       --model:satphasecenter |grep SATPVT> glo.trs

gLAB_linux -input:SP3 gal15591.sp3
       --model:satphasecenter |grep SATPVT> gal.trs
```

Transform coordinates from TRF to (roughly) CRF and compute the osculating elements.
Set `input_r="m"`, `input_v="m/s"`, `output_r="km"` and `output_v="km/s"` in the `trs2crs.nml` file:

```
cat gps.trs | gawk '{print $6,$2,$3,$4,
   $7,$8,$9,$10,$11,$12}' |trs2crs |rv2osc> gps.orb

cat glo.trs | gawk '{print $6,$2,$3,$4,
   $7,$8,$9,$10,$11,$12}' |trs2crs |rv2osc> glo.orb

cat gal.trs| gawk '{print $6,$2,$3,$4,
   $7,$8,$9,$10,$11,$12}' |trs2crs |rv2osc> gal.orb
```

2. Plot the results:

```
graph.py -f gps.orb -x4 -y5 -s. -l "GPS"
 -f glo.orb -x4 -y5 -s. -l "Glonass"
 -f gal.orb -x4 -y5 -s. -l "Galileo" -t "Semi-major"

graph.py -f gps.orb -x4 -y8 -s. -l "GPS"
 -f glo.orb -x4 -y8 -s. -l "Glonass"
 -f gal.orb -x4 -y8 -s. -l "Galileo" -t "Asc.  Node"
```

Produce similar plots for the Eccentricity, Inclination, Argument of Ascending Node, Argument of Perigee and Mean Anomaly.

(b) Compare results and discuss the main differences and similarities.

8. Satellite orbit integration

The program `GLOeph2sp3.f` integrates the satellite orbit by applying the algorithm defined in section 3.3.2.1, Volume I. This algorithm involves a simple potential model (with only the central body and J2 – Earth's oblateness – terms) and the Sun and Moon accelerations as external perturbing forces. This program was initially designed to integrate the Glonass broadcast ephemeris according to its [GLONASS ICD, 1998], but it also allows the analysis of some effects on the orbits due to Earth's potential and external perturbation terms. The impact of these effects on the satellite orbit will be analysed in this exercise.

As input data, the program uses the Glonass broadcast navigation message, which must be provided in the file named `GLO_EPH.dat`, in RINEX format. This data file includes, as explained in section 3.3.2, Volume I, the Sun+Moon accelerations among the satellite's initial conditions ($\mathbf{r}_{t_0}$,$\mathbf{v}_{t_0}$), in a TRF system,[12] typically every 30 minutes. On the other hand, the program is able to compute such acceleration terms by itself, allowing its use not only for Glonass, but also for GPS or Galileo satellites.[13]

Execution of the `GLOeph2sp3.f` orbit integrator, as well as its input file and configuration parameters (namelist `GLOeph2sp3.nml`), are illustrated within this exercise. More details can be found in the source code.

(a) Example of program running:

i. Copy the broadcast Glonass navigation file `brdc2320.09g` to the input file `GLO_EPH.dat`.
Only the broadcast values for the reference epoch t_0 of 20 August 2009 at 0 hours[14], 15 minutes and 0 seconds will be used in this exercise. Thus, all records after the last block of data corresponding to the reference time `[09 8 20 0 15 0.0]` are removed. After that, the input file `GLO_EPH.dat` remains as follows (where dotted lines indicate intermediate blocks in between):

[12]The Glonass broadcast message is given in the Parametry Zemli 1990 (Parameters of Earth 1990) (PZ-90) system up to 20 September 2007 and afterwards in the PZ-90.02 system.

[13]Only requiring the initial conditions, ($\mathbf{r}_{t_0}$,$\mathbf{v}_{t_0}$), provided in file `GLO_EPH.dat`.

[14]Note: This is $\text{GLONASS}_{\text{time}} - 3\,\text{h}$, i.e. UTC(SU), as it is used in the RINEX files (see section 3.3.2, Volume I).

```
    2.01           GLONASS NAV DATA                         RINEX VERSION / TYPE
CCRINEXG V1.4 UX    CDDIS               21-AUG- 9 16:53     PGM / RUN BY / DATE
IGS BROADCAST EPHEMERIS FILE                                 COMMENT
CCRINEXG V1.4 UX    CDDIS               21-AUG- 9 03:53     COMMENT
  2009     8    20    0.187195837498E-06                    CORR TO SYSTEM TIME
    15                                                       LEAP SECONDS
                                                             END OF HEADER
 2 09  8 20  0 15  0.0-0.197580084205E-04-0.272848410532E-11 0.300000000000E+02
    0.179436245117E+05-0.138885307312E+01 0.372529029846E-08 0.100000000000E+01
   -0.106202993164E+05 0.146085643768E+01 0.000000000000E+00 0.100000000000E+01
    0.147005888672E+05 0.275223255158E+01-0.279396772385E-08 0.100000000000E+01
                  ...      ...      ...
                  ...      ...      ...
                  ...      ...      ...
24 09  8 20  0 15  0.0-0.201585702598E-03 0.272848410532E-11 0.300000000000E+02
    0.239867841797E+05 0.121475696564E+01 0.465661287308E-08 0.000000000000E+00
   -0.168961425781E+03 0.249032974243E-01-0.931322574616E-09 0.200000000000E+01
    0.881358935547E+04-0.330398941040E+01-0.279396772385E-08 0.000000000000E+00
```

Note that the previous file can be easily generated by executing:

```
grep -B10 "END OF HEADER" brdc2320.09g >  GLO_EPH.dat
grep -A3 "09  8 20  0 15" brdc2320.09g >> GLO_EPH.dat
```

Indeed, the previous sentence selects the `brdc2320.09g` header and the broadcast ephemeris of all satellites at the time `09 8 20 0 15 0.0` and merges the content into the file `GLO_EPH.dat`.

ii. Set the `GLOeph2sp3.nml` namelist as:

GLOeph2sp3.nml

```
$parameters
  tsys="GLO"
  wmode="TXT
  wptv0="NO"
  idt_out=1
  dt=300.d0
  niter=276
  useC20="YES"
  useSM="YES"
  use_brdSM="NO"
$end
```

iii. Execute:

```
GLOeph2sp3 > glo15454.txt
```

Using the previous namelist, the program `GLOeph2sp3.f` will integrate the orbit from $[t_0 : t_0 + 276 \times 300$ seconds$]$ for each satellite having initial conditions $(\mathbf{r}_{t_0}, \mathbf{v}_{t_0})$ in file `GLO_EPH.dat`. That is, it will perform 276 iterations (`niter=276`) with a time step of 300 seconds (`dt=300.d0`).

The J2 term (i.e. the C20) of Earth's potential and the Sun+Moon acceleration can be set/unset by the namelist parameters `useC20= "YES/NO"` and `useSM="YES/NO"` (`"YES"` is the default option). In this example, as the parameter `useSMbrdc` is set to '`NO`', then the Sun+Moon accelerations will be computed by the program itself, neglecting the broadcast values of file `GLO_EPH.dat`.[15]

When the output mode `wmode="TXT"` is set, the output is column formatted,[16] containing the following fields (see file `glo15454.txt`):

```
[SV,YYY,MM,DD,hh,mm,ss.ss,x,y,z,vx,vy,vz,clock]
```

where time is given in Glonass system time (actually UTC[SU]).[17] Coordinates are in m and velocities in m/s, in an ECEF frame;[18] that is, tied to Earth's rotation.

The results are set to be output with a sampling rate of 1 second. (`idt_out=1`).[19] Nevertheless, as the integration step is `dt=300`, the solutions will be written every 300 seconds.

(b) Using the `GLOeph2sp3` program, integrate the orbit along a 23 h time interval ($276 \times 300 = 82\,800\,\mathrm{s} = 23\,\mathrm{h}$), from the initial conditions and satellites given in file `GLO_EPH.dat`. Name the output file as `glo15454.txt`. Afterwards, applying the same procedure as in previous exercises, compute the osculating elements from the positions and velocities in file `glo15454.txt`. Select satellite PRN02 and plot its osculating orbital elements.

Complete the following steps:

1. Computations:

i. Orbit integration:

```
GLOeph2sp3 > glo15454.txt
```

ii. Transform coordinates from TRF to (roughly) CRF.
Set `input_r="m"`, `input_v="m/s"`, `output_r="km"` and `output_v="km/s"` in the `trs2crs.nml` file:

```
cat glo15454.txt |gawk '{print $1,$2,"232",$5*3600
+$6*60+$7,$8,$9,$10,$11,$12,$13}'|trs2crs > glo.crs
```

iii. Compute the osculating elements from the `glo.crs` file and select satellite PRN02:

```
cat glo.crs | gawk '{if ($1==2) print $0}'
                                  | rv2osc > glo.orb
```

[15] The broadcast values are only valid for about an hour from the reference time, and here the computation will be extended for a 23 h period.

[16] It can also be written in the `SP3` format as well.

[17] The system time (GPS or Glonass) is defined by the parameter `tsys="GLO"` in the `GLOeph2sp3.nml` namelist.

[18] This frame is PZ-90 or PZ-90.02 if the Glonass broadcast navigation message is used.

[19] Actually, the output will be written using 'modulo 1 second', that is '$t\%(1\,\mathrm{second})$'.

2. Plot the results:

```
graph.py -f glo.orb -x4 -y5 -so -t "Semi-major"

graph.py -f glo.orb -x4 -y6 -so -t "Eccent."
```

Produce similar plots for the Inclination, Argument of Ascending Node, Argument of Perigee and Mean Anomaly.

(c) Repeat previous computations but set (1) `useSM= "NO"` and (2) `useC20="NO"` and `useSM="NO"` in the `GLOeph2sp3.nml` namelist, in order to assess the effect on J2 of Earth's potential and the Sun+Moon accelerations on computation of the satellite orbit.
Complete the following steps:

1. Computations:

i. Orbit integration:
A. Set `useC20="YES"` and `useSM="NO"` in `GLOeph2sp3.nml` and execute:

```
GLOeph2sp3 > glo15454.txt.YN
```

B. Set `useC20="NO"` and `useSM="NO"` in `GLOeph2sp3.nml` and execute:

```
GLOeph2sp3 > glo15454.txt.NN
```

ii. Transform coordinates from TRF to (roughly) CRF.
Set `input_r="m"`, `input_v="m/s"`, `output_r="km"` and `output_v="km/s"` in the `trs2crs.nml` file.

A. Execute:

```
cat glo15454.txt.YN|gawk '{print $1,$2,"232",$5*3600
 +$6*60+$7,$8,$9,$10,$11,$12,$13}'|trs2crs>glo.crs.YN
```

B. Execute:

```
cat glo15454.txt.NN |gawk '{print $1,$2,"232",$5*3600
+$6*60+$7,$8,$9,$10,$11,$12,$13}'|trs2crs >glo.crs.NN
```

iii. Compute the osculating elements from the `glo.crs` file and select satellite PRN02:

Execute:

```
cat glo.crs.YN | gawk '{if ($1==2) print $0}'
                                  | rv2osc > glo.orb.YN
cat glo.crs.NN | gawk '{if ($1==2) print $0}'
                                  | rv2osc > glo.orb.NN
```

2. Plot the results:

Produce the plots:

```
graph.py -f glo.orb -x4 -y5 -s. -l"All" -t"Semimajor"
  -f glo.orb.YN -x4 -y5 -s. -l "No SM"
  -f glo.orb.NN -x4 -y5 -s. -l "No J2 No SM"

graph.py -f glo.orb -x4 -y6 -s. -l"All" -t"Eccent."
  -f glo.orb.YN -x4 -y6 -s. -l "No SM"
  -f glo.orb.NN -x4 -y6 -s. -l "No J2 No SM"
```

Produce similar plots for the Inclination, Argument of Ascending Node, Argument of Perigee and Mean Anomaly.

9. Evaluation of acceleration terms

Using the orbit integrator `GLOeph2sp3` generate a file with the different acceleration terms due to the central body, the J2 term and the Sun+Moon attraction.

Set the `GLOeph2sp3.nml` namelist as:

```
$parameters
 tsys="GLO"
 wmode="OUT"
 wptv0="NO"
 dt=300.d0
 niter=12
$end
```

Note that by setting `wmode="OUT"`, the program outputs several internal computations, including the acceleration terms: a_{Total}, $a_{Central}$, a_{J2}, $a_{Sun+Moon}$. The output file format is:

[SV,YYY,MM,DD,hh,mm,ss.ss,x,y,z,vx,vy,vz,clk_τ,clk_γ,a_{Total}, $a_{Central}$,a_{J2},$a_{Sun+Moon}$]

where the position, velocity and acceleration are given in m, m/s and m/s^2, respectively.

Complete the following steps:

1. Computations:

i. Use the full broadcast file to compute 24 h results:

```
cat brdc2320.09g > GLO_EPH.dat
```

ii. Execute the program (exclude rows with the INI label):

```
GLOeph2sp3 |grep -v INI > glo15454.out
```

iii. Select the acceleration terms:

```
cat glo15454.out|gawk '{print $1,$5*3600+$6*60+$7,
                           $16,$17,$18,$19}' > glo.acc
```

iv. Select satellite PRN22:

```
grep ^22 glo.acc > glo.tmp
```

2. Plot the results:

Compare the central body and total acceleration:

```
graph.py -f glo.acc -x2 -y3 -so -l "TOTAL"
       -f glo.acc -x2 -y4 -s+ -l "Central body"
       -f glo.tmp -x2 -y3 -so -l "TOTAL PRN22"
       -f glo.tmp -x2 -y4 -so -l "Central body PRN22"
```

J2 acceleration term plot:

```
graph.py -f glo.acc -x2 -y5 -s+ -l "All satellites"
             -f glo.tmp -x2 -y5 -so --cl r -l "PRN22"
             --xl "Time (seconds)" --yl "m/s2"
             -t "Acceleration (J2 term)"
```

Sun+Moon acceleration plot:

```
graph.py -f glo.acc -x2 -y6 -s+ -l "All satellites"
             -f glo.tmp -x2 -y6 -so --cl r -l "PRN22"
             --xl "Time (seconds) " --yl "m/s2"
             -t "Sun + Moon Acceleration"
```

(a) Compare the order of magnitude of the different acceleration terms. Contrast the values with those of Table 3.7 in Volume I.

(b) Discuss the different patterns of acceleration terms.

10. Miscellaneous

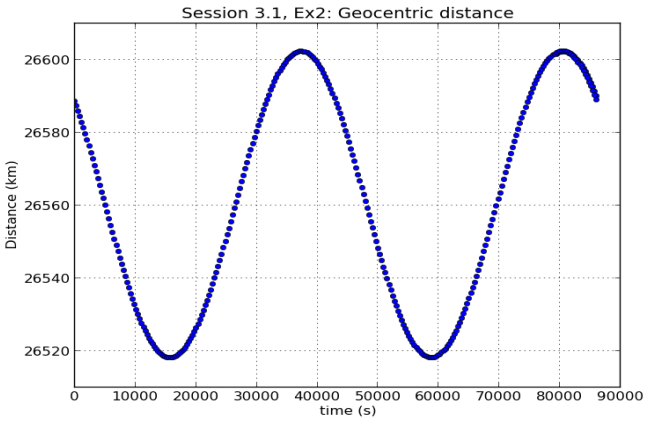
(a) How long does it take approximately for a signal to travel from the satellite to the receiver?

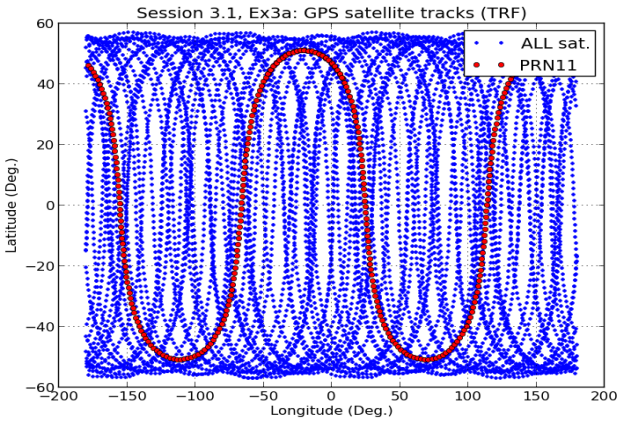
Hint: Take an approximate value of 20 000 km for the distance between satellite and receiver.

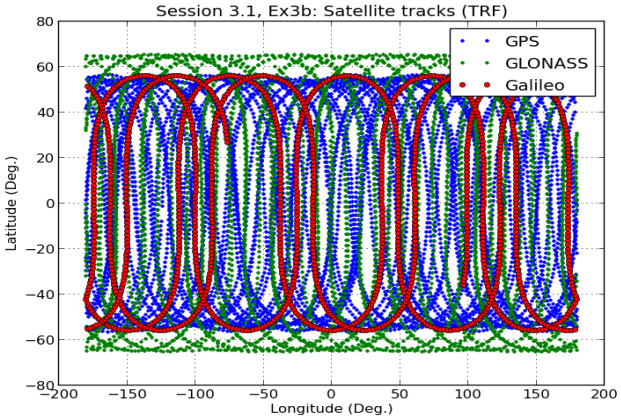
(b) How much has the satellite position changed approximately during this time? (Take an approximate value for the satellite velocity: for example, the file `2000-05-15.eci` with velocities (in km/s) referred to an inertial system.)

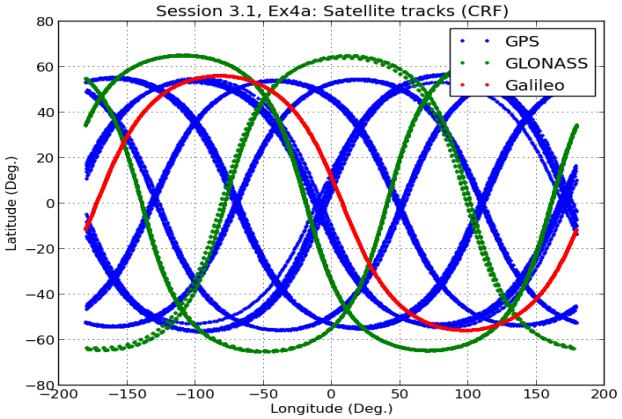
(c) How much has the ground receiver's position changed approximately due to Earth's rotation?

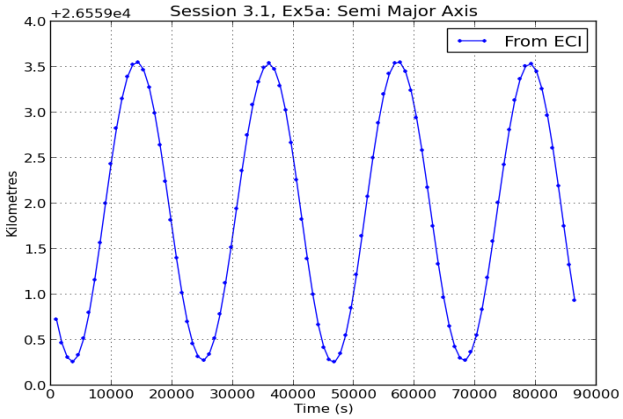
Graphs Session 3.1

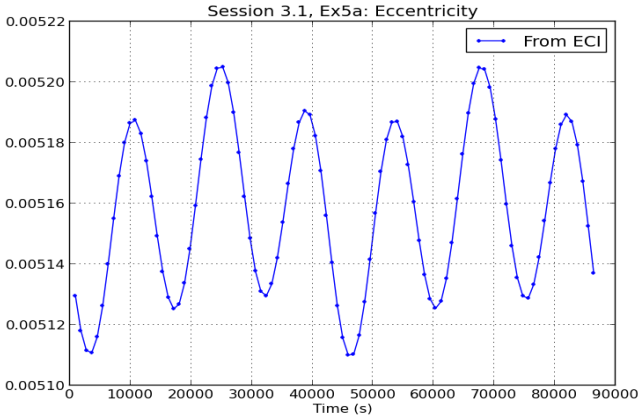


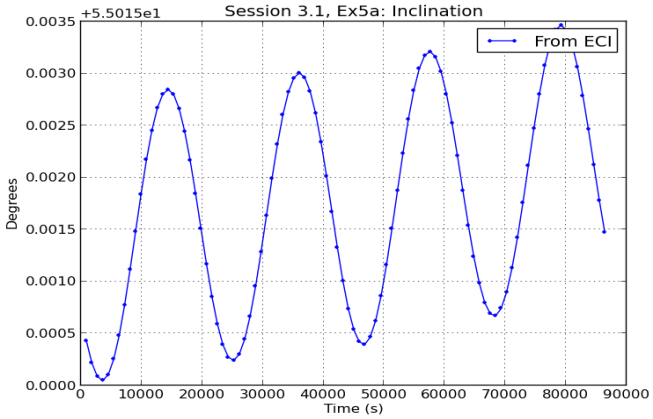


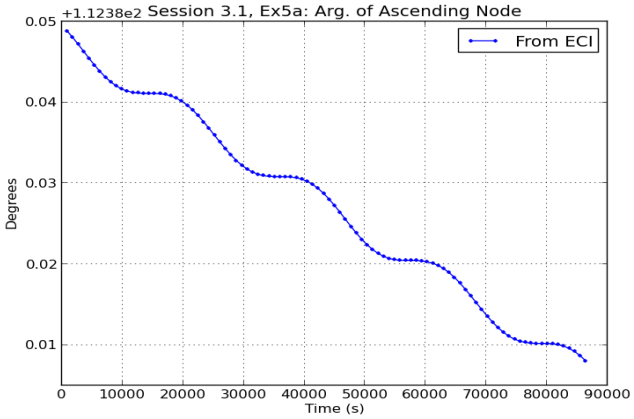


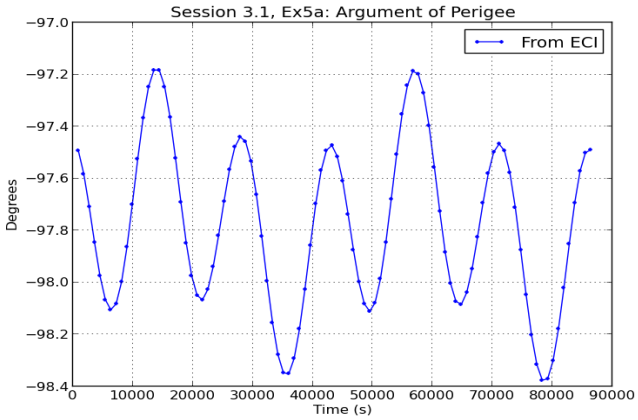


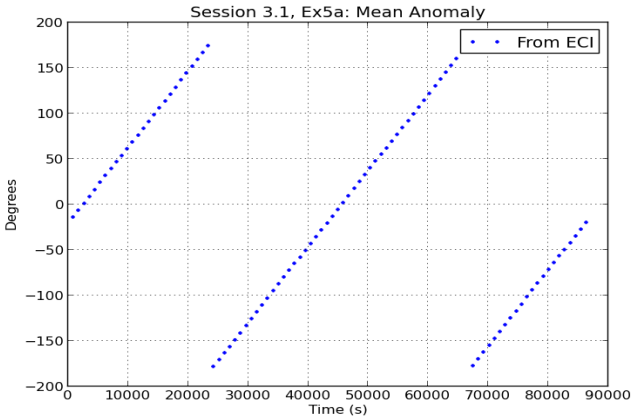


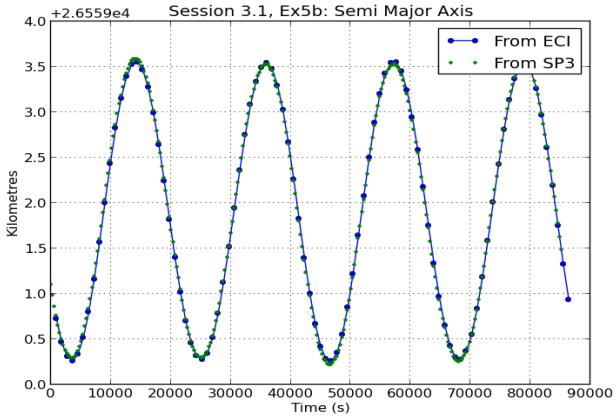


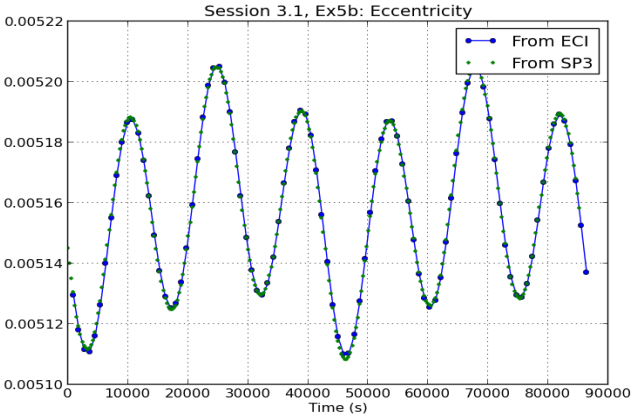


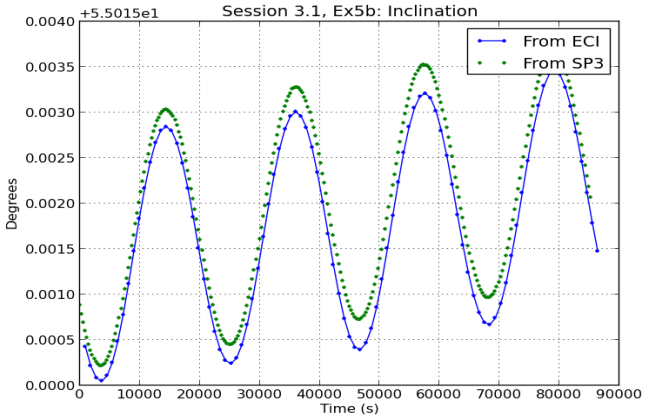


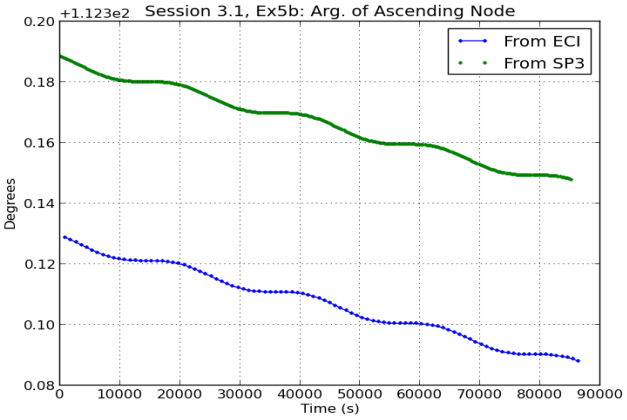

Session 3.1, Ex5a: Argument of Perigee
From ECI
Degrees
Time (s)

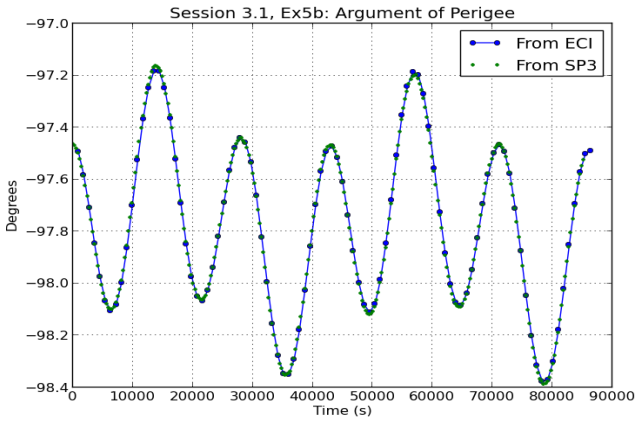

Session 3.1, Ex5a: Mean Anomaly
From ECI
Degrees
Time (s)

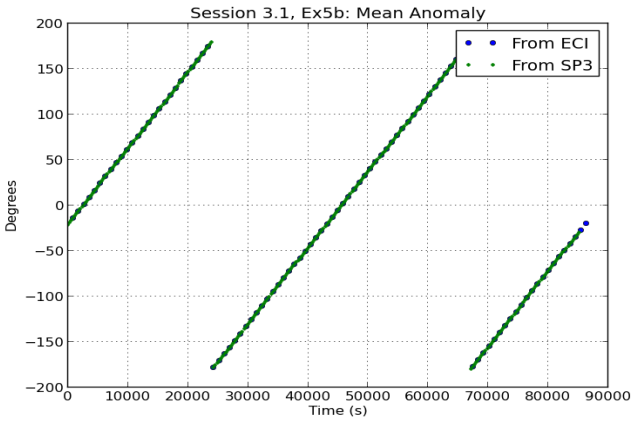

+2.6559e4
Session 3.1, Ex5b: Semi Major Axis
From ECI
From SP3
Kilometres
Time (s)

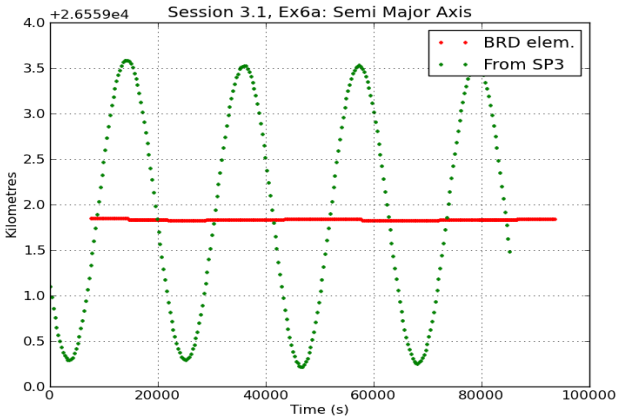

Session 3.1, Ex5b: Eccentricity
From ECI
From SP3
Time (s)

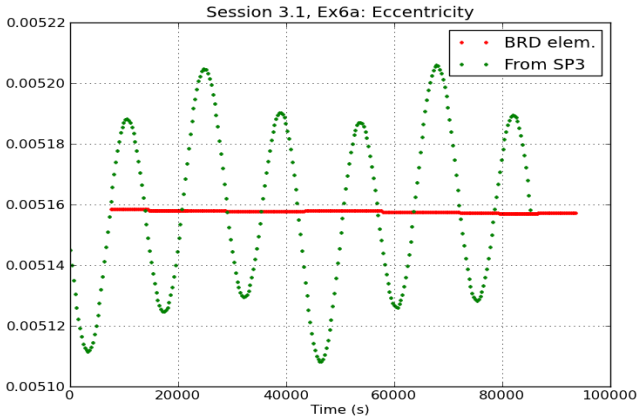

+5.5015e1
Session 3.1, Ex5b: Inclination
From ECI
From SP3
Degrees
Time (s)

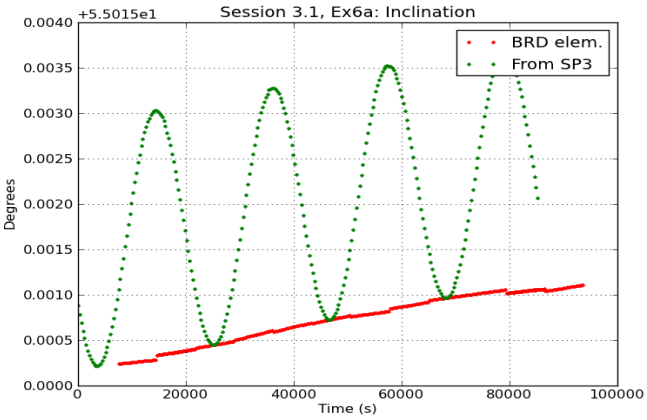

+1.123e2
Session 3.1, Ex5b: Arg. of Ascending Node
From ECI
From SP3
Degrees
Time (s)

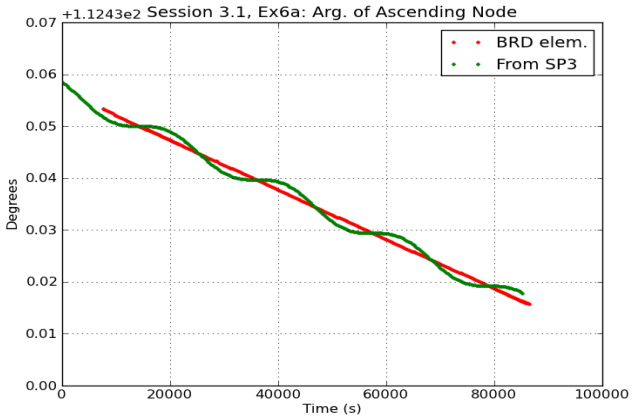

Session 3.1, Ex5b: Argument of Perigee
From ECI
From SP3
Degrees
Time (s)

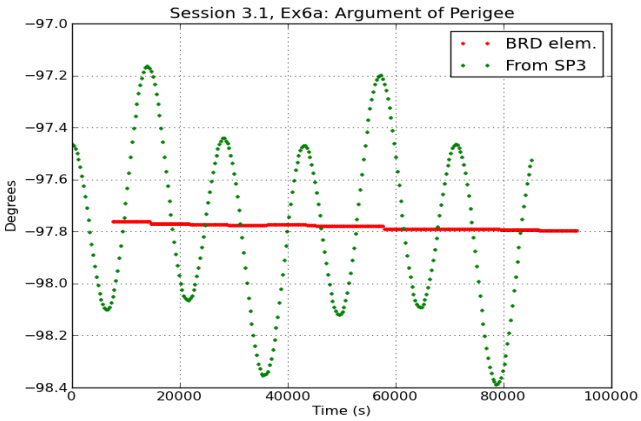

Session 3.1, Ex5b: Mean Anomaly
From ECI
From SP3
Degrees
Time (s)

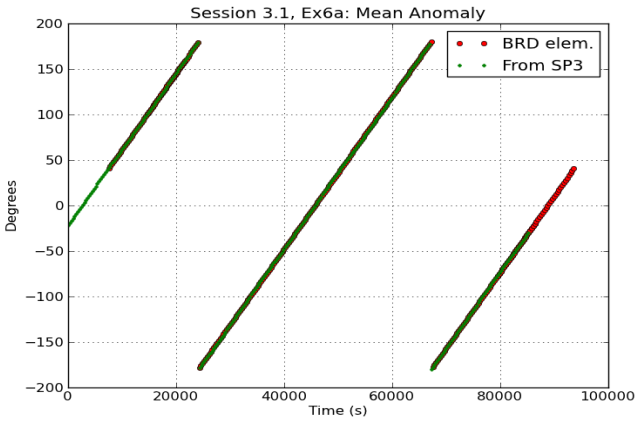

+2.6559e4
Session 3.1, Ex6a: Semi Major Axis
BRD elem.
From SP3
Kilometres
Time (s)

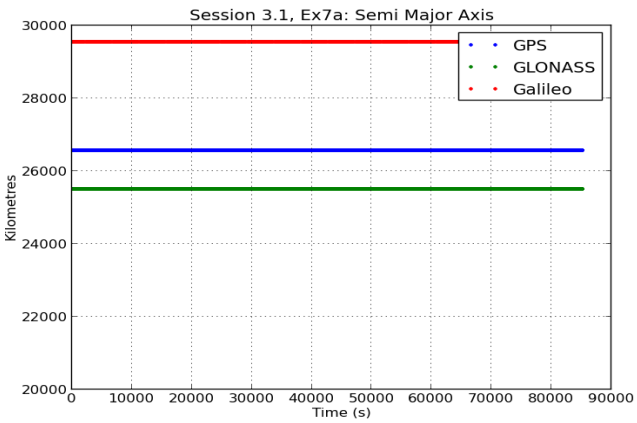

Session 3.1, Ex6a: Eccentricity
BRD elem.
From SP3
Time (s)

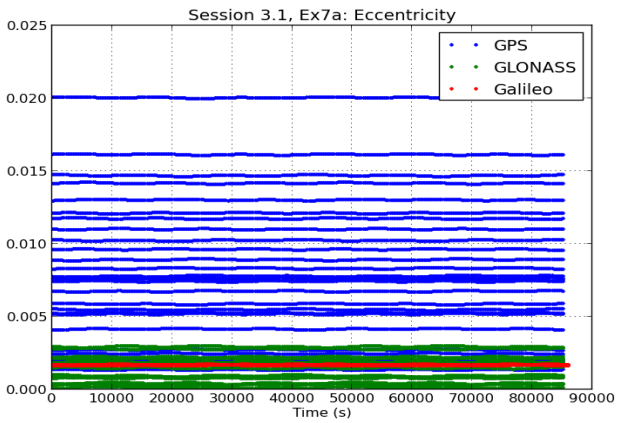

+5.5015e1
Session 3.1, Ex6a: Inclination
BRD elem.
From SP3
Degrees
Time (s)

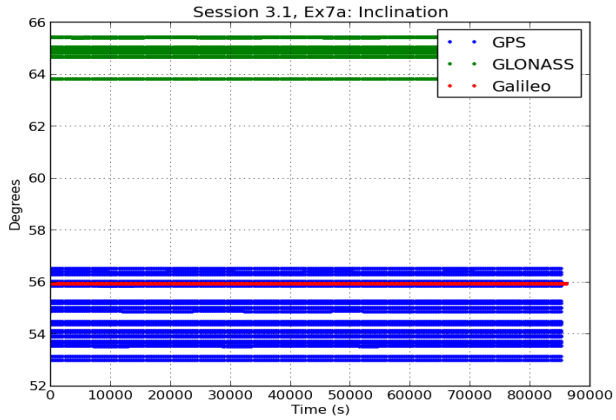

+1.1243e2
Session 3.1, Ex6a: Arg. of Ascending Node
BRD elem.
From SP3
Degrees
Time (s)

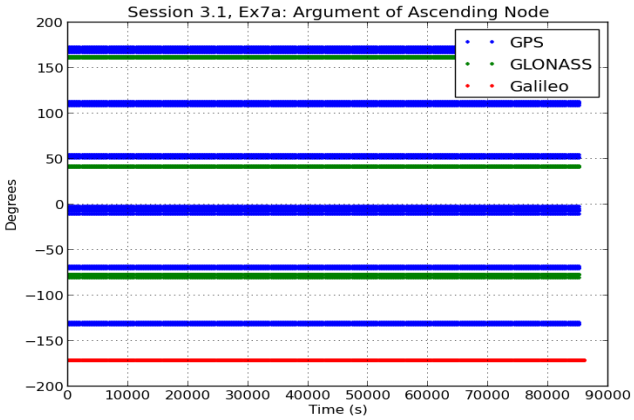

Session 3.1, Ex6a: Argument of Perigee
BRD elem.
From SP3
Degrees
Time (s)

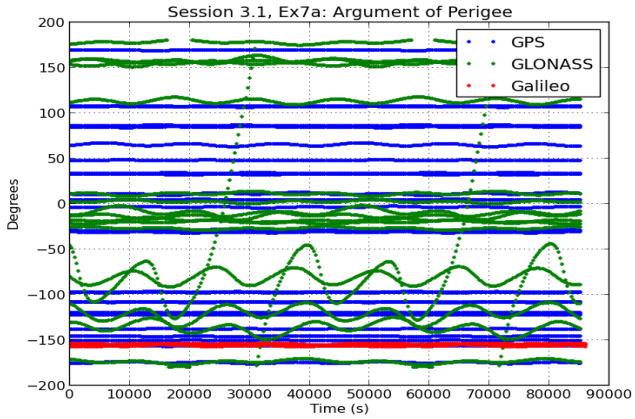

Session 3.1, Ex6a: Mean Anomaly
BRD elem.
From SP3
Degrees
Time (s)

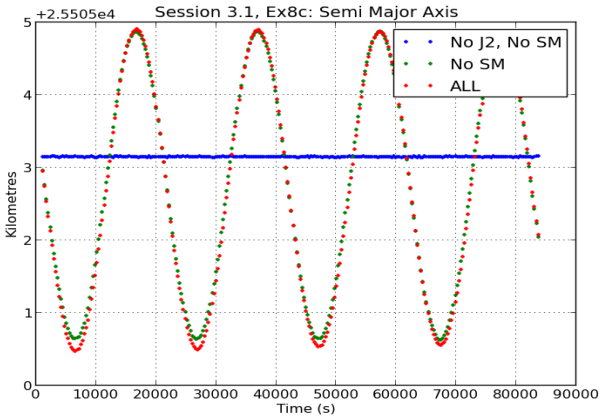

Session 3.1, Ex7a: Semi Major Axis
GPS
GLONASS
Galileo
Kilometres
Time (s)

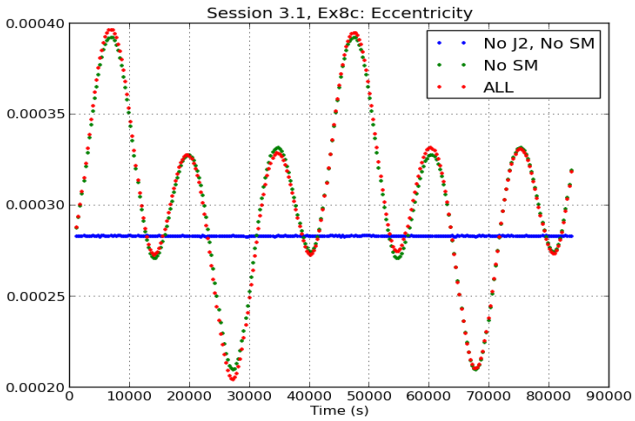

Session 3.1, Ex7a: Eccentricity
GPS
GLONASS
Galileo
Time (s)

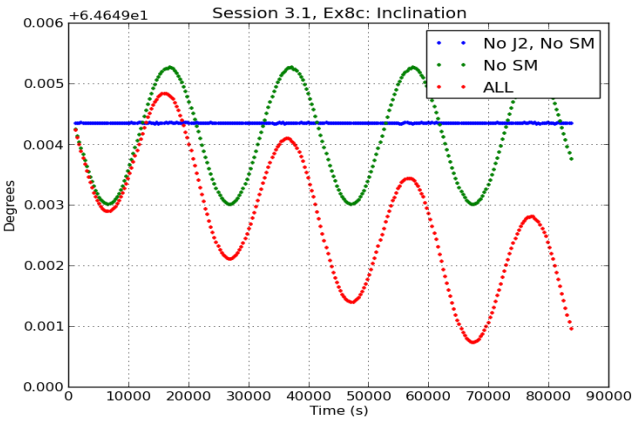

Session 3.1, Ex7a: Inclination
GPS
GLONASS
Galileo
Degrees
Time (s)

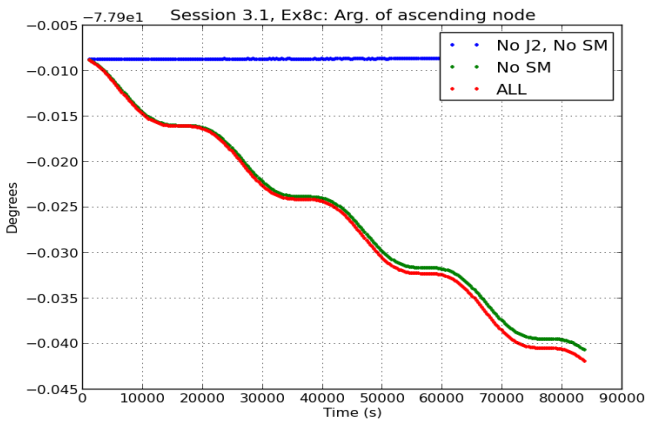

Session 3.1, Ex7a: Argument of Ascending Node
GPS
GLONASS
Galileo
Degrees
Time (s)

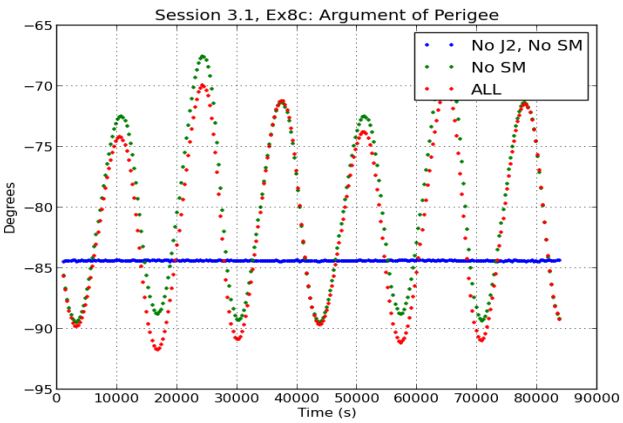

Session 3.1, Ex7a: Argument of Perigee
GPS
GLONASS
Galileo
Degrees
Time (s)

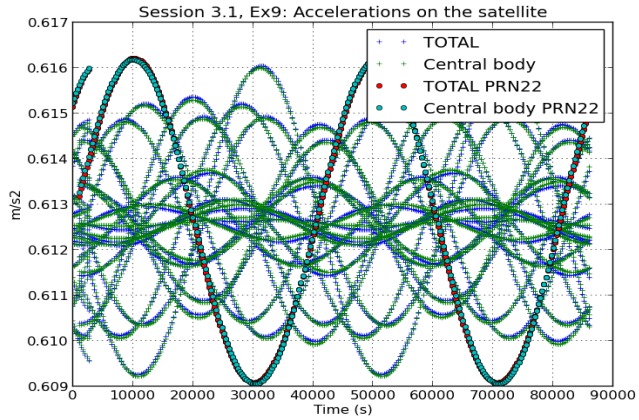

+2.5505e4
Session 3.1, Ex8c: Semi Major Axis
No J2, No SM
No SM
ALL
Kilometres
Time (s)

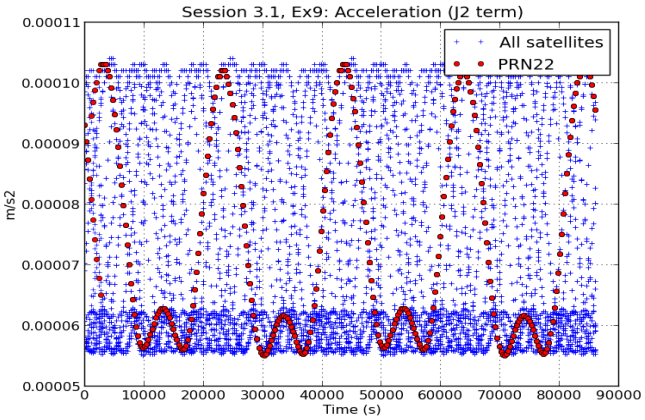

Session 3.1, Ex8c: Eccentricity
No J2, No SM
No SM
ALL
Time (s)

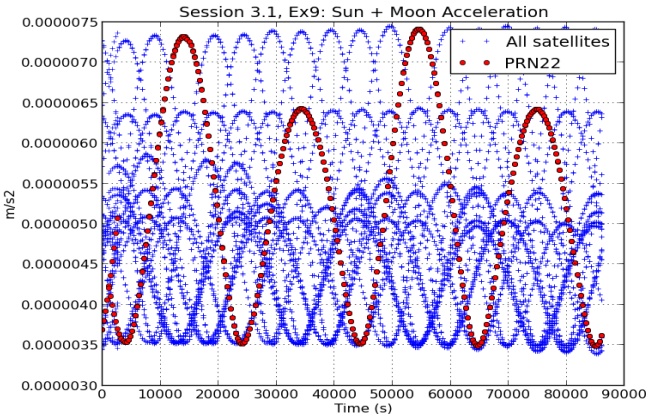

+6.4649e1
Session 3.1, Ex8c: Inclination
No J2, No SM
No SM
ALL
Degrees
Time (s)


−7.79e1
Session 3.1, Ex8c: Arg. of ascending node
No J2, No SM
No SM
ALL
Degrees
Time (s)


Session 3.1, Ex8c: Argument of Perigee
No J2, No SM
No SM
ALL
Degrees
Time (s)


Session 3.1, Ex9: Accelerations on the satellite
TOTAL
Central body
TOTAL PRN22
Central body PRN22
m/s2
Time (s)


Session 3.1, Ex9: Acceleration (J2 term)
All satellites
PRN22
m/s2
Time (s)


Session 3.1, Ex9: Sun + Moon Acceleration
All satellites
PRN22
m/s2
Time (s)

Session 3.2. Errors in Orbits and Clocks of GPS Satellites

Objectives

To study and quantify the errors in orbits and clocks of GPS satellites. To analyse the effect of the GPS Selective Availability (S/A) in the Signal In Space (SIS) domain. To analyse the effect of the Antenna Phase Centre (APC) bias on the orbit and clock determinations.

Files to use
`emr15681.sp3`, `emr15681.clk`, `igu15681_00.sp3`, `cod15681.clk`, `cod15681.eph`, `igu15681_12.sp3`, `cod10602.eph`, `cod10602.clk`, `brdc0250.10n`, `brdc0820.99n`, `brdc1230.00n`, `igs15681.sp3`, `igs10602.sp3`, `igs15681.clk`, `igr15681.sp3`, `igr15681.clk`, `cod15681.clk_05s`, `nga15681`, `apc15681`, `igs05_1569.atx`, `igs_pre1400.atx`

Programs to use
`gLAB_linux`, `graph.py`

Preliminary

The `gLAB` tool suite has a specific function to compare orbits and clocks from RINEX navigation files and/or SP3 files. This function is not available in the `gLAB` GUI and must be executed in console mode.

As an example, the following command-line sentence provides the difference between coordinates and clock offsets computed from the broadcast navigation message file `brdc0250.10n` and IGS precise orbit and clock file `igs15681.sp3`:[20]

Execute (in a single line):

```
gLAB_linux -input:nav brdc0250.10n -input:SP3 igs15681.sp3
           -input:ant igs05_1569.atx > dif.all
```

It must be pointed out that the satellite coordinates of the SP3 IGS files are referred to the mass centre of the satellite. Thus, an ANTEX file is required to transform the `igs15681.sp3` file coordinates relative to the Satellite Mass Centre (MC) to the APC, see section 5.6.3, Volume I.[21] The ANTEX file `igs05_1569.atx` contains the APCs associated with the orbits and clocks of file `igs15681.sp3`, as indicated in its header.

[20] This file corresponds to the IGS final combined product and is accurate at the level of a few centimetres (see Table 3.10, Volume I).

[21] The APC corrections can be disabled using the option '`--model:satphasecenter`' (with double '`--`'). Using this option, coordinates are given relative to the satellite MC.

Note: The APCs are only applied to the SP3 files. Moreover, when gLAB *compares two SP3 files, the APC file applies only to the second file. Consequently, if both SP3 files are relative to the same APC, this correction must be disabled.*

By default, the sentence above computes the differences in the time tags of the second file (i.e. every 900 s for the `igs15681.sp3` file in this example). Nevertheless, in order to have results every 300 s, the orbits and clocks of the second file can be interpolated by setting the following arguments to `gLAB_linux`:

```
gLAB_linux -input:nav brdc0250.10n -input:SP3 igs15681.sp3
       -pre:dec 300 -model:clock:deg 1 -model:orbit:deg 10
                          -input:ant igs05_1569.atx > dif.all
```

- `[-pre:dec 300]`: Indicates the output sampling rate. In this case, the results will be written every 300 seconds.
- `[-model:clock:deg 1]`: Indicates the polynomial degree used to interpolate the satellite clocks. A one-degree polynomial is used in this case.
- `[-model:orbit:deg 10]`: Indicates the polynomial degree used to interpolate the SP3 orbits (a 10-degree polynomial is used by default).

Note: Both conditions `[-pre:dec]` *and* `[-model:clock:deg]` *are needed to interpolate SP3 files, because, by default,* gLAB *only computes coordinates when the clocks are available. Another possibility is to disable the satellite clock's computation by setting the option* `[--model:satclocks]` *instead of* `[-model:clock:deg]`.

The output file `dif.all` contains lots of information on the comparison of both files (differences of coordinates and clocks, statistics, etc.). This information is provided in three main types of data blocks, labelled as: `SATDIFF` (satellite orbit and clock differences between both files), `SATSTAT` (statistics per satellite) and `SATSTATTOT` (overall statistics). The rows labelled by `SATDIFF`, `SATSTAT` and `SATSTATTOT` contain the fields described in Tables 3.2 to 3.4, where the Signal in Space Range Error (SISRE) is given by

$$\text{SISRE}=\sqrt{(\text{radDiff}-\text{clkDiff})^2+(\text{ctDiff}^2+\text{atDiff}^2)/49}$$

Note that, in the computation of SISRE, the clkDiff term is included in field 7 of the tables, but not in field 8.

For more details execute: `gLAB_linux -messages`

Table 3.2: Description of data in the SATDIFF rows.

Field	Content
1	'SATDIFF'
2	Year
3	Day of Year (DoY)
4	Seconds of day (seconds)
5	GNSS (GPS, GAL, GLO or GEO)
6	PRN satellite identifier
7	SISRE difference (m)
8	SISRE orbit-only difference (m)
9	3D orbit difference (m)
10	clkDiff: Clock difference (m)
11	radDiff: Radial position difference (m)
12	atDiff: Along-track position difference (m)
13	ctDiff: Cross-track position difference (m)

Table 3.3: Description of data in the SATSTAT rows.

Field	Content
1	'SATSTAT'
2	GNSS (GPS, GAL, GLO or GEO)
3	PRN satellite identifier
4	Number of samples
5–7	SISRE comparisons [mean,RMS,stdDev]
8–10	SISRE orbit-only comparisons [mean,RMS,stdDev]
11–13	3D orbit discrepancies [mean,RMS,stdDev]
14–16	Clock discrepancies [mean,RMS,stdDev]
17–19	Radial position discrepancies [mean,RMS,stdDev]
20–22	Along-track position discrepancies [mean,RMS,stdDev]
23–25	Cross-track position discrepancies [mean,RMS,stdDev]

Table 3.4: Description of data in the SATSTATTOT rows.

Field	Content
1	'SATSTATTOT'
2	Number of samples
3–5	SISRE comparisons [mean,RMS,stdDev]
6–8	SISRE orbit-only comparisons [mean,RMS,stdDev]
9–11	3D orbit discrepancies [mean,RMS,stdDev]
12–14	Clock discrepancies [mean,RMS,stdDev]
15–17	Radial position discrepancies [mean,RMS,stdDev]
18–20	Along-track position discrepancies [mean,RMS,stdDev]
21–23	Cross-track position discrepancies [mean,RMS,stdDev]

Development

Session files.

Copy and uncompress these session files in the working directory:

```
cp ~/GNSS/PROG/SES32/* .
cp ~/GNSS/FILES/SES32/* .
gzip -d *.Z  *.gz
```

1. GPS broadcast orbit and clock accuracy

(a) Compute the discrepancies between the broadcast and the IGS precise orbit and clock determinations. Represent such discrepancies graphically on radial, along-track and cross-track orbit error components. Plot the clock error.

Complete the following steps:

1. Computations:

i. Compute orbit and clock differences:

```
gLAB_linux -input:nav brdc0250.10n
                -input:SP3 igs15681.sp3
                   -input:ant igs05_1569.atx > dif.all
```

ii. Select rows labelled as `SATDIFF`:

```
grep SATDIFF dif.all > dif.sel
```

2. Plot results:[22]

From file `dif.sel`, plot 'all satellites' and overlap the plot of 'PRN02':

3D error:

```
graph.py -f dif.sel -x4 -y9
                  -f dif.sel -x4 -y9 -c'($6==2)' -so
```

Radial error:

```
graph.py -f dif.sel -x4 -y11
                 -f dif.sel -x4 -y11 -c'($6==2)' -so
```

Along-track error:

```
graph.py -f dif.sel -x4 -y12
                 -f dif.sel -x4 -y12 -c'($6==2)' -so
```

[22] Note that, in the Windows OS, the instruction `'($6==2)'` or `'($6=="2")'` must be written as `"($6=='2')"`; that is, replacing (") by (').

Cross-track error:

```
graph.py -f dif.sel -x4 -y13
            -f dif.sel -x4 -y13 -c'($6==2)' -so
```

Clock error:

```
graph.py -f dif.sel -x4 -y10
            -f dif.sel -x4 -y10 -c'($6==2)' -so
```

(b) Using IGS orbits and clocks as reference values (i.e. the truth), give a rough assessment of the broadcast orbit and clock accuracy. Identify possible patterns or biases. Compare results with those in Table 3.10, Volume I.

(c) Reprocess using `-pre:dec 300 -model:clock:deg 1`. Zoom in on previous plots for satellite PRN02 for the time interval between 0 and 30 000 seconds. Observe the different arcs appearing in the figures and discuss possible explanations.

2. GPS satellites' broadcast orbits and clocks and APC

The effect of the APC associated with the satellite orbit and clock determinations will be analysed in this exercise.

Execute again the same instructions (or use the previous results):

```
gLAB_linux -input:nav brdc0250.10n
           -input:SP3 igs15681.sp3
           -input:ant igs05_1569.atx  > dif.all

grep  SATDIFF dif.all > dif.sel
```

(a) Represent graphically the discrepancies in the radial component and clock error.

Radial error and clock error:

```
graph.py -f dif.sel -x4 -y11 -so -l Radial
               -f dif.sel -x4 -y10 -so -l clock
```

Are there any biases appearing in the radial component? And in the clocks? Discuss the possible source of such biases.
To help this analysis, the following plots can be produced:

Radial − clock error:

```
graph.py -f dif.sel -x4 -y'($10-$11)' -so
                          -l "Radial - Clock error"
```

Radial error and clock shifted by 2.4 m:

```
graph.py -f dif.sel -x4 -y'($11+2.4)' -so -l Radial
                  -f dif.sel -x4 -y10 -so -l clock
```

Discuss the results found. What might be the reason for the correlation between the radial orbit and clock errors seen in the plot?

Hint: Consider that different APCs are used to estimate the broadcast and the IGS orbit and clock products.

(b) Repeat the process, but using the ANTEX file `gps_brd.atx` with the APCs used to estimate the broadcast orbits and clocks (see the table in Appendix F, Volume I).

1. Computations:

```
gLAB_linux -input:nav brdc0250.10n
               -input:SP3 igs15681.sp3
                    -input:ant gps_brd.atx > dif.all

grep SATDIFF dif.all > dif.sel
```

2. Plot the results:

Individual plots for each error component, `dif.sel`:

Radial error:

```
graph.py -f dif.sel -x4 -y11 -so -t "Radial"
```

Along-track error:

```
graph.py -f dif.sel -x4 -y12 -so -t "Along track"
```

Cross-track error:

```
graph.py -f dif.sel -x4 -y13 -so -t "Cross track"
```

Clock error:

```
graph.py -f dif.sel -x4 -y10 -so -t "Clock"
```

Compare these plots with those obtained using the ANTEX file `igs05_1569.atx`. Have the biases in the radial component disappeared? And in the clocks? Justify the results.

3. GPS satellites' broadcast orbit and clock accuracy: S/A on

Repeat the previous exercise using files `brdc0820.99n` and `igs10022.sp3` corresponding to 23 March 1999, with S/A on. As in the previous exercise, use the ANTEX file `gps_brd.atx` file with the APCs applied for the broadcast orbits.

Plot the 3D orbit error as a function of time. Produce another plot comparing the 3D orbit error with the clock error.

Complete the following steps:

1. Computations:

```
gLAB_linux -input:nav brdc0820.99n
                 -input:SP3 igs10022.sp3
                       -input:ant gps_brd.atx > dif.all

grep SATDIFF dif.all > dif.sel
```

2. Plot the results:

3D orbit error plot:

```
graph.py -f dif.sel -x4 -y9 -so -l "Orbit 3D error"
```

3D orbit and clock error plot:

```
graph.py -f dif.sel -x4 -y9 -so -l "Orbit 3D error"
                -f dif.sel -x4 -y10 -so -l "Clock error"
```

Compare the results with those of the previous exercises with S/A off. Which error component was mostly affected by the S/A (orbits or clocks)?

4. Date 2 May 2000: S/A off

On 2 May 2000 the S/A was switched off. This effect can be clearly seen by analysing the broadcast navigation messages of this day.

(a) To visualise the changes when the S/A is off, repeat the previous exercise using the files `brdc1230.00n` and `igs10602.sp3` corresponding to such a day.

Complete the following steps:

1. Computations:

```
gLAB_linux -input:nav brdc1230.00n
              -input:SP3 igs10602.sp3
                    -input:ant gps_brd.atx > dif.all

grep SATDIFF dif.all > dif.sel
```

2. Plot the results:

3D orbit error and the clock error:

```
graph.py -f dif.sel -x4 -y9 -so -l "Orbit 3D error"
         -f dif.sel -x4 -y10 -so -l "Clock error"
```

Individual plots for each error component:

Clock error:

```
graph.py -f dif.sel -x4 -y10 -so -t "Clock"
```

3D error:

```
graph.py -f dif.sel -x4 -y9 -so -t "3D error"
```

Radial error:

```
graph.py -f dif.sel -x4 -y11 -so -t "Radial"
```

Along-track error:

```
graph.py -f dif.sel -x4 -y12 -so -t "Along track"
```

Cross-track error:

```
graph.py -f dif.sel -x4 -y13 -so -t "Cross track"
```

(b) Compare errors before and after the S/A was switched off. Which error component of the SIS was mainly affected by the S/A at that

time? Quantify the reduction in SIS error budget after switching the S/A off and its impact on user domain accuracy (see exercise 7 in session 1.1).

The paper 'Broadcast vs. Precise GPS Ephemerides: A Historical Perspective' [Warren and Raquet, 2003] provides additional background for readers wishing to explore these issues further.

5. Comparison of Precise Orbit and Clock Products

The precise orbit and clock products computed by different centres will be compared in this exercise.

(a) Compare the discrepancies between the orbit and clock estimates of Centre for Orbit Determination in Europe (CODE) `cod15681.eph` and IGS final combined product `igs15681.sp3`.
Note: As all these products are referred to the MC, turn off the APC correction by adding '`--model:satphasecenter`*'.*[23]

Complete the following steps:

1. Computations:

```
gLAB_linux -input:SP3 cod15681.eph
               -input:SP3 igs15681.sp3
                     --model:satphasecenter > dif.all

grep SATDIFF dif.all > dif.sel
```

2. Plot the results:

3D error:

```
graph.py -f dif.sel -x4 -y9 -so
```

Radial, along-track and cross-track errors:

```
graph.py -f dif.sel -x4 -y11 -l Radial -f dif.sel
      -x4 -y12 -l Alon -f dif.sel -x4 -y13 -l Cross
```

Clock error:

```
graph.py -f dif.sel -x4 -y10 -so
```

What is the discrepancy level between both products? Discuss the drift and bias of the clock seen in the plot. Can a bias, common to all satellites, affect user positioning?

(b) Compare the discrepancies between the orbit and clock estimates of Energy Mines and Resources (EMR) `emr15681.sp3` and the IGS final combined product `igs15681.sp3`.

1. Computations:

```
gLAB_linux -input:SP3 emr15681.sp3
               -input:SP3 igs15681.sp3
                     --model:satphasecenter > dif.all

grep SATDIFF dif.all > dif.sel
```

[23] Notice the double '-' in '`--model:satphasecenter`'.

2. Plot the results:

Clock error:

```
graph.py -f dif.sel -x4 -y10
```

Produce similar plots as in the previous section for the 3D, radial, along-track and cross-track errors.

(c) SP3 file `nga15681` comes from the National Geospatial-Intelligence Agency (NGA)[24] and contains precise orbits and clocks computed using the same APC as with the GPS broadcast orbits.[25]

Analyse the discrepancies between the coordinates and clocks of this file and the IGS final combined product `igs15681.sp3`.

Note: Like the IGS `igs15681.sp3` *file, satellite coordinates in the* `nga15681` *file are relative to the MC.*

Execute for instance:

```
gLAB_linux -input:SP3 nga15681 -input:SP3 igs15681.sp3
                        --model:satphasecenter> dif.all

grep SATDIFF dif.all > dif.sel
```

Produce similar plots as in the previous case and discuss the results.

(d) The SP3 file `apc15681` contains the same orbits and clocks as the previous `nga15681` file, but with the coordinates relative to the APC (the same APC as the GPS broadcast orbits). This file is also from the NGA.[26]

Analyse the discrepancies between the coordinates and clocks of this file and the IGS final combined product `igs15681.sp3`.

Note: Unlike the IGS `igs15681.sp3` *file, the satellite coordinates in the* `apc15681` *file are relative to the satellite's APC. Therefore, the APC correction must be applied to the IGS file to relate both coordinates to their respective APC.*

Note also that the APC corrections (from the ANTEX file) apply only over the second SP3 file (the `igs15681.sp3` *file, next).*

1. Computations:

```
gLAB_linux -input:SP3 apc15681 -input:SP3 igs15681.sp3
                    -input:ant igs05_1569.atx > dif.all

grep SATDIFF dif.all > dif.sel
```

[24] See *http://earth-info.nga.mil/GandG/sathtml/ephemeris.html*. See also *ftp://ftp.nga.mil/pub2/gps/pedata*.

[25] See table in Appendix F, Volume I.

[26] *ftp://ftp.nga.mil/pub2/gps/apcpe*.

2. Plot the results:

Radial − clock error:

```
graph.py -f dif.sel -x4 -y'($10-$11)' -so
                                  -l "Radial-Clock error"
```

Radial error and clock shifted by 2.4 m :

```
graph.py -f dif.sel -x4 -y'($11+2.4)' -l Radial
                             -f dif.sel -x4 -y10 -l clock
```

Compare the results with those of exercises 2 and 3 with broadcast orbits. Discuss the biases observed in `radial-clock error`.

6. IGS product comparison: final, rapid and ultra-rapid

The accuracy of final, rapid and ultra-rapid IGS orbit and clock products is assessed in this exercise.

(a) *Rapid vs final products*
Compute the discrepancies between the IGS rapid orbit and clock products `igr15681.sp3` and `igr15681.clk`, and the IGS final products `igs15682.sp3` and `igs15681.clk`.

Execute:[27]

```
gLAB_linux -input:orb igr15681.sp3 -input:clk igr15681.clk
           -input:orb igs15681.sp3 -input:clk igs15681.clk
 -input:ant igs05_1569.atx --model:satphasecenter> dif.all

grep SATDIFF dif.all > dif.sel
```

Plot the results as in previous exercises.

'Grep' label `SATSTATTOT` in the `dif.all` file, and give the `mean`, `RMS` and σ of the discrepancies (see message description in Table 3.2).

Execute:

```
grep SATSTATTOT dif.all > dif.stat
```

What is the discrepancy level found in the different error components? Compare the values with those in Table 3.10 Volume I. How will this error affect user position accuracy?

(b) *Ultra-rapid versus final products*
Compute the discrepancies between the ultra-rapid orbit and clock file `igu15681_00.sp3` estimates and the IGS final combined product `igs15682.sp3`.

Execute:

```
gLAB_linux -input:SP3 igu15681_00
           -input:SP3 igs15681.sp3
           --model:satphasecenter> dif.all

grep SATDIFF dif.all > dif.sel
```

[27] Note that the sentence used to run `gLAB` is different from previous cases, because the precise clock files `*.clk` are used together with the orbit and clock files `*.sp3`.

Plot the results and discuss figures as in previous exercises.

'Grep' label `SATSTATTOT` in the `dif.all` file, and give the `mean`, `RMS` and σ of the discrepancies. Discuss the results.

(c) *Ultra-rapid product overlap*

Compute the discrepancies between the two consecutive IGS ultra-rapid orbit and clock files `igu15681_00.sp3` and `igu15681_12.sp3`.

Execute:

```
gLAB_linux -input:SP3 igu15681_00.sp3
           -input:SP3 igu15681_12.sp3
           --model:satphasecenter> dif.all

grep SATDIFF dif.all > dif.sel
```

Plot the results and discuss figures as in previous exercises.

'Grep' label `SATSTATTOT` in the `dif.all` file, and give the `mean`, `RMS` and σ of the discrepancies. Discuss the results.

7. Analysis of GPS satellite clock interpolation

The accuracy of the interpolation of IGS high-rate clock files is assessed in this exercise.

(a) *Interpolating 300 s clock files*

Interpolate the time-stepped 300 s clocks of file `igs15681.sp3` on a 30 s time grid using a one-degree polynomial. Compare the results with the precise clocks of file `igs15681.clk`, at 30 s.

Complete the following steps:

1. Computations:[28]

```
gLAB_linux -input:SP3 igs15681.sp3
     -input:orb igs15681.sp3 -input:clk igs15681.clk
     --model:satphasecenter
     -pre:dec 30 -model:clock:deg 1 > dif.all

grep SATDIFF dif.all > dif.sel
```

2.- Plotting results:

```
Clock error:
graph.py -f dif.sel -x4 -y10
```

What is the level of the interpolation error found?

In this example the satellite clocks are under the S/A=off condition. Taking into account the results of previous exercises with S/A on, what would happen in this case? (Note that this item will be analysed in more detail in the next exercise 8).

[28] Add the instructions `-model:clock:deg 1` to interpolate the clocks with a one-degree polynomial, and `-pre:dec 30` to output results every 30 s. Note that, when applying these conditions, orbits are interpolated with a 10-degree polynomial, by default. This default value can also be changed by the option '`-model:orbit:deg`'. More details can be found by executing `gLAB_linux -help`.

(b) *Interpolating 30 s clock files*

Interpolate the 30 s clocks of file `cod15681.clk` using a 5 s time grid and compare the results with the precise clocks at a 5 s sampling rate of file `cod15681_05.clk`. Complete the following steps:

1. Computations:

```
gLAB_linux -input:orb cod15681.eph
    -input:clk cod15681.clk -input:orb cod15681.eph
    -input:clk cod15681.clk_05s --model:satphasecenter
    -pre:dec 5 -model:clock:deg 1 > dif.all

grep SATDIFF dif.all > dif.sel
```

2. Plot the results:

Clock error:

```
graph.py -f dif.sel -x4 -y10
```

What is the accuracy of the interpolated clocks?

Comment: The paper 'Characterisation of Periodic Variations in the GPS Satellite Clocks' [Kenneth et al., 2008] is recommended to readers wishing to find a deeper study of this issue.

8. GPS satellite clock interpolation with S/A on and off

The accuracy of interpolation of the IGS high-rate clock files is assessed in this exercise with S/A on and S/A off.

(a) *Interpolating 300 s clock files*

Interpolate the 300 s clocks of the file `cod10602.eph` using a 30 s time grid and compare the results with the precise clocks of the file `cod10602.clk`, at a 30 s sampling rate.

Note: These files correspond to 2 May 2000, when the S/A was switched off.

Complete the following steps:

1. Computations:[29]

```
gLAB_linux -input:SP3 cod10602.eph
       -input:orb cod10602.eph -input:clk cod10602.clk
       --model:satphasecenter
       -pre:dec 30 -model:clock:deg 1 > dif.all

grep SATDIFF dif.all > dif.sel
```

[29] Add the instructions `-model:clock:deg 1` to interpolate clocks with a one-degree polynomial, and `-pre:dec 30` to output results every 30 s. Note that, when applying these conditions, orbits are interpolated with a 10-degree polynomial, by default. This default value can also be changed by the option '`-model:orbit:deg`'. More details can be found by executing `gLAB_linux -help`.

2. Plot the results:

Clock error. Overlap PRN01 plot and:

A. Zoom between [0:25000] seconds

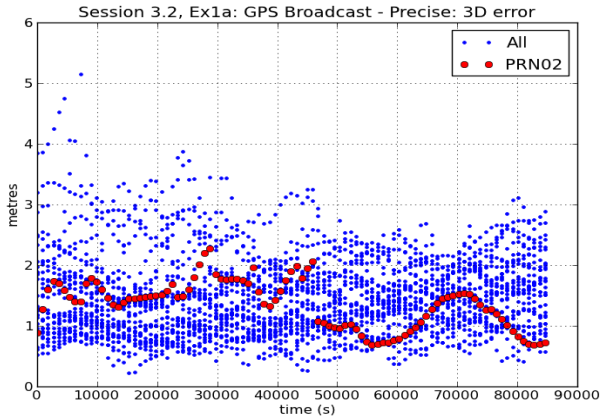
```
graph.py -f dif.sel -x4 -y10 -f dif.sel -c '($6==1)' -x4
                                          -y10 --xx 25000
```

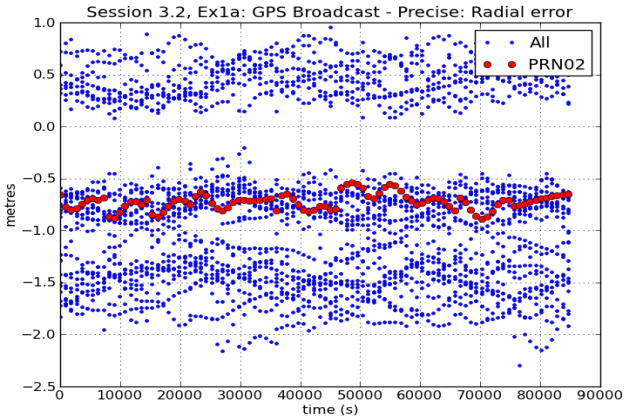
B. Zoom between [0:3000] seconds

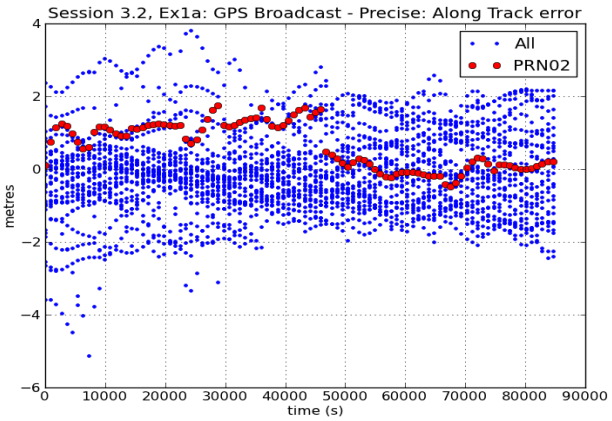
```
graph.py -f dif.sel -x4 -y10 -f dif.sel -c '($6==1)'
                                    -so -x4 -y10 --xx 3000
```

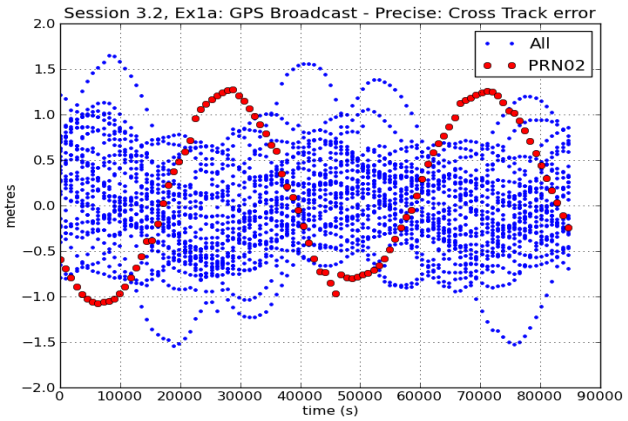
Would it make sense to interpolate the satellite clocks with S/A on?

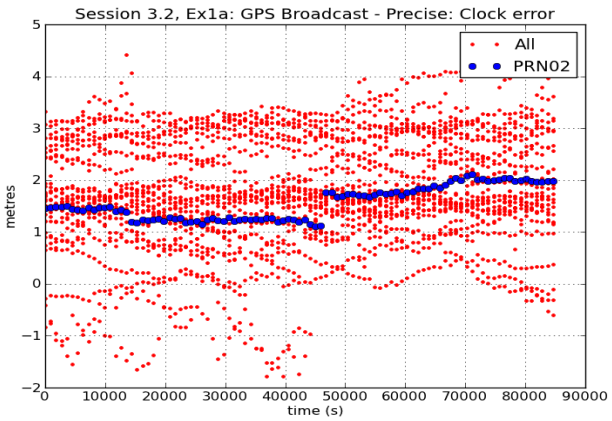
Graphs Session 3.2

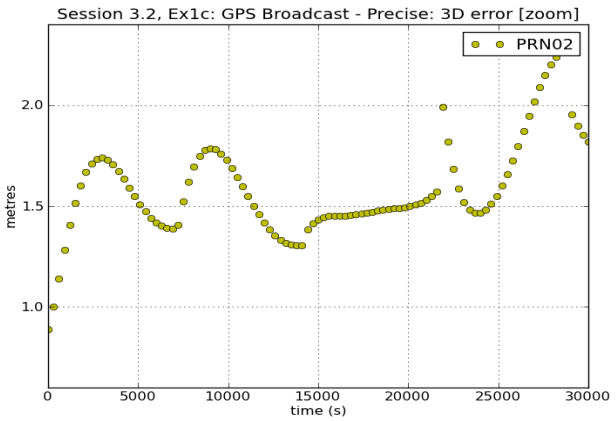


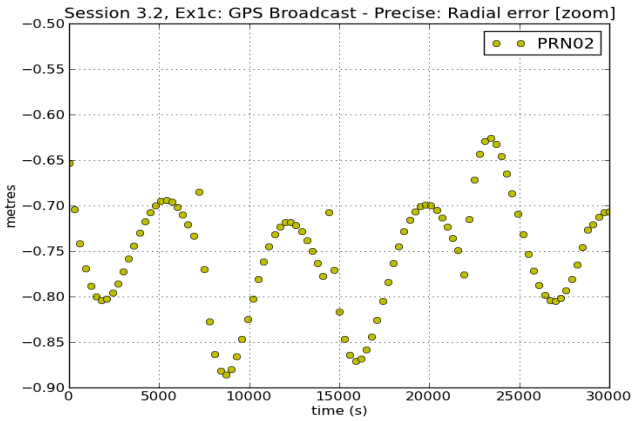


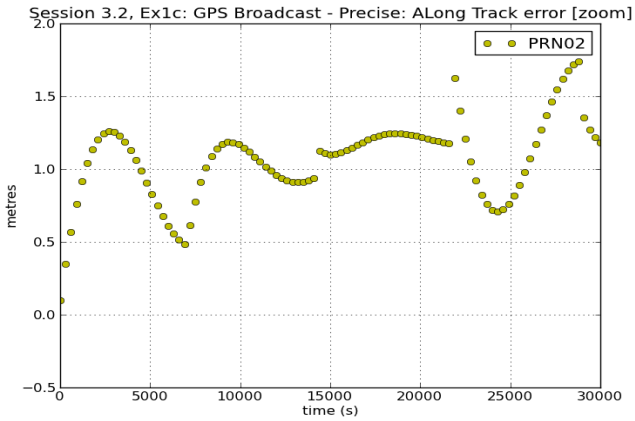


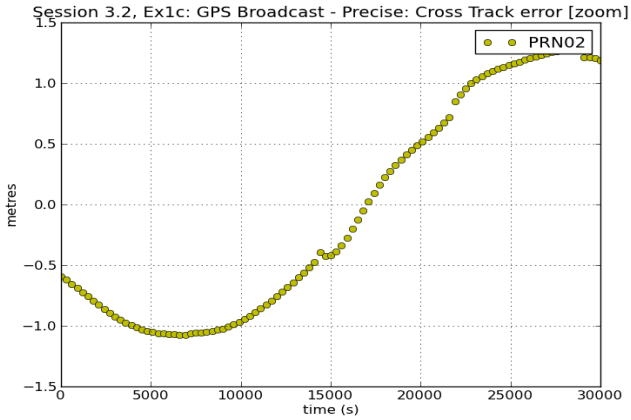


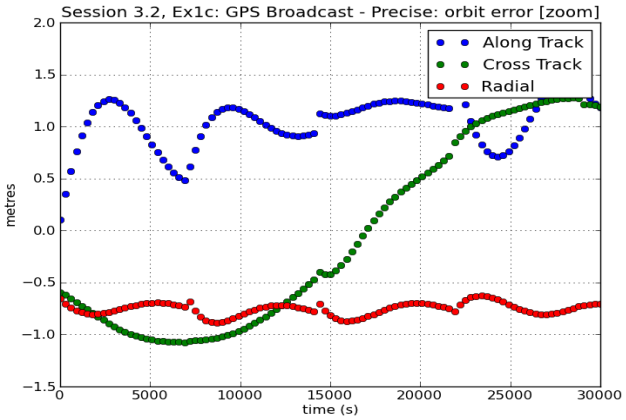


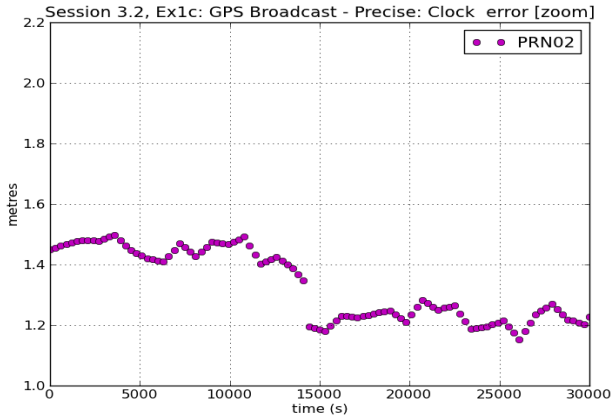


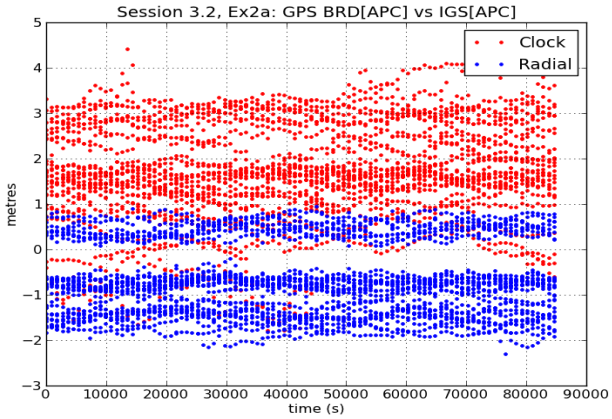


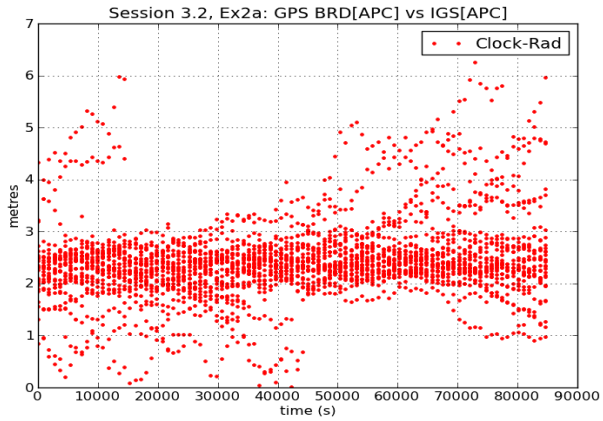


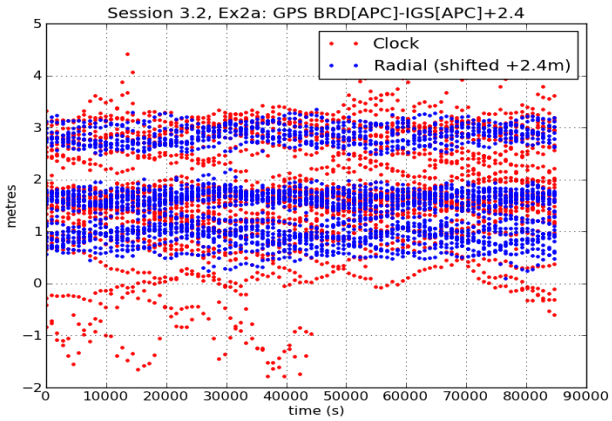

Session 3.2, Ex1c: GPS Broadcast - Precise: Cross Track error [zoom]
PRN02
metres
time (s)

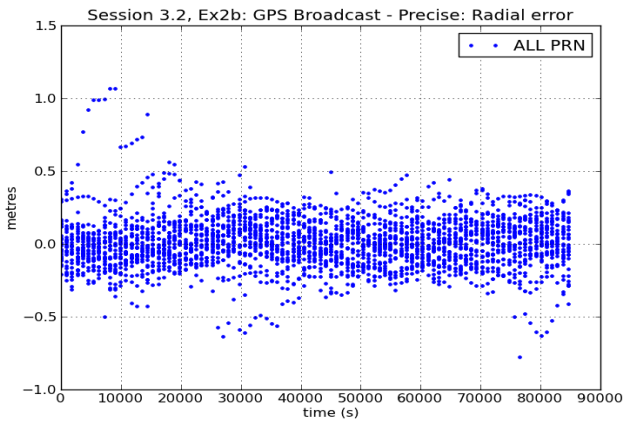

Session 3.2, Ex1c: GPS Broadcast - Precise: orbit error [zoom]
Along Track
Cross Track
Radial
metres
time (s)

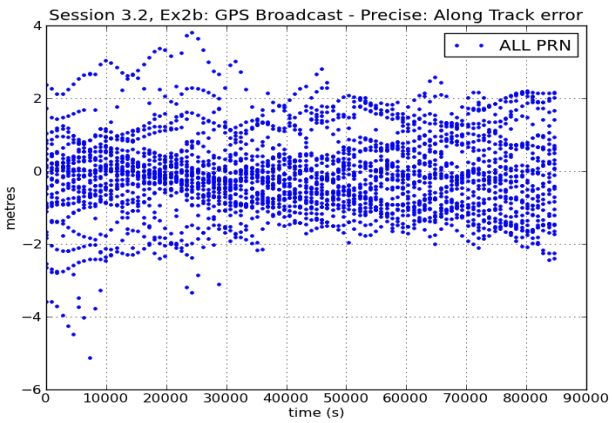

Session 3.2, Ex1c: GPS Broadcast - Precise: Clock error [zoom]
PRN02
metres
time (s)

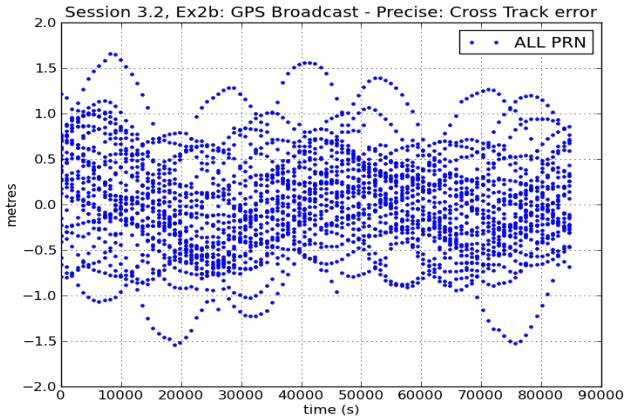

Session 3.2, Ex2a: GPS BRD[APC] vs IGS[APC]
Clock
Radial
metres
time (s)

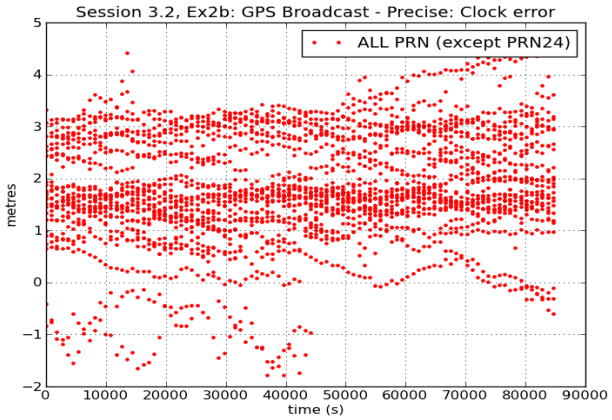

Session 3.2, Ex2a: GPS BRD[APC] vs IGS[APC]
Clock-Rad
metres
time (s)

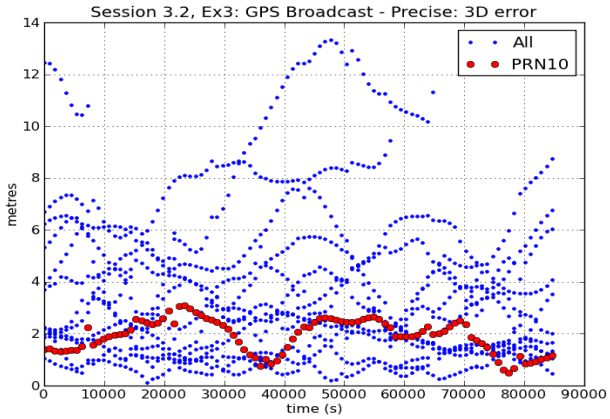

Session 3.2, Ex2a: GPS BRD[APC]-IGS[APC]+2.4
Clock
Radial (shifted +2.4m)
metres
time (s)

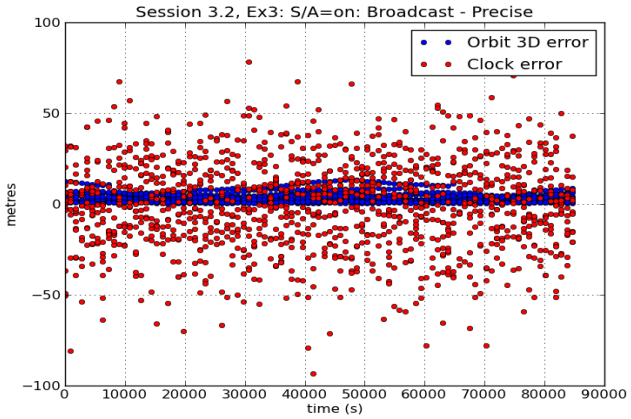

Session 3.2, Ex2b: GPS Broadcast - Precise: Radial error
ALL PRN
metres
time (s)

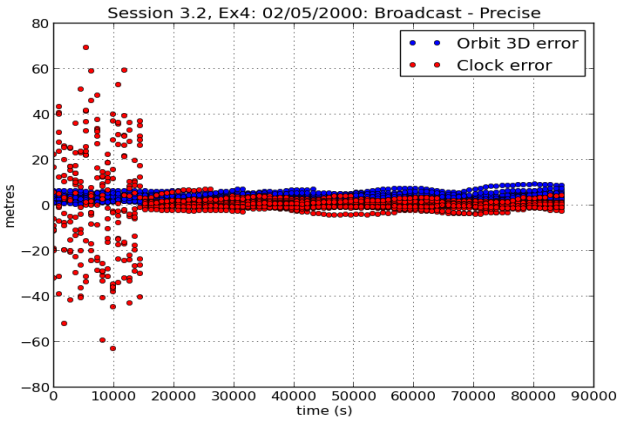

Session 3.2, Ex2b: GPS Broadcast - Precise: Along Track error
ALL PRN
metres
time (s)

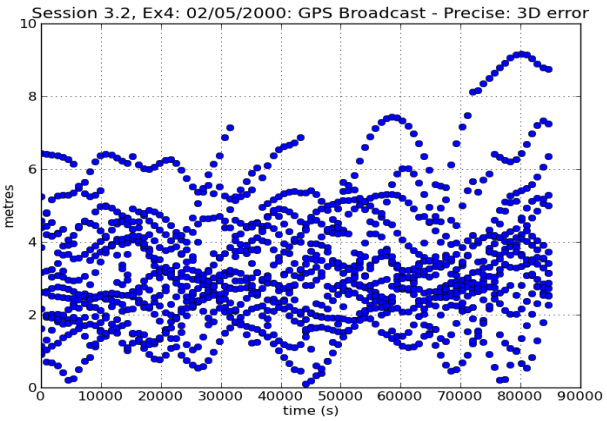

Session 3.2, Ex2b: GPS Broadcast - Precise: Cross Track error
ALL PRN
metres
time (s)

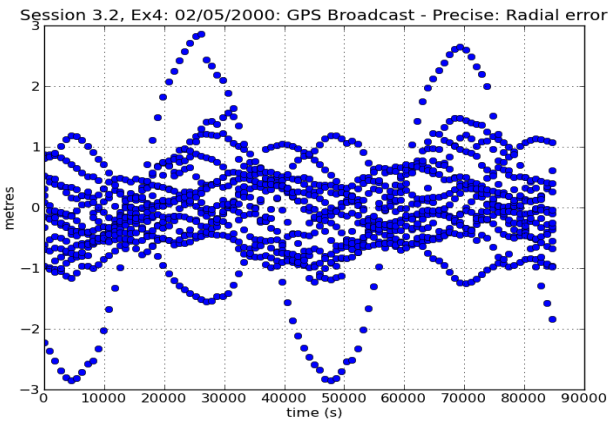

Session 3.2, Ex2b: GPS Broadcast - Precise: Clock error
ALL PRN (except PRN24)
metres
time (s)

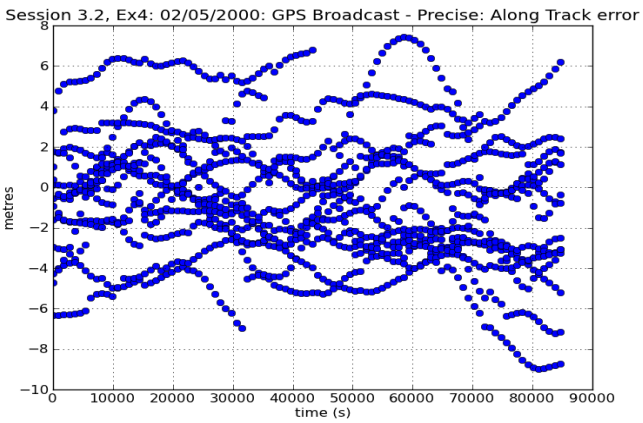

Session 3.2, Ex3: GPS Broadcast - Precise: 3D error
All
PRN10
metres
time (s)

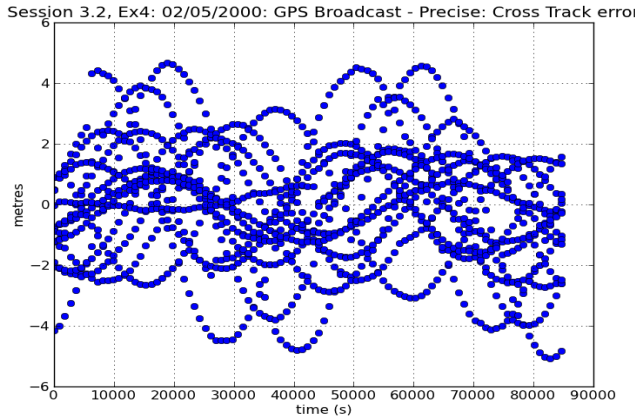

Session 3.2, Ex3: S/A=on: Broadcast - Precise
Orbit 3D error
Clock error
metres
time (s)

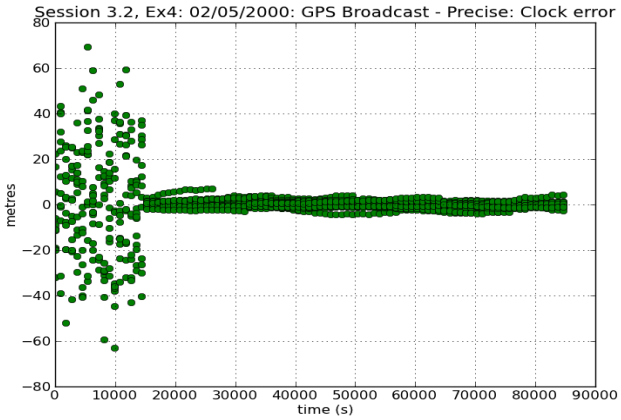

Session 3.2, Ex4: 02/05/2000: Broadcast - Precise
Orbit 3D error
Clock error
metres
time (s)

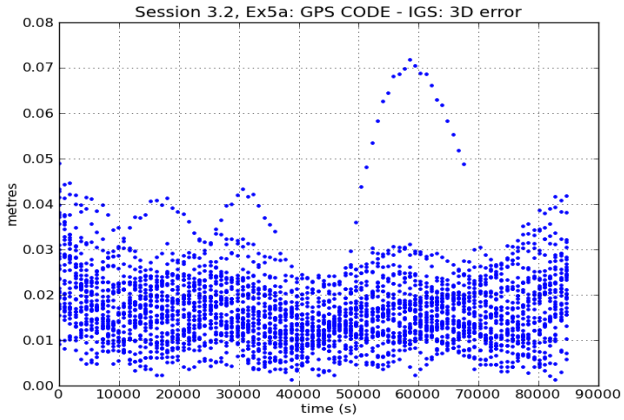

Session 3.2, Ex4: 02/05/2000: GPS Broadcast - Precise: 3D error
metres
time (s)

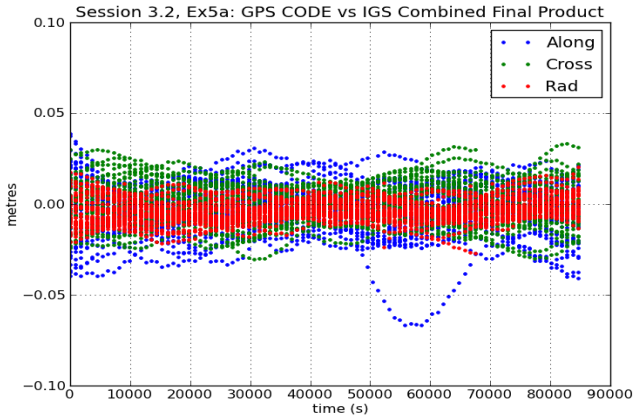

Session 3.2, Ex4: 02/05/2000: GPS Broadcast - Precise: Radial error
metres
time (s)

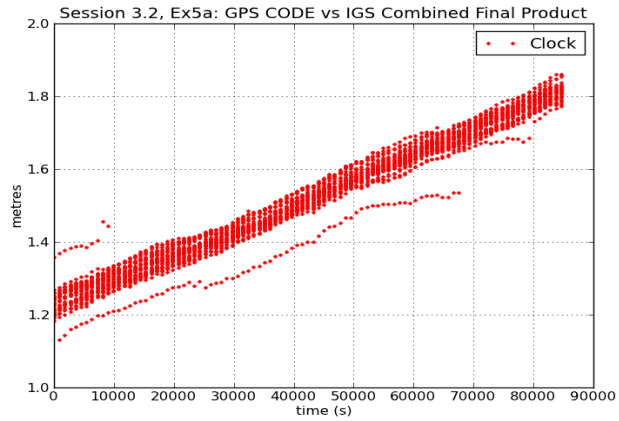

Session 3.2, Ex4: 02/05/2000: GPS Broadcast - Precise: Along Track error
metres
time (s)

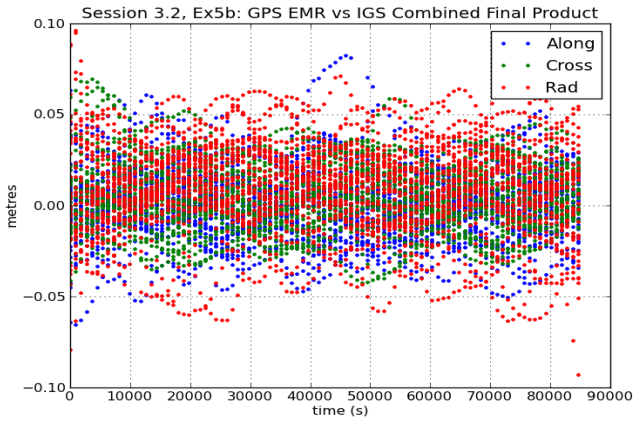

Session 3.2, Ex4: 02/05/2000: GPS Broadcast - Precise: Cross Track error
metres
time (s)

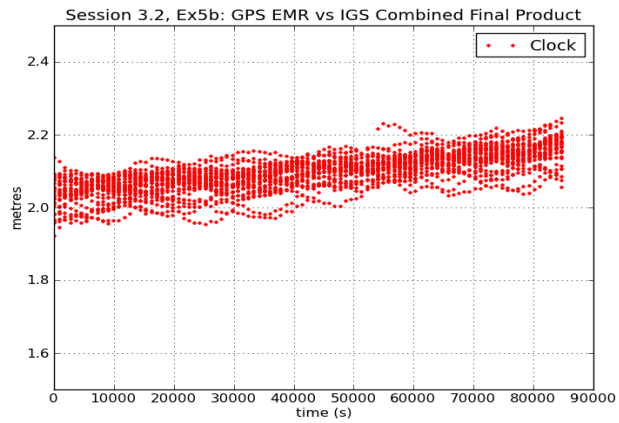

Session 3.2, Ex4: 02/05/2000: GPS Broadcast - Precise: Clock error
metres
time (s)

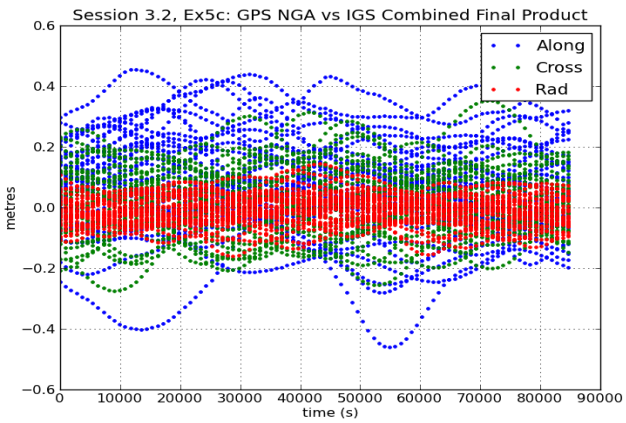

Session 3.2, Ex5a: GPS CODE - IGS: 3D error
metres
time (s)

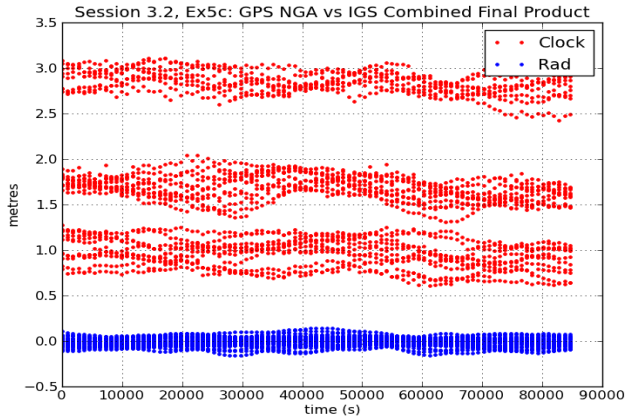

Session 3.2, Ex5a: GPS CODE vs IGS Combined Final Product
Along
Cross
Rad
metres
time (s)

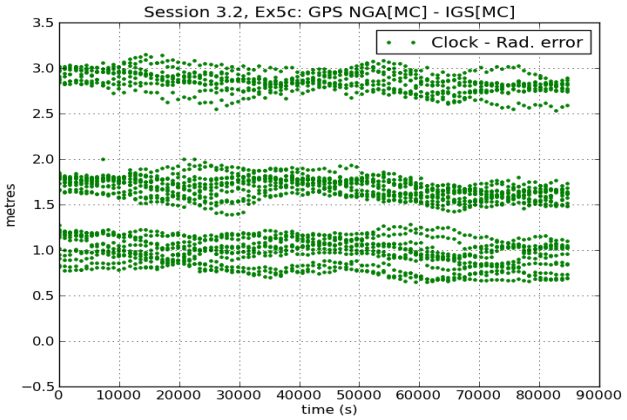

Session 3.2, Ex5a: GPS CODE vs IGS Combined Final Product
Clock
metres
time (s)

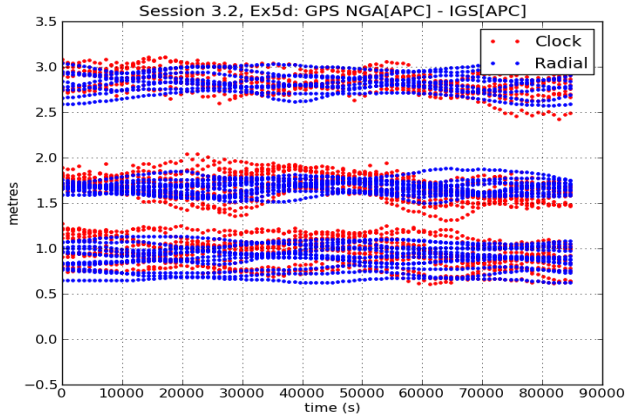

Session 3.2, Ex5b: GPS EMR vs IGS Combined Final Product
Along
Cross
Rad
metres
time (s)

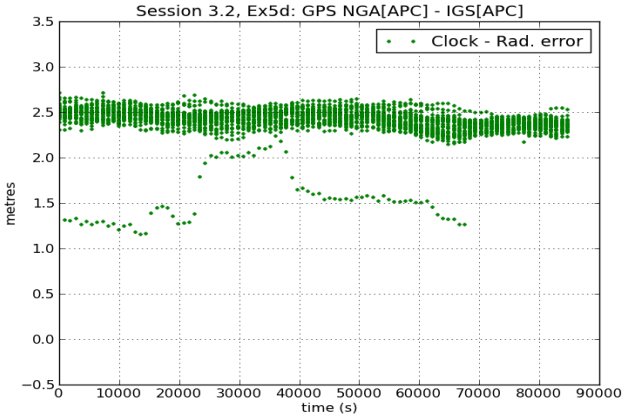

Session 3.2, Ex5b: GPS EMR vs IGS Combined Final Product
Clock
metres
time (s)

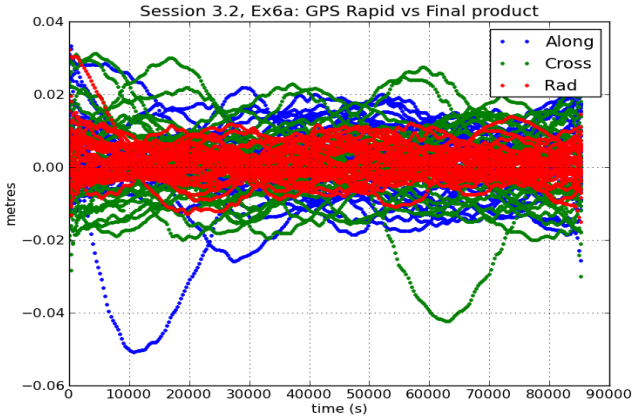

Session 3.2, Ex5c: GPS NGA vs IGS Combined Final Product
Along
Cross
Rad
metres
time (s)

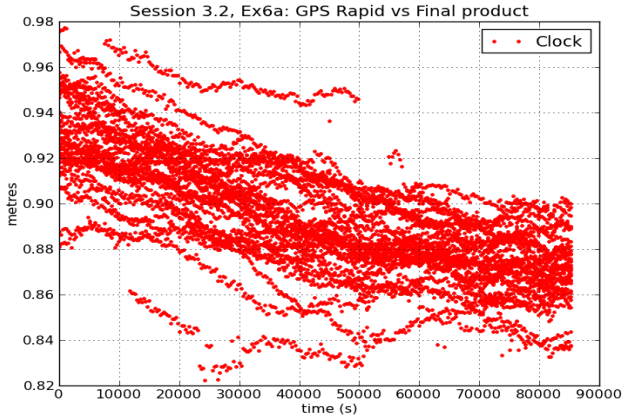

Session 3.2, Ex5c: GPS NGA vs IGS Combined Final Product
Clock
Rad
metres
time (s)

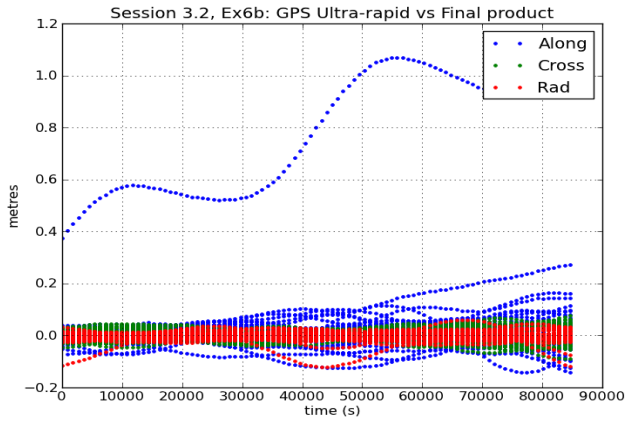

Session 3.2, Ex5c: GPS NGA[MC] - IGS[MC]
Clock - Rad. error
metres
time (s)

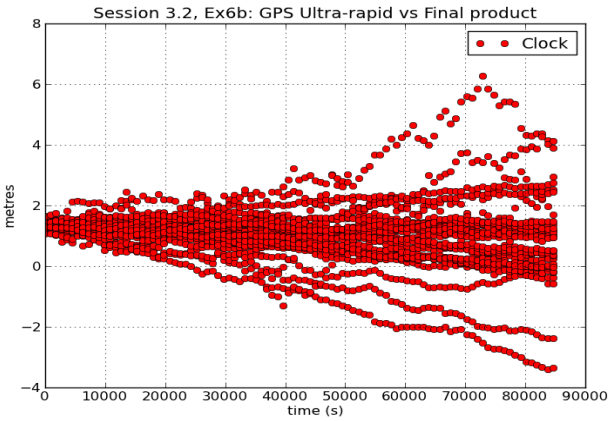

Session 3.2, Ex5d: GPS NGA[APC] - IGS[APC]
Clock
Radial
metres
time (s)

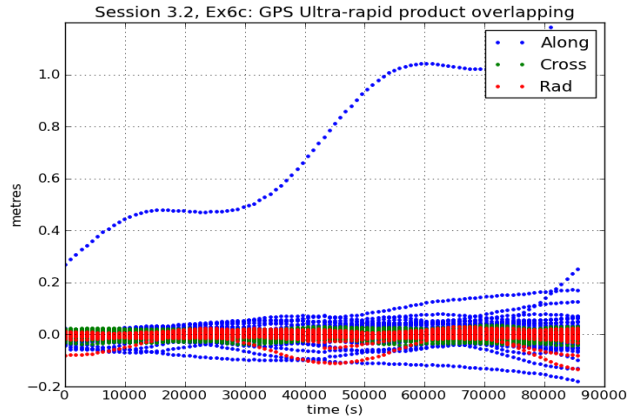

Session 3.2, Ex5d: GPS NGA[APC] - IGS[APC]
Clock - Rad. error
metres
time (s)

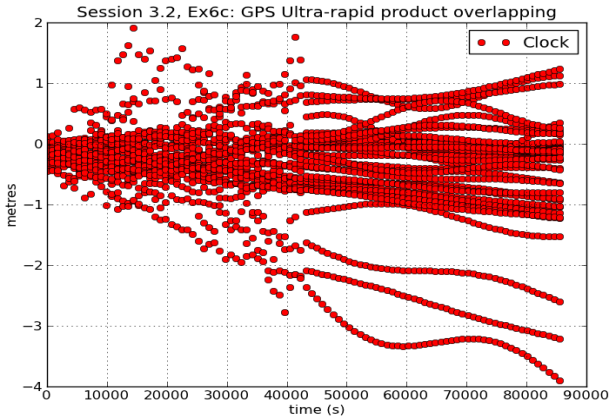

Session 3.2, Ex6a: GPS Rapid vs Final product
Along
Cross
Rad
metres
time (s)

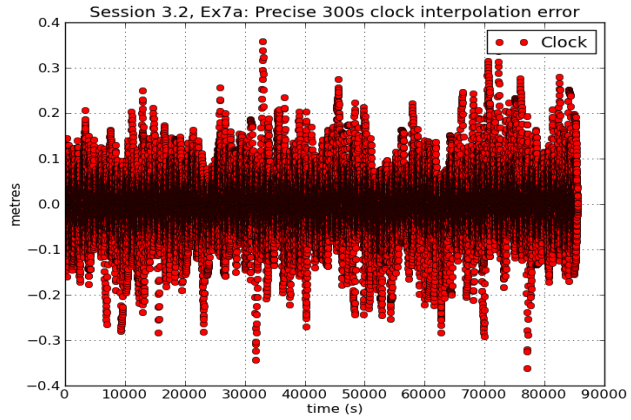

Session 3.2, Ex6a: GPS Rapid vs Final product
Clock
metres
time (s)

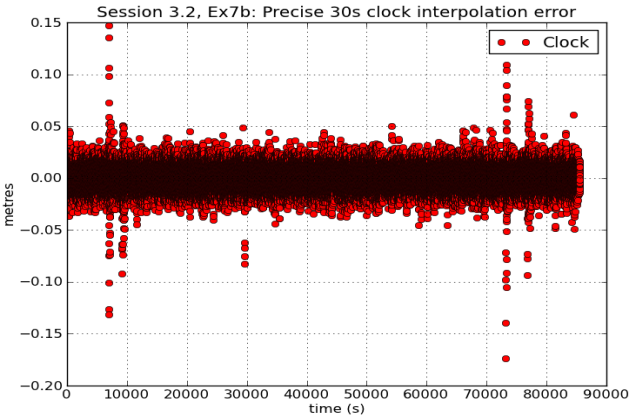

Session 3.2, Ex6b: GPS Ultra-rapid vs Final product
Along
Cross
Rad
metres
time (s)

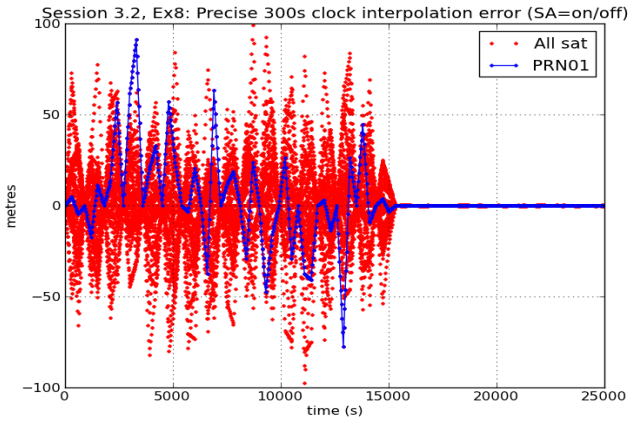

Session 3.2, Ex6b: GPS Ultra-rapid vs Final product
Clock
metres
time (s)

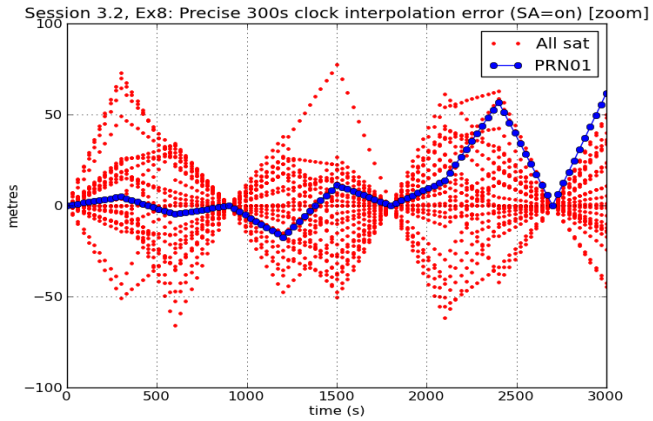

Session 3.2, Ex6c: GPS Ultra-rapid product overlapping
Along
Cross
Rad
metres
time (s)


Session 3.2, Ex6c: GPS Ultra-rapid product overlapping
Clock
metres
time (s)


Session 3.2, Ex7a: Precise 300s clock interpolation error
Clock
metres
time (s)


Session 3.2, Ex7b: Precise 30s clock interpolation error
Clock
metres
time (s)


Session 3.2, Ex8: Precise 300s clock interpolation error (SA=on/off)
All sat
PRN01
metres
time (s)


Session 3.2, Ex8: Precise 300s clock interpolation error (SA=on) [zoom]
All sat
PRN01
metres
time (s)

Session 3.3. Glonass Broadcast Orbit Integration

Objectives

To analyse the performance of Glonass orbits integrated from broadcast navigation data and its short- and long-term accuracy. To compare broadcast and precise coordinates taking into account the reference system involved (PZ-90, PZ-90.02, ITRF2000). To visualise the effect of different perturbations on orbit integration.

Files to use
`iac15454.sp3`, `igl15454.sp3`, `mcc15454.sp3`, `iac14454.sp3`, `brdc2320.09g`, `brdc2630.07g`, `ITRF00.ITRF05`, `ITRF00.itr9`, `ITRF00.pz90`, `GLOeph2sp3.nml`

Programs to use
`gLAB_linux`, `graph.py`, `GLOeph2sp3.f`, `trnfsp3n.f`

Preliminary

The program `GLOeph2sp3.f` implements [GLONASS ICD, 1998] to compute Glonass satellite coordinates from the broadcast navigation message (see Volume I section 3.3.2.1). This navigation data must be given in RINEX2.0 format in file `GLO_EPH.dat`, which contains satellite initial conditions $(\mathbf{r}_{t_0}, \mathbf{v}_{t_0})$ and Sun+Moon accelerations in the TRF (PZ-90/90.02), in 30 min time intervals in Glonass time.[30]

The program is executed, for example, as follows:

```
cp brdc2320.09g GLO_EPH.dat
GLOeph2sp3 > glo15454.txt
```

The different options to manage the program are set by the configuration namelist file `GLOeph2sp3.nml`, having as the default parameters:

```
$parameters
  tsys="GLO"
  wmode="SP3
  wptv0="NO"
  idt_out=30
  dt=30.d0
  niter=60
  useC20="YES"
  useSM="YES"
  use_brdSM="YES"
$end
```

[30] Actually, the RINEX files use UTC[SU], not Moscow time, as in the Glonass ICD.

The previous namelist configures the program `GLOeph2sp3.f` to integrate the orbit in the interval $[t_0 : t_0 + 60 \times 30$ seconds] for each satellite having the initial condition $(\mathbf{r}_{t_0}, \mathbf{v}_{t_0})$ in file `GLO_EPH.dat`. That is, 60 iterations (`niter=60`) with a time step of 30 s (`dt=30.d0`), which covers the 1800 s (i.e. 1/2 h) interval between two consecutive sets of broadcast orbits. The results are written with a 30 s sampling rate (`idt_out=30`).[31]

The parameter `tsys` defines the system time used in the output solutions. The options are `[tsys="GLO"]`[32] or `[tsys="GPS"]`. The option `wmode` defines the format of the output file. The option `[wmode="SP3"]` writes the file in the standard SP3 format (see session 2.2). With `[wmode="TXT"]` the output is in text mode format containing the following fields:

```
[SV,YYY,MM,DD,hh,mm,ss.ss,x,y,z,vx,vy,vz,clock]
```

where the coordinates are given in m and velocities in m/s in the TRF systems PZ-90 or PZ-90.02.[33] The option `wmode="OUT"` outputs the integration results including some internal computational values as the accelerations, according to the format:

`[SV,YYY,MM,DD,hh,mm,ss.ss,x,y,z,vx,vy,vz,clk_`τ`,clk_`γ`,`a_{Total}`,` $a_{Central}$`,`a_{J2}`,`$a_{Sun+Moon}$`]`

where the position, velocity and acceleration are in m, m/s and m/s^2, respectively.

The option `[wptv0="YES/NO"]` is to enable/disable the writing of the initial conditions given in the RINEX file. This parameter is set to `"NO"` by default.

The options `[useC20="YES/NO"]` and `[useSM="YES/NO"]` allow the user to enable/disable use of Earth's potential J2 (i.e. C20) term and the Sun+Moon accelerations (see Volume I section 3.3.2.1). They are activated by default (i.e., set to `"YES"`).

Finally, the option `[use_brdSM="YES/NO"]` sets the program to use the Sun+Moon acceleration from the broadcast file (`use_brdSM="YES"`) or compute such values itself (`use_brdSM="NO"`). This option is set to `"YES"`, by default.

[31] More precisely, the output will be written modulo 30 seconds, that is $t\%(30 seconds)$. Note that the orbit is integrated at times $t_0 + k \times$ `dt` with $k = 1, \cdots,$ `niter`, and the results are written when $[t_0 + k \times$ `dt` $]\%($`idt_out`$) = 0$. That is, only the computed solutions matching both times are written. This means, for instance, that if $t_0 = 10$ s, with `dt=15.d0` and `idt_out=20` only the solutions at times $t = 40, 100, \ldots$ will be written.
Comment: With `[dt=1.d0]` the output at any `idt_out` is assured. Nevertheless, a 1 s integration step is not actually required from the point of view of accuracy. Indeed, as shown in the exercise 3 next, from the integration error point of view an integration step of 30 s performs just as well as one of 1 s.

[32] As mentioned above, Glonass time is taken from RINEX files. That is, UTC[SU], not Moscow time, as in the Glonass ICD.

[33] From 20 September 2007, the Glonass broadcast ephemerides are given in PZ-90.02. Before this date, they were given in the PZ-90 reference system.

Development

Session files.

Copy and uncompress these session files in the working directory:

```
cp ~/GNSS/PROG/SES33/* .
cp ~/GNSS/FILES/SES33/* .
gzip -d *.gz *.Z
```

1. Glonass broadcast orbit integration: short-term accuracy

The accuracy of Glonass satellite coordinates computed from the broadcast navigation message is assessed in this exercise. This is done by comparing the satellite coordinates at the reference epoch of one ephemeris data set with the coordinates computed from the data set of a previous reference epoch. That is, after 30 minutes of orbit integration.

The following procedure can be applied:

(a) From the broadcast navigation file, compute the integrated orbit. Write the results in SP3 file format.

Set `GLOeph2sp3.nml` as follows to generate SP3 files with coordinates[34] written at a 30 s sampling rate:

```
$parameters
 tsys="GLO"
 wmode="SP3"
 wptv0="NO"
 idt_out=30
 dt=30.d0
 niter=60
$end
```

Execute:

```
cp brdc2320.09g GLO_EPH.dat
GLOeph2sp3 > glo15454.sp3
```

Note: The parameter `wptv0` *is set to* `"NO"` *to disable the writing of the initial conditions. That is, only the outputs from the integrated orbits will be written.*

[34] According to [GLONASS ICD, 1998], these coordinates are relative to the satellite's APC.

(b) Generate an SP3 file with the initial conditions at the reference epochs. Set `GLOeph2sp3.nml` as follows:

```
$parameters
 tsys="GLO"
 wmode="SP3"
 wptv0="YES"
 idt_out=900
 dt=1.d0
 niter=0
$end
```

Note: No iterations are done (i.e. `niter=0`*) and therefore only the initial conditions are written (notice that* `wptv0="YES"`*).*

Execute:

```
GLOeph2sp3 > glo15454.sp3_ini
```

(c) Use `gLAB` to compare both files (disable the APC[35] correction).

Complete the following steps:

i. Compute differences:

```
gLAB_linux -input:SP3 glo15454.sp3
           -input:SP3 glo15454.sp3_ini
                --model:satphasecenter > dif.all

grep SATDIFF dif.all > dif.sel
```

ii. Plot the results:

```
graph.py -f dif.sel -x4 -y11 -l Radial
         -f dif.sel -x4 -y12 -l Along
         -f dif.sel -x4 -y13 -l Cross

graph.py -f dif.sel -x4 -y9 -l "3D Error"
```

(d) To assess the evolution of the error, compare both files using `gLAB`, but interpolate the coordinates at the reference epoch with a 10-degree polynomial.[36]

[35] Note that, when comparing two SP3 files, `gLAB` applies the APC correction only to the second file. Disabling the APC correction (i.e. setting option `--model:satphasecenter`) relates the coordinates computed from both files to the same reference (i.e. the MC).

[36] By default, the SP3 file coordinates are interpolated by a 10-degree polynomial when using the options `[-pre:dec 30]` and `[ -model:clock:deg 1]` in `gLAB`. Nevertheless, different polynomial degrees for the coordinate interpolation can be set by the condition '`-model:orbit:deg`'.

Complete the following steps:

i. Compute differences:

```
gLAB_linux -input:SP3 glo15454.sp3
           -input:SP3 glo15454.sp3_ini
           -input:ant igs05_1545.atx
           --model:satphasecenter
           -pre:dec 30 -model:clock:deg 1 > dif.all
```

ii. Select rows starting with the `SATDIFF` label and excluding the first and last two hours, to avoid the Runge effect (i.e. oscillation at the edges of the interval due to polynomial interpolation).

```
grep SATDIFF dif.all| gawk '{if ($4 > 7200 &&
                          $4 < 79200) print $0}' > dif.sel
```

iii. Plot the results:

```
graph.py -f dif.sel -x4 -y11 -l Radial
         -f dif.sel -x4 -y12 -l Along
         -f dif.sel -x4 -y13 -l Cross

graph.py -f dif.sel -x4 -y9 -l "3D Error"
```

Repeat the previous plots but merge all the solutions in the 30 min integration window (i.e. from $[t_0\colon\ t_0 + 1800]$, with $t_0 = 900, 2\,700\ldots$ seconds).

Execute for instance:

```
graph.py -f dif.sel -x'(($4-900)%1800)' -y11 -l Radial
graph.py -f dif.sel -x'(($4-900)%1800)' -y12 -l Along
graph.py -f dif.sel -x'(($4-900)%1800)' -y13 -l Cross
graph.py -f dif.sel -x'(($4-900)%1800)' -y9 -l "3D Error"
```

2. Glonass broadcast orbit integration: long-term accuracy

The long-term accuracy of the Glonass integrated orbit is assessed in this exercise, by selecting the broadcast ephemerides at a given reference epoch and integrating them forward by several hours (6 h in this example). The following procedure can be applied:

(a) From the RINEX broadcast message file `brdc2320.09g` select the ephemerides at the reference epoch 2009 08 20 0 h 15 min 0.0 s.

This can be done by editing the file and removing all data blocks after such a time, as in exercise 8 of session 3.1, or by executing:

```
grep -B10 "END OF HEADER" brdc2320.09g >  GLO_EPH.dat
grep -A3 "09  8 20  0 15" brdc2320.09g >> GLO_EPH.dat
```

(b) Using this file for the initial conditions, integrate the orbit with time steeps of `dt=300.d0` seconds over `niter=72` iterations (i.e. in the interval $[t_0 : t_0 + 6\,\mathrm{h}]$, with $t_0 = 15$ minutes).[37]

This computation is done by the `GLOeph2sp3.nml` namelist as:

```
$parameters
 tsys="GLO"
 wmode="SP3"
 wptv0="NO"
 idt_out=300
 dt=300.d0
 niter=72
$end
```

and executing:

```
GLOeph2sp3 > glo15454.sp3_6h
```

(c) Compare the computed coordinates with the satellite coordinates at the reference epochs of file `glo15454.sp3_ini` generated in the previous exercise.

Complete the following steps:

i. Compute differences (the APC correction is disabled):

```
gLAB_linux -input:SP3 glo15454.sp3_6h
           -input:SP3 glo15454.sp3_ini
              --model:satphasecenter > dif.all

grep SATDIFF dif.all > dif.sel
```

ii. Plot the results:

```
graph.py -f dif.sel -x4 -y11 -l Radial
         -f dif.sel -x4 -y12 -l Along
         -f dif.sel -x4 -y13 -l Cross

graph.py -f dif.sel -x4 -y9 -l "3D Error"
```

3. Glonass broadcast orbit integration: time step width effect

The effect of the time step width on the orbit integration, using the algorithm defined in the Glonass ICD, is assessed in this exercise.

Using program `GLOeph2sp3.f` and the RINEX broadcast ephemeris file `brdc2320.09g`, integrate the Glonass satellite orbit from a given reference epoch with different integration time step widths. Compare the results of the integrated orbit against the ephemeris at the next reference epoch, to assess the integration error.

Consider the following time steps `dt`: 1, 60, 120, 300, 900 s. Update the number of iterations (i.e. the `niter` parameter in the namelist) to cover the 1800 s interval between two consecutive reference epochs.

[37] The ephemeris reference epoch is 2009 08 20 0 h 15 min 0.0 s.

The following procedure can be applied:

(a) `[dt= 1.d0]`: Integrate the orbit using the namelist `GLOeph2sp3.nml` given as follows, where the number of iterations is set to `niter=1800` to reach the 1800 s interval between the consecutive reference epochs of broadcast ephemeris file `brdc2320.09g`:

```
$parameters
 tsys="GLO"
 wmode="SP3"
 wptv0="NO"
 idt_out=30
 dt=1.d0
 niter=1800
$end
```

Execute:

```
cp brdc2320.09g GLO_EPH.dat
GLOeph2sp3 > glo15454.sp3_1s
```

As in the previous exercise, compare the results with those of file `glo15454.sp3_ini` containing the coordinates at the reference epochs, generated in the previous exercise.

Complete the following steps:

i. Compute differences (the APC correction is disabled):

```
gLAB_linux -input:SP3 glo15454.sp3_1s
             -input:SP3 glo15454.sp3_ini
               --model:satphasecenter > dif.all_1

grep SATDIFF dif.all_1 > dif.sel_1
```

ii. Plot the results:

```
graph.py -f dif.sel_1 -x4 -y11 -l Radial
         -f dif.sel_1 -x4 -y12 -l Along
         -f dif.sel_1 -x4 -y13 -l Cross

graph.py -f dif.sel_1 -x4 -y9 -l "3D Error"
```

(b) Repeat the same procedure with `[dt= 60.d0, niter=30]`, `[dt= 120.d0, niter=15]`, `[dt= 300.d0, niter=6]`, `[dt= 600.d0, niter=3]`, `[dt= 900.d0, niter=2]`.

Name the files as `dif.sel_1`, `dif.sel_60`, ..., `dif.sel_900`.

(c) Show the previous results in a common plot:

```
graph.py -f dif.sel_900 -x4 -y9 -l 900s
         -f dif.sel_600 -x4 -y9 -l 600s
         -f dif.sel_300 -x4 -y9 -l 300s
         -f dif.sel_120 -x4 -y9 -l 120s
         -f dif.sel_60 -x4 -y9 -l 60s
         -f dif.sel_1 -x4 -y9 -l 1s -t "3D Error"
```

(d) Discuss the results. Is there any significant difference between 1 and 300 seconds? And with 600 or 900 seconds?

4. Glonass broadcast orbit: J2, Sun+Moon perturbation effect

The effect of Earth's oblateness (J2 term in Earth's potential) and the Sun+Moon attraction on the coordinates computed from the Glonass broadcast orbit is assessed in this exercise.

The following procedure can be applied: (1) integrate the orbit according to the Glonass ICD; (2) repeat the integration, but without applying the Sun+Moon acceleration term (i.e. setting `[useSM="NO"]`); (3) repeat the integration, but without applying either the Sun+Moon acceleration term or the J2 term (i.e. setting `[useSM="NO"]` and `[useC20="NO"]`). Then, applying the same procedure as in previous exercises, compare the computed orbits with the ephemerides at the reference epochs.[38]

(a) *Full Glonass ICD model*:
Set `GLOeph2sp3.nml` as:

```
$parameters
 tsys="GLO"
 wmode="SP3"
 wptv0="NO"
 idt_out=30
 dt=30.d0
 niter=60
 useC20="YES"
 useSM="YES"
 use_brdSM="YES"
$end
```

Complete the following steps:

i. Orbit integration:

Set `useC20="YES"` and `useSM="YES"` in `GLOeph2sp3.nml`.

Execute:

```
cp brdc2320.09g GLO_EPH.dat
GLOeph2sp3 > glo15454.sp3_11
```

ii. Compute differences (interpolating at the reference epochs):

```
gLAB_linux -input:SP3 glo15454.sp3_11
           -input:SP3 glo15454.sp3_ini
           --model:satphasecenter
           -pre:dec 30 -model:clock:deg 1 >dif.all_11
```

[38] The file `glo15454.sp3_ini` generated in exercise (1b) can be used, as well.

iii. Select rows starting with the `SATDIFF` label and exclude the first and last two hours, to avoid the Runge effect.

```
grep SATDIFF dif.all_11| gawk '{if ($4 > 7200 &&
                        $4 < 79200) print $0}' >dif.sel_11
```

iv. Plot the results:

```
graph.py -f dif.sel_11 -x4 -y11 -l Radial
         -f dif.sel_11 -x4 -y12 -l Along
         -f dif.sel_11 -x4 -y13 -l Cross
graph.py -f dif.sel_11 -x4 -y9 -l "3D Error"
```

(b) Repeat the previous scheme but disable the perturbation terms:

i. Set `useC20="YES"` and `useSM="NO"` in `GLOeph2sp3.nml`. Rename the results as `dif.sel_10`.

ii. Set `useC20="NO"` and `useSM="NO"` in `GLOeph2sp3.nml`. Rename the results `dif.sel_00`.

(c) Compare the results from the following plots:

1. Comparison results: **J2 effect**

Radial error component:

```
graph.py -f dif.sel_10 -x'(($4-900)%1800)' -y11 -l 10
   -f dif.sel_11 -x'(($4-900)%1800)' -y11 -l 00 -t Radial
```

Along-track error component:

```
graph.py -f dif.sel_10 -x'(($4-900)%1800)' -y12 -l 10
  -f dif.sel_11 -x'(($4-900)%1800)' -y12 -l 00 -t Alon
```

Cross-track error component:

```
graph.py -f dif.sel_10 -x'(($4-900)%1800)' -y12 -l 10
  -f dif.sel_11 -x'(($4-900)%1800)' -y12 -l 00 -t Cross
```

2. Comparison results: **J2 plus Sun+Moon accelerations effect**

Repeat the previous plots with files `dif.sel_00` and `dif.sel_11` to compare results without using J2 and Sun+Moon accelerations versus the full model (of the Glonass ICD).

Readers aiming to go further into the topic of this exercise can find complementary background in, for instance, [Montenbruck and Eberhard, 2005] or [Seeber, 1993].

5. Glonass broadcast versus precise orbits

The Glonass coordinates and clocks computed from the broadcast message are compared with the IGS precise determinations in this exercise. The RINEX navigation file `brdc2320.09g` and the IGS file `iac15454.sp3` with precise orbits and clocks will be used in this study.

The following procedure can be applied: Using program `GLOeph2sp3.f` generate an SP3 file with the broadcast integrated orbit. The GPS system time must be used to allow a direct comparison with the `iac15454.sp3` file. Afterwards, compare the broadcast and precise coordinates using `gLAB`.

Set `GLOeph2sp3.nml` as follows:[39]

```
$parameters
 tsys="GPS"
 wmode="SP3"
 idt_out=30
 dt=1.d0
 niter=1979
$end
```

Complete the following steps:

i. Broadcast orbit integration:

```
cp brdc2320.09g GLO_EPH.dat
GLOeph2sp3 > glo15454.sp3
```

ii. Compute differences (the APC correction is disabled):

```
gLAB_linux -input:SP3 glo15454.sp3
           -input:SP3 iac15454.sp3
           --model:satphasecenter > dif.all
```

iii. Select rows starting with the `SATDIFF` label:

```
grep SATDIFF dif.all > dif.sel
```

iv. Plot the results:

```
graph.py -f dif.sel -x4 -y11 -so -l Radial
         -f dif.sel -x4 -y12 -so -l Along
         -f dif.sel -x4 -y13 -so -l Cross

graph.py -f dif.sel -x4 -y9 -so -l "3D Error"

graph.py -f dif.sel -x4 -y10 --ymin -30 --ymax 30 -l Clk
```

[39] Note that, although 120 s would be enough for orbit integration (see exercise 3), an integration step of `dt=1.d0` seconds is used to avoid problems with the sampling time when writing the output. As commented above, the output is written at $t\%$(`idt_out`) seconds, where $t = t_0 + n \times$`dt`, with t_0 the reference epoch of the broadcast file. Thus, although it is not necessary from an accuracy point of view, a time step of `dt=1.d0` seconds ensures a solution is computed for any value of $t\%$(`idt_out`).

(a) What is the reference system used by Glonass in the broadcast orbits from 20 September 2007? What is the reference system used in the `iac15454.sp3` file? (See file header.)

(b) What is the level of discrepancy between reference system PZ-90.02 and ITRF2000? See Volume I, equation (3.9).

(c) What is the level of discrepancy between both determinations of the satellite coordinates?

(d) What is the level of discrepancy between the satellite clocks?

(e) Plot the radial component of the error as a function of the satellite:

```
graph.py -f dif.sel -x6 -y11 -so -l Rad
```

What would the bias be between the MC and the APC? (See Volume I, Table 5.5.)

6. From PZ-90 to PZ-90.02

The transition between reference frames PZ-90 and PZ-90.02 of Glonass is analysed in this exercise.

(a) Repeat the previous exercise but using the files `brdc2630.07g` and `iac14454.sp3`. The same namelist can be used.[40] Name the output file `glo14454.sp3`.

i. What are the reference systems used by Glonass in the broadcast orbits during this day? What reference system is used in the `iac14454.sp3` file? (See file header.)
Note: Add the options '`--model:satclocks`*' and '*`-pre:dec 30`*' when executing* `gLAB_linux` *to have the results at 30 s rate (i.e. to improve the plot resolution).*

ii. What is the level of discrepancy between the reference systems PZ-90, PZ-90.02 and ITRF2000? See Volume I, equations (3.8) and (3.9).

iii. Justify the jump in the satellite coordinates between 14 and 17 hours.

iv. What orbit error component is most affected?

v. Is there any jump in the satellite clocks?

(b) Using program `trnfsp3n.f`[41] and the parameter file `ITRF00.pz90`, transform the coordinates of the `glo14454.sp3` file from PZ-90 to the International Terrestrial Reference Frame (ITRF) reference system ITRF2000. Name the file obtained `glo14454.sp3_00`. Afterwards, compare the transformed coordinates with those of file `iac14454.sp3`.

Complete the following steps:

1. Coordinate transformation:

Transformation from PZ-90 to ITRF2000:

```
trnfsp3n glo14454.sp3 glo14454.sp3_00 ITRF00.pz90
```

[40] Use `dt=1.d0` to avoid problems with writing the output.
[41] This program is from *http://igscb.jpl.nasa.gov/igscb/resource/tutorial/APPENDIX_IGS_ITRF.txt*.

2. Coordinate comparison:

i. Compute differences (the APC correction is disabled):

```
gLAB_linux -input:SP3 glo14454.sp3_00
           -input:SP3 iac14454.sp3
           --model:satphasecenter
           --model:satclocks
           -pre:dec 30 > dif.all0
```

ii. Select rows starting with the `SATDIFF` label:

```
grep SATDIFF dif.all0 > dif.sel0
```

3. Plot the results:

```
graph.py -f dif.sel0 -x4 -y11 -l Radial
         -f dif.sel0 -x4 -y12 -l Along
         -f dif.sel0 -x4 -y13 -l Cross

graph.py -f dif.sel0 -x4 -y9 -l "3D Error"
```

i. Compare the parameters of file `ITRF00.pz90` with equation (3.8), Volume I.

ii. How did this transformation affect results?

iii. Which error components are mostly affected?

7. Comparison of Glonass precise orbit and clock products

The precise orbit and clock products computed by different centres are compared in this exercise using the files `iac15454.sp3`, `mcc15454.sp3` and `igl15454.sp3`.

(a) Compare the discrepancies between the orbit and clock estimates of the IGS Glonass orbit products (IGL) file `igl15454.sp3` and the Information Analytical Centre (IAC)[42] `iac15454.sp3` files.
Unset the APC correction by adding '`--model:satphasecenter`'.
Complete the following steps:

1. Computations:

i. Compute differences:

```
gLAB_linux -input:SP3 igl15454.sp3 -input:SP3
   iac15454.sp3 --model:satphasecenter > dif.all
```

ii. Select rows starting with the `SATDIFF` label:

```
grep SATDIFF dif.all > dif.sel
```

[42]Federal IAC website: *http://www.glonass-ianc.rsa.ru*.

2.- Plot the results:

Radial, along-track and cross-track errors:

```
graph.py -f dif.sel -x4 -y11 -l Radial
         -f dif.sel -x4 -y12 -l Along
         -f dif.sel -x4 -y13 -l Cross
```

Clock error:

```
graph.py -f dif.sel -x4 -y10 -l Clock
```

What is the order of magnitude of the discrepancies? Why does a large bias appear in the clock?

(b) Compare the discrepancies between the orbit estimates of the MCC file `mcc15454.sp3` and the IAC file `iac15454.sp3`.

Note: The MCC file does not contains clock estimates, so the option `--model:satclocks` *must be applied to unset the satellite clock comparison.*

Execute for instance:

1. Coordinate comparison:

i. Compute differences
(the APC correction is disabled):

```
gLAB_linux -input:SP3 mcc15454.sp3
           -input:SP3 iac15454.sp3
           --model:satphasecenter
           --model:satclocks > dif.all
```

ii. Select rows with the `SATDIFF` label:

```
grep SATDIFF dif.all > dif.sel
```

2. Plot the results:

```
graph.py -f dif.sel -x4 -y11 -l Radial
         -f dif.sel -x4 -y12 -l Along
         -f dif.sel -x4 -y13 -l Cross
```

(c) Check if the coordinates are given in the same reference frame. Transform the coordinates to a common frame if necessary.

i. Edit `mcc15454.sp3` and `iac15454.sp3` and check the reference system used for the coordinates in such files.

ii. Transform the coordinates of `mcc15454.sp3` from ITRF97 to ITRF2000 and again compare the coordinates with those present in file `iac15454.sp3`. Do the results improve?
Note that the file `mcc15454.sp3` contains Microsoft Disk Operating System (MSDOS) carriage returns that can produce a malfunction in program `trnfsp3n.f`. Such carriage returns can be removed as follows:

```
cat mcc15454.sp3 | sed 's/\r//g' > file.tmp
mv file.tmp mcc15454.sp3
```

Complete the following steps:

1. Coordinate transformation:[43]

Transformation from ITRF97 to ITRF2000:

```
trnfsp3n mcc15454.sp3 mcc15454.sp3_00 ITRF00.itr97
```

2. Coordinate comparison:

i. Compute differences:

```
gLAB_linux -input:SP3 mcc15454.sp3_00
             -input:SP3 iac15454.sp3
                -input:ant igs05_1545.atx
                  --model:satphasecenter
                    --model:satclocks > dif.all_00
```

ii. Select rows starting with the `SATDIFF` label:

```
grep  SATDIFF dif.all_00 > dif.sel_00
```

3. Plot the results:

Total error comparison:

```
graph.py -f dif.sel_00 -x4 -y9 -so -l "ITRF2000"
       -f dif.sel -x4 -y9 -so -l "ITRF97/ITRF2000"
                                    -t "3D Error"
```

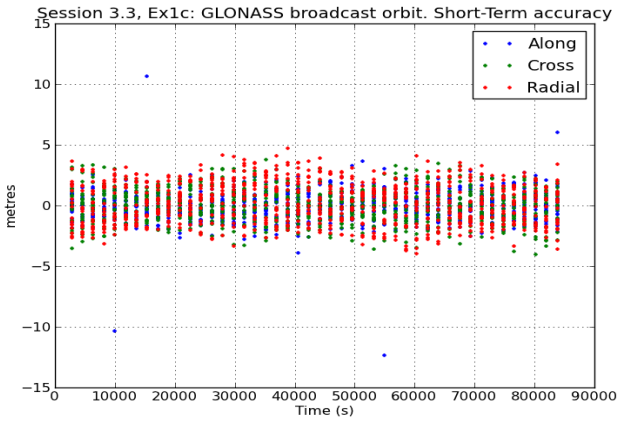
Radial, along track and cross track errors:

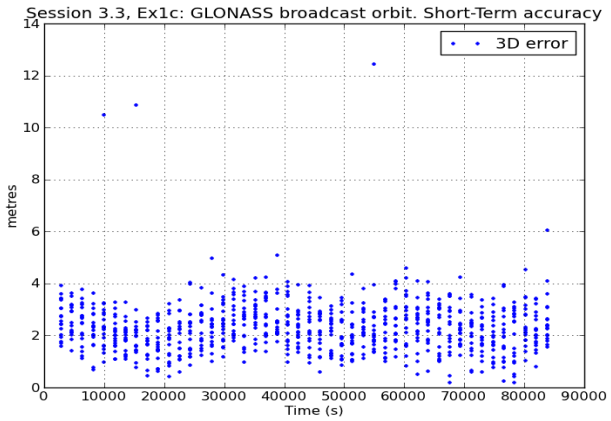
```
graph.py -f dif.sel_00 -x4 -y11 -l "ITRF2000"
          -f dif.sel -x4 -y11 -l "ITRF97/ITRF2000"
                                  -t Radial error

graph.py -f dif.sel_00 -x4 -y12 -l "ITRF2000"
          -f dif.sel -x4 -y12 -l "ITRF97/ITRF2000"
                               -t Along Track error

graph.py -f dif.sel_00 -x4 -y13 -l "ITRF2000"
          -f dif.sel -x4 -y13 -l "ITRF97/ITRF2000"
                               -t Cross Track error
```

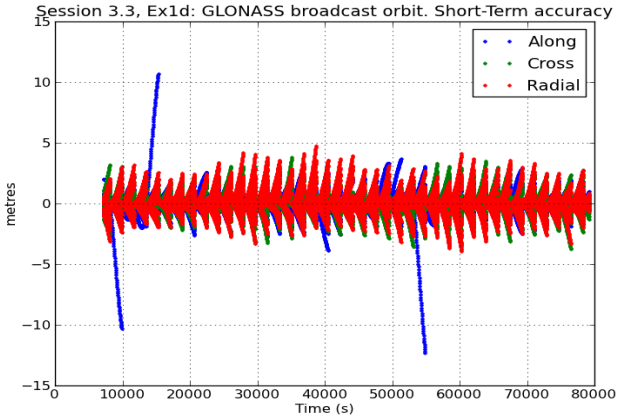
Do the results improve? What is the level of discrepancy?

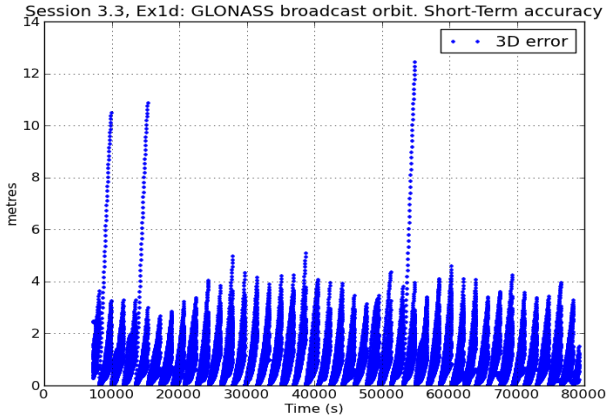
[43] The `ITRF00.itr97` file is from *ftp://macs.geod.nrcan.gc.ca/pub/requests/itrf96_97/*.

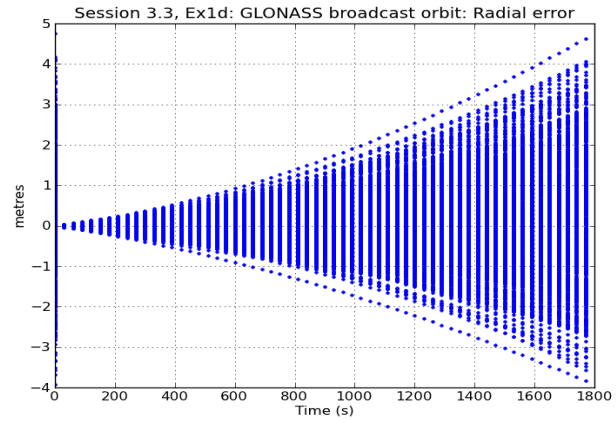
Graphs Session 3.3

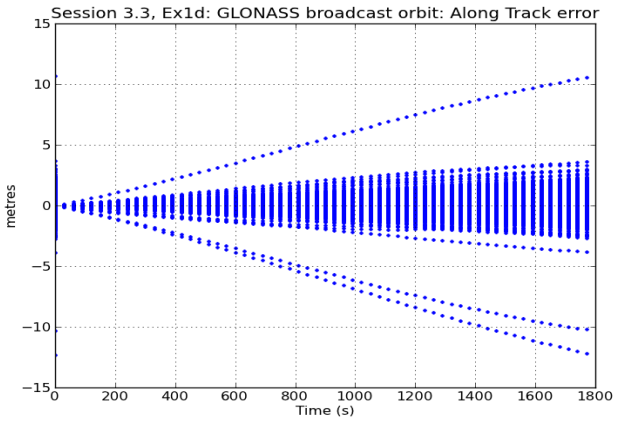


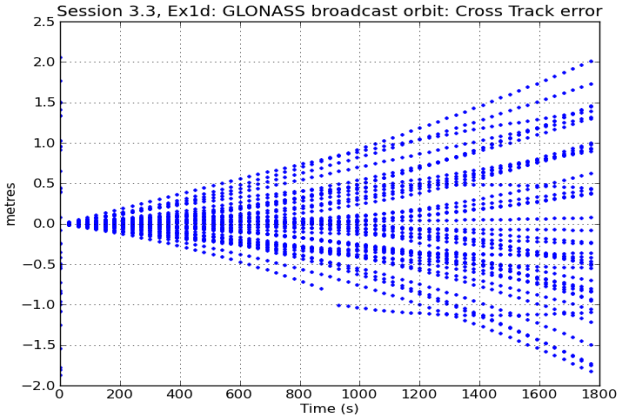


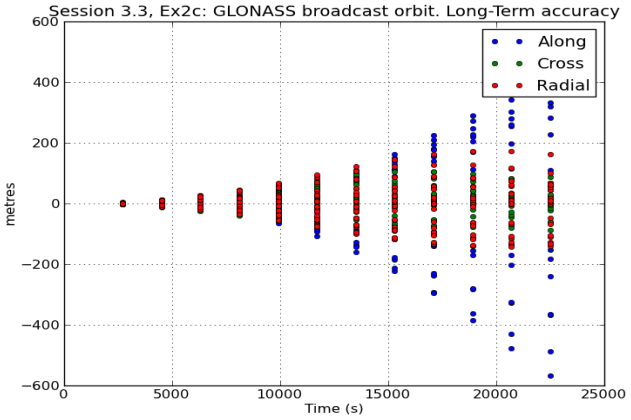


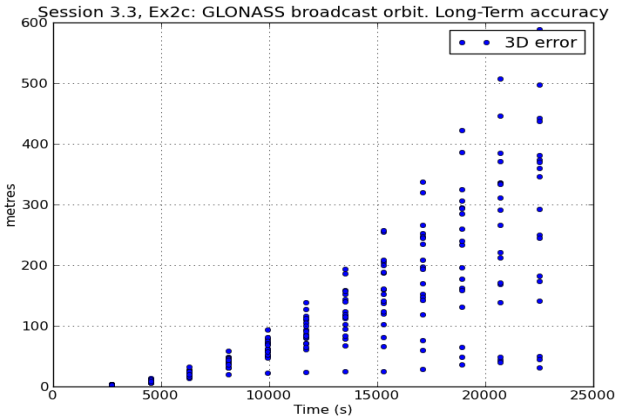


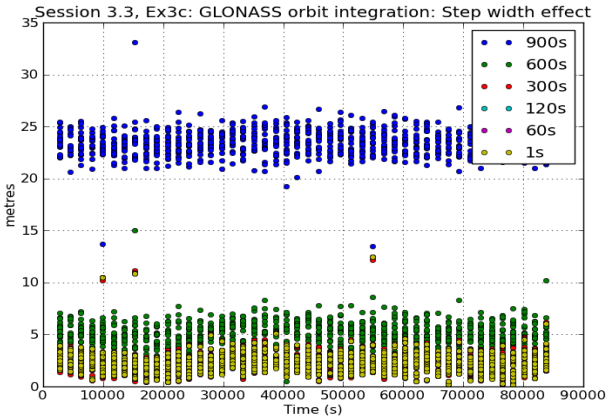


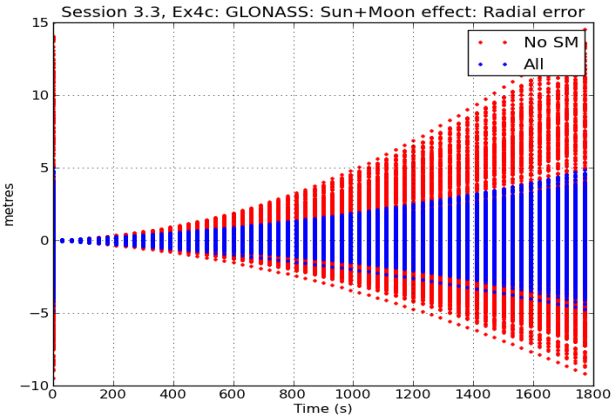


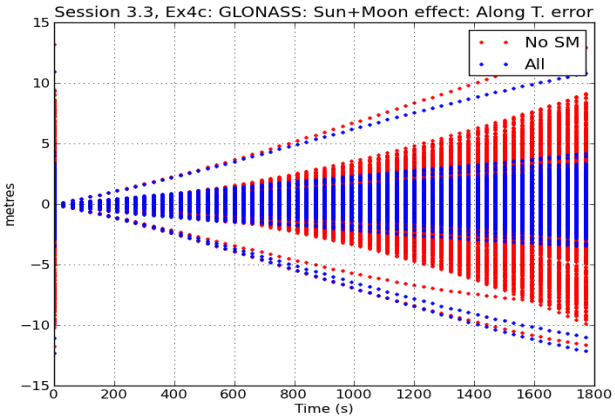


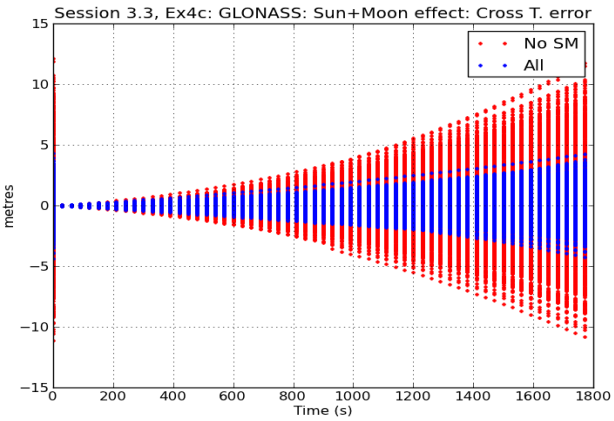


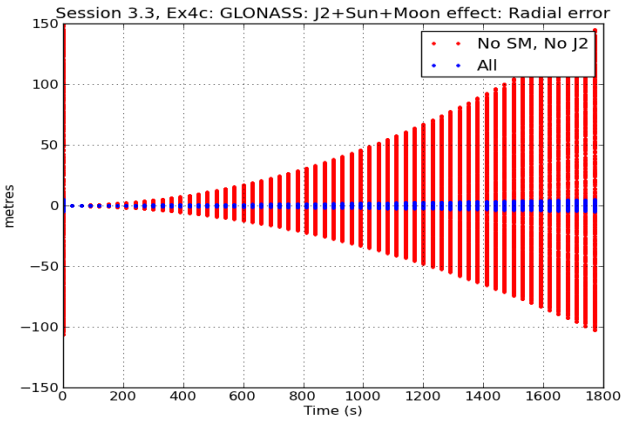

Session 3.3, Ex2c: GLONASS broadcast orbit. Long-Term accuracy
3D error
metres
Time (s)

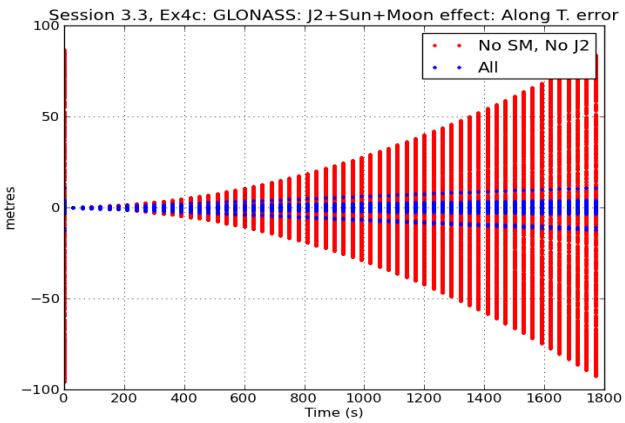

Session 3.3, Ex3c: GLONASS orbit integration: Step width effect
900s
600s
300s
120s
60s
1s
metres
Time (s)

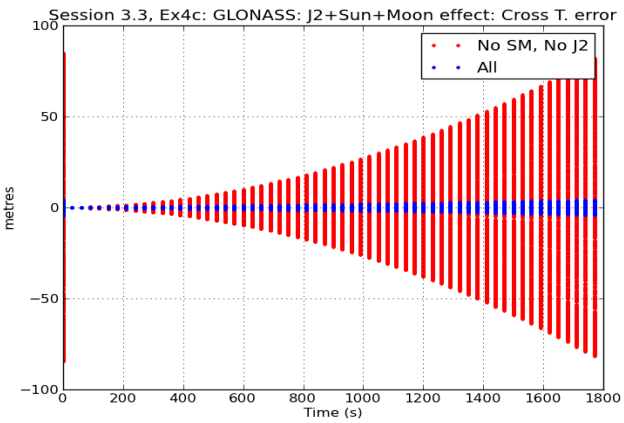

Session 3.3, Ex4c: GLONASS: Sun+Moon effect: Radial error
No SM
All
metres
Time (s)

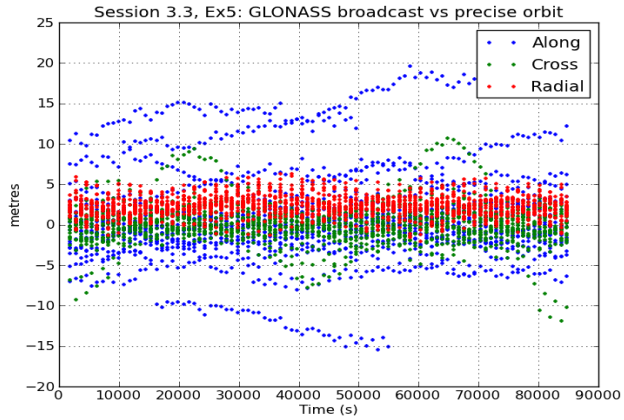

Session 3.3, Ex4c: GLONASS: Sun+Moon effect: Along T. error
No SM
All
metres
Time (s)

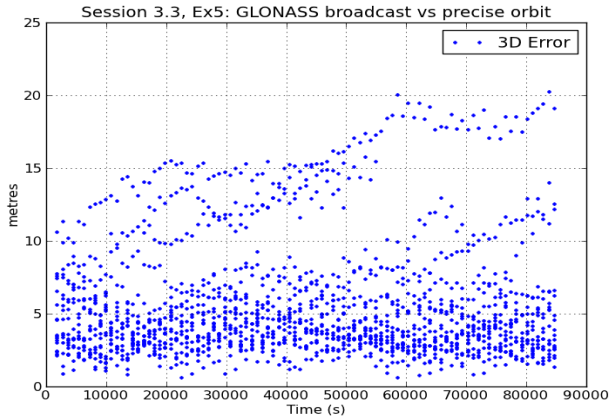

Session 3.3, Ex4c: GLONASS: Sun+Moon effect: Cross T. error
No SM
All
metres
Time (s)

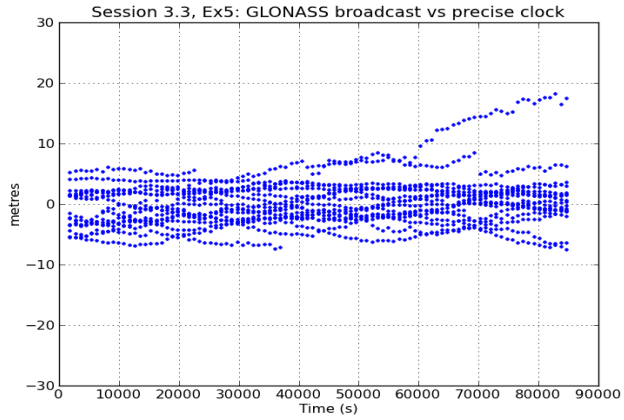

Session 3.3, Ex4c: GLONASS: J2+Sun+Moon effect: Radial error
No SM, No J2
All
metres
Time (s)

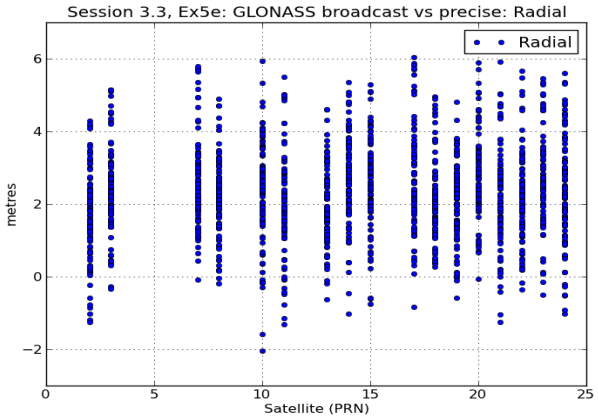

Session 3.3, Ex4c: GLONASS: J2+Sun+Moon effect: Along T. error
No SM, No J2
All
metres
Time (s)

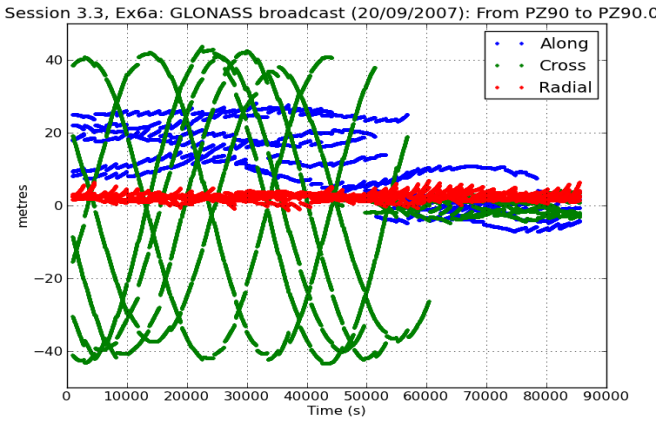

Session 3.3, Ex4c: GLONASS: J2+Sun+Moon effect: Cross T. error
No SM, No J2
All
metres
Time (s)

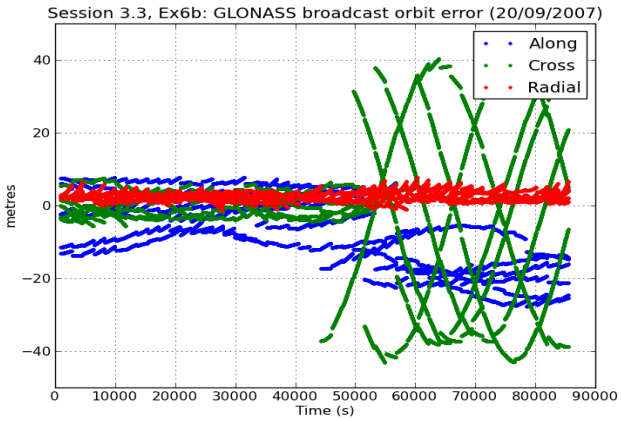

Session 3.3, Ex5: GLONASS broadcast vs precise orbit
Along
Cross
Radial
metres
Time (s)

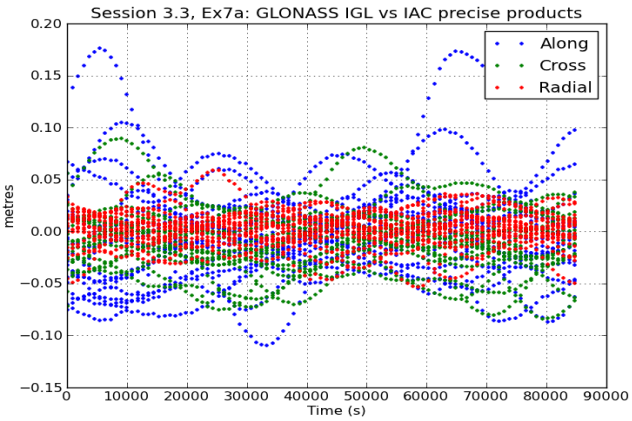

Session 3.3, Ex5: GLONASS broadcast vs precise orbit
3D Error
metres
Time (s)

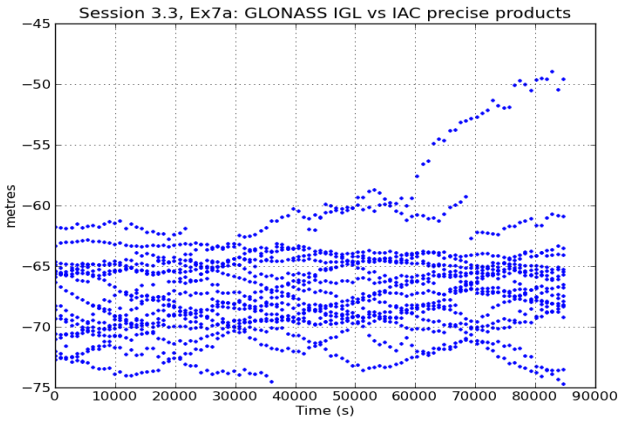

Session 3.3, Ex5: GLONASS broadcast vs precise clock
metres
Time (s)

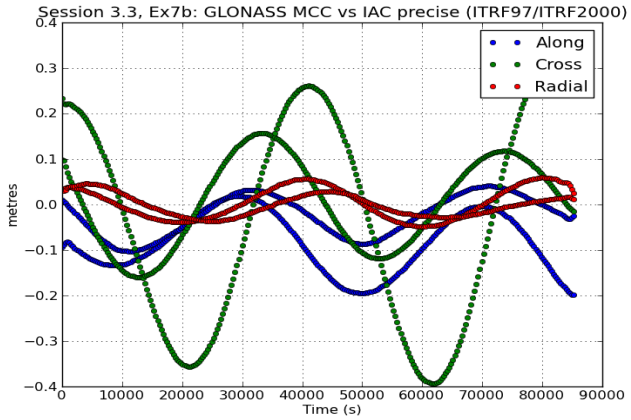

Session 3.3, Ex5e: GLONASS broadcast vs precise: Radial
Radial
metres
Satellite (PRN)

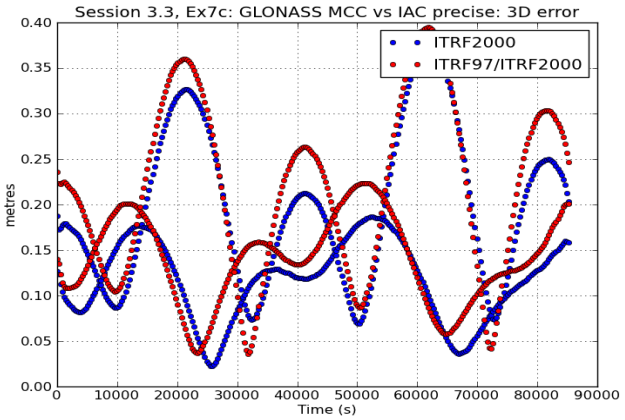

Session 3.3, Ex6a: GLONASS broadcast (20/09/2007): From PZ90 to PZ90.0
Along
Cross
Radial
metres
Time (s)

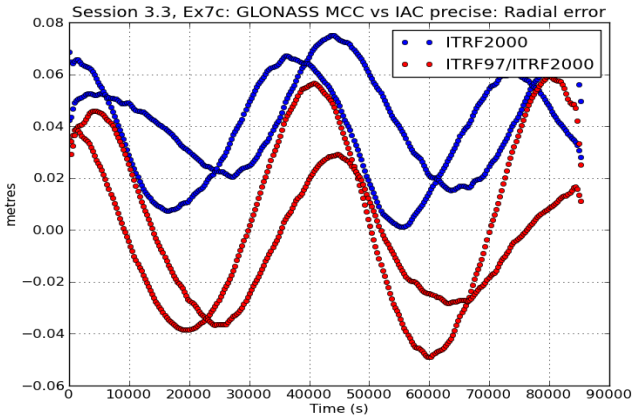

Session 3.3, Ex6b: GLONASS broadcast orbit error (20/09/2007)
Along
Cross
Radial
metres
Time (s)

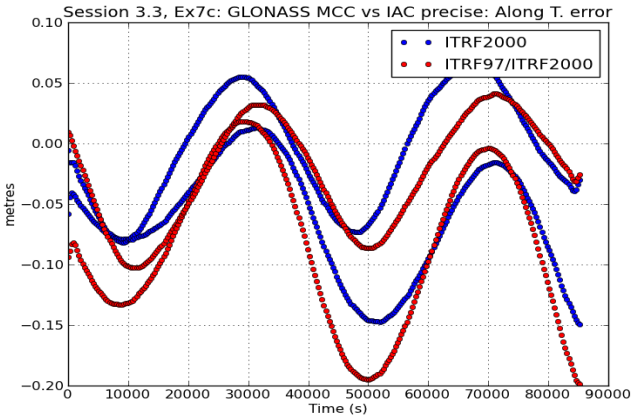

Session 3.3, Ex7a: GLONASS IGL vs IAC precise products
Along
Cross
Radial
metres
Time (s)

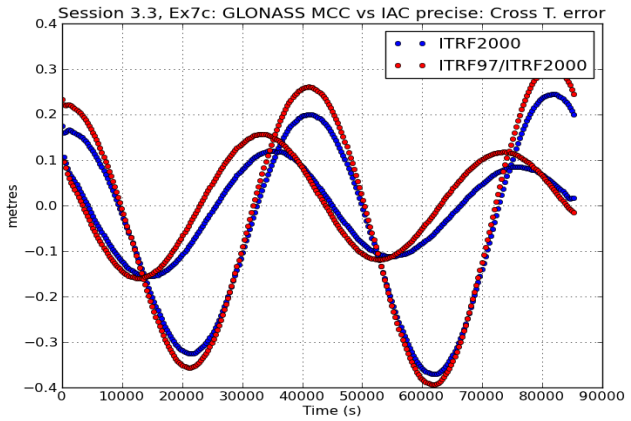

Session 3.3, Ex7a: GLONASS IGL vs IAC precise products
metres
Time (s)


Session 3.3, Ex7b: GLONASS MCC vs IAC precise (ITRF97/ITRF2000)
Along
Cross
Radial
metres
Time (s)


Session 3.3, Ex7c: GLONASS MCC vs IAC precise: 3D error
ITRF2000
ITRF97/ITRF2000
metres
Time (s)


Session 3.3, Ex7c: GLONASS MCC vs IAC precise: Radial error
ITRF2000
ITRF97/ITRF2000
metres
Time (s)


Session 3.3, Ex7c: GLONASS MCC vs IAC precise: Along T. error
ITRF2000
ITRF97/ITRF2000
metres
Time (s)


Session 3.3, Ex7c: GLONASS MCC vs IAC precise: Cross T. error
ITRF2000
ITRF97/ITRF2000
metres
Time (s)

4. Measurement Analysis

GNSS measurements and their different combinations are analysed in detail in the next three laboratory sessions. As in the previous chapter, `gLAB` is used in command-line mode and new functionalities and capabilities for the measurement analysis are introduced and explained.

The first session focuses on multipath and ionospheric effects, which are directly measurable from the signals.

The code multipath is assessed from GPS measurements collected by static receivers and using the repeatability of the satellite geometry with the sidereal day. This is done for the C1 code and for the different combinations of two-frequency signals (ionosphere-free, Melbourne–Wübbena and geometry-free) over selected data sets collected under Anti-Spoofing (A/S) activated and deactivated conditions.

The analysis of the ionospheric effects on the GNSS signals[1] is one of the main targets of this session. The selected examples include events such as the solar flare on 28 October 2003 and its associated Halloween storm, one of the worst disturbances ever recorded. Data files collected from receivers in North America and Europe are used to illustrate and compare its effects on such regions.

A Medium-Scale Travelling Ionospheric Disturbance (MSTID) is also depicted, using GPS measurements from a small permanent stations network, as another example of ionospheric effects on GNSS signals.

In the last exercise of this session, an assessment of the code–carrier ionospheric divergence on single-frequency carrier-smoothed codes is undertaken. This example is introduced to illustrate the trade-off between the measurement noise reduction by the single-frequency smoothing and code–carrier divergence.

The second session is devoted to the analysis of measurements from the three-frequency Galileo and modernised GPS signals. The measurement noises are compared for individual signals and for different combinations of pairs of signals. In this way, analyses are conducted on how the noise is amplified depending on the combination used. From the results, the suitability of using the different possible combinations to navigate or to estimate the ionospheric delays is discussed, as well. The new wide-lane combinations and the extra-large wavelength achievable with these new signals are also studied together with the multipath noise analysis.

[1] Mostly with GPS, but including an exercise with Glonass. The three-frequency Galileo signals are analysed in session 4.2.

The examples include a data set collected by a GPS+Galileo receiver where a large anomalous noise signature appears. This anomaly is roughly analysed from both Galileo and GPS signals to assess whether it is due to the SIS or to local environment phenomena.

The last session is devoted to an advanced topic, where new combinations of three-frequency signals (carriers or codes) are built as an academic exercise, and discussing its suitability for GNSS applications.

In particular, the discussion focuses on the feasibility of estimating the second-order ionospheric effects using a combination of three carriers. Indeed, it is shown how a combination can be built cancelling both the geometry and the first-order ionospheric effects. Then, using actual measurements from Galileo and from modernised GPS signals, this combination is plotted and the associated noise and the effects of APC calibration are depicted.

Other combinations explored are: (1) to cancel not only the first-order ionospheric effects (it is done with two-frequency signals), but also second-order ones; and (2) to cancel the geometry and the second-order ionosphere. These combinations are plotted and their suitability for navigation (1) or for ionospheric determinations (2) is discussed.

Session 4.1. Code and Carrier Measurement Analysis

Objectives

To analyse graphically the GPS code and phase measurements and their combinations, as well as studying their characteristics and properties: *cycle slips*, ionospheric refraction, multipath, receiver noise, including measurements under A/S on/off conditions. To determine empirically the order of magnitude of these effects.

Files to use

`casa2910.95o`, `brdc2910.95n`, `coco0090.97o`, `brdc0090.97n`, `mhcb2910.01o`, `monb2910.01o`, `sodb2910.01o`, `UPC33510.08O`, `brdc34[5-7]0.04n`, `brdc30[0-6]0.03n`, `brus30[0-6]0.03d`, `garl30[0-6]0.03d`, `htv134[5-7]0.04o`, `galb34[5-7]0.04o`, `gage27[1-3]0.98o`, `UPC336[0-2]0.08O`, `UPC536[0-2]0.08O`, `dlft1810.09`, `iac15382.sp3`, `upci00178.tec0.anim.gif`

Programs to use

`gLAB_linux, graph.py`

Preliminary

The `gLAB` tool suite implements a specific function to read different RINEX file formats. This function can be executed both from the GUI panel or in console mode.

As an example, the following command makes `gLAB` read the RINEX measurement and navigation files, `casa2910.95o` and `brdc2910.95n`, and generates a text file with the data written in a column format, according to Table 4.1:

```
gLAB_linux -input:cfg meas.cfg -input:obs casa2910.95o
           -input:nav brdc2910.95n --pre:prealign > casa.meas
```

The last argument '`--pre:prealign`' (with double '`--`') is set to avoid the pre-alignment of carrier measurements with the code (shifting the carrier by an integer number of wavelengths), which is done by default. In general, this option is useful to avoid the large bias in the carrier measurements, but in the example that follows we are interested in showing the jumps in the carrier phase ambiguities after the cycle slips occur.

The navigation file `brdc2910.95n` is needed to compute satellite elevation and azimuth.[2] If the navigation file is not provided, then the elevation and azimuth are set to zero.

[2] By default, `gLAB` uses the '`A priori receiver coordinates`' given in the RINEX files. Such coordinates can also be provided as external values or computed by the program itself, using the input option `-pre:setrecpos <val>` (see `gLAB_linux -help`).

Table 4.1: Description of MEAS rows.

Field	Content
1	'MEAS'
2	Year
3	DoY
4	Seconds of day
5	GNSS (GPS, GAL, GLO or GEO)
6	PRN satellite identifier
7	Elevation of the satellite [degrees]
8	Azimuth of the satellite [degrees]
9	Number of measurements
10	List of measurements included [e.g. C1P:L1P:C2P:L2P] Note: The measurement descriptors follow the RINEX-3.01 notation.
11–	Each one of the subsequent columns is the measurement value specified (in m), according to the order specified in field 10. (The carrier phase measurements are also given in m, except for Glonass satellites, which are in cycles)

The previous sentence generates the `casa.meas` file with the contents given in Table 4.1. The file `meas.cfg` is a configuration file used to change the default configuration of `gLAB` (which is preset to navigate in SPP or PPP mode).[3] This file sets the input parameters as in Table 4.2. (Note that the double '`--`' is used to disable a parameter.)

The `gLAB` tool suite is able to read measurement files in RINEX-2.11 and RINEX-3.0 formats. These formats include GPS, Glonass, Galileo and Beidou measurements. The tool suite also reads GPS, Glonass, Galileo and Beidou satellite coordinates from SP3 files. Nevertheless, only the GPS RINEX navigation files can be read by the current `gLAB-version-2.0.0`.

Note that the SP3 files with precise coordinates can also be used instead of the RINEX navigation file. In this case the argument is '`-input:sp3 <file>`' instead of '`-input:nav <file>`'.

[3]The `gLAB` default configuration is set to navigate in PPP mode when SP3 files with precise orbits and clocks are given as input files, or in SPP when the RINEX navigation file is provided instead of the SP3 file. These default configurations for SPP and PPP involve, in particular, 300 seconds of data decimation (`-pre:dec 300`) and the observation model computation only when the satellite clocks are available (this can be disabled by `--model:satclocks`). In the case of PPP, it also includes the satellite and receiver APC computation (from an ANTEX file, by default) and the two- frequency cycle-slip detection with the geometry-free (li) and Melbourne–Wübbena (bw) combinations. The (li) and (bw) cycle-slip detections can be unset with the input options `--pre:cs:li` and `--pre:cs:bw`, respectively.

Table 4.2: meas.cfg configuration file description. Single-frequency cycle-slip detection and carrier pre-alignment are done by default. The parameter '-model:dcb:p1c1 flexible' is set by default. This flexible mode provides P1 from C1 code when P1 is missing. It is required for GPS L1 cycle-slip detection, because it uses the P1 code (not C1).

Parameter	Description
-pre:dec 1	Output sampling rate at 1 s (default 300)
-print:none	Disable all output messages
-print:meas	Print the message 'MEAS' (see Table 4.1)
--model:satclocks	Disable the computation of satellite clocks (to provide output even when clocks unavailable)
--model:satphasecenter	Disable the satellite and receiver APC comput.
--model:recphasecenter	(hence the ANTEX file will not be required)
--pre:cs:li	Disable the cycle-slip detection with
--pre:cs:bw	geometry-free (li) & Melbourne–Wübbena (bw)

Notation

Since the RINEX files are used as the main input data in the exercises, the notation used to indicate the code and carrier measurements will follow the RINEX nomenclature and thus will be slightly different than in Volume I.

For instance, C1, P1, P2 will be used to represent the code pseudorange measurements associated with C1, P1 and P2 codes instead of $R_{P_{f_i}}$ or R_i, or L1, L2 for carrier phase measurements instead of $\Phi_{L_{f_i}}$ or Φ_i.

Other examples are PC and LC, which will indicate code and carrier ionosphere-free combinations of P1, P2 codes and L1, L2 carriers. These combinations are referred to as R_C and Φ_C in Volume I. See exercises 2 to 8 in this session, or equations (4.4) to (4.35) in session 4.2. Table 4.3 summarises these notation changes.

Table 4.3: Different notations used in Volume I and Volume II.

Measurements	Volume I	Volume II
Raw measurements:	$R_{P_{f_i}}$, R_i	C1, P1, P2,...,P7, P8
	$\Phi_{L_{f_i}}$, Φ_i	L1, L2, L5,...,L7, L8
Ionosphere-free combinations	R_C	PC,...,$PC_{[1,2]}$,...,$PC_{[7,8]}$
	Φ_C	LC,...,$LC_{[1,2]}$,...,$LC_{[7,8]}$
Geometry-free combinations	R_I	PI,...,$PI_{[1,2]}$,...,$PI_{[7,8]}$
	Φ_I	LI,...,$LI_{[1,2]}$,...,$LI_{[7,8]}$
Wide-lane combination	Φ_W	LW,...,$LW_{[1,2]}$,...,$LW_{[7,8]}$
Narrow-lane combination	R_N	PN,...,$PN_{[1,2]}$,...,$PN_{[7,8]}$

Note: To simplify notation, we will use the RINEX-2.1 measurement descriptors in the exercises (i.e. C1, P1, P2, L1, L2..., as given above in Table 4.3). Nevertheless, gLAB uses the RINEX-3.01 measurement descriptors, as indicated in Table 4.1 (i.e. C1C, C1P, C2P, L1P, L2P...).

Development

Session files.

Copy and uncompress these session files in the working directory:

```
cp ~/GNSS/PROG/SES41/* .
cp ~/GNSS/FILES/SES41/* .
gzip -d *.gz *.Z
```

1. Cycle slips

Generate the file `casa.meas` as indicated before and draw the following plots to analyse the measurements:

(a) Taking into account the MEAS file description of Table 4.1, verify the following field contents in the generated file `casa.meas`:

```
MEAS YY DoY sec GPS PRN El Az N. list C1P L1P C2P L2P
  1   2  3   4   5   6   7  8  9  10   11  12  13  14
```

(b) Plot the carrier phase measurement L1 as a function of time for the satellite PRN28 and identify the epochs in which *cycle slips* occur. Why do negative values appear in such carrier measurements?

Execute for instance:

```
graph.py -f casa.meas -c '($6==28)' -x4 -y12 -l "L1"
```

(c) Plot phase L1 and code P1 on the same graph.

Execute for instance:

```
graph.py -f casa.meas -c '($6==28)' -x4 -y11 -l "P1"
         -f casa.meas -c '($6==28)' -x4 -y12 -l "L1"
```

(d) Repeat for L2 and P2 (optional).

2. Multipath (PC)

The code multipath can be seen by plotting the difference of code and carrier ionosphere-free combinations (PC–LC). The evolution of this difference can be followed with a sampling rate of 1 Hz. Due to its geometric nature, the effect of multipath repeats with the receiver–satellite geometry.

The RINEX files `UPC33600.08N`, `UPC33610.08N`, `UPC33620.08N` contain observations at 1 Hz collected by a GPS receiver with fixed coordinates over the same period of time on three consecutive days.[4] The corresponding navigation files are `BRD3600.08N`, `BRD3610.08N`, `BRD3620.08N`.

Using `gLAB`, read the RINEX files, plot the combination PC–LC and identify the effect of multipath.[5] Analyse the measurements of satellites PRN20 and PRN25 in the time interval $67\,000 < t < 68\,000$ s. Include the satellite's elevation in the plots.

[4] These files were collected with the dual-frequency receiver Ashtech Z12, with the Aero Antenna Technology (AT2775-01W).

[5] Overlap the graphics and shift them along the x-axis to make the comparison easier.

To read the RINEX files, execute:

```
gLAB_linux -input:cfg meas.cfg -input:obs UPC33600.08O
                           -input:nav BRD3600.08N > upc3360.meas
```

(Similarly for the other two files `UPC33610.08N` and `UPC33620.08N`.)

Taking into account the MEAS file description of Table 4.1, verify the following field contents in the generated file `upc3360.meas` and others:

```
MEAS YY DoY sec GPS PRN El Az N. list C1C L1C C1P L1P C2P L2P
  1   2  3   4   5   6   7  8  9  10   11  12  13  14  15  16
```

Complete the following steps:

1. Compute the difference of code and carrier ionosphere-free combinations (see the ionosphere-free combination in equation (4.4), Volume I, or in equation (4.14) of session 4.2). Select the time interval [66 000: 69 000] to avoid big data files:

```
gawk 'BEGIN{g12=(154/120)**2} {if ($4>66000 && $4< 69000)
   print $6,$4,(g12*$13-$15)/(g12-1)-(g12*$14-$16)/(g12-1),$7}'
                                        upc3360.meas > upc3360.LcPc
```

(Similarly for the other two files `upc3361.meas` and `upc3362.meas`.)

2. Plot the results:

```
graph.py -f upc3360.LcPc -c '($1==20)' -x2 -y3 -s- -l "DoY 360"
         -f upc3361.LcPc -c '($1==20)' -x2 -y3 -s- -l "DoY 361"
         -f upc3362.LcPc -c '($1==20)' -x2 -y3 -s- -l "DoY 362"
         -f upc3360.LcPc -c '($1==20)' -x2 -y4 -l "Elev (deg.)"
```

(a) Compare the plots corresponding to the three days.

(b) Repeat the same plots, but shift the plot of the second day by $3^{m}\,56^{s} = 236^{s}$ and the third day by $2 \times (3^{m}\,56^{s}) = 472^{s}$.

```
graph.py -f upc3360.LcPc -c '($1==20)' -x2 -y3 -s- -l "DoY 360"
         -f upc3361.LcPc -c '($1==20)' -x'($2+236)' -y3 -s- -l "DoY 361"
         -f upc3362.LcPc -c '($1==20)' -x'($2+472)' -y3 -s- -l "DoY 362"
         -f upc3360.LcPc -c '($1==20)' -x2 -y4 -l "Elev (deg.)"
```

(c) What is the reason for the observed $3^{m}\,56^{s}$ displacement between the graphs for two consecutive days? (See section 4.2.2, Volume I.)

(d) Repeat the previous plot for the satellite PRN25 and compare results.

(e) The RINEX files `UPC53600.08O, UPC53610.08O, UPC53620.08O` have been collected by another permanent receiver[6] located a few metres from the previous one. Repeat the previous computations and plots for these data files and compare the results.

Why are the patterns seen for the UPC3 and UPC5 receivers different?

[6]The receiver used was a dual-frequency Novatel OEM3 with a Novatel 600 L1/L2 GPS antenna (pinwheel).

Note: The P1 code is not available in these files, but, with the model option '`-model:dcb:p1c1 flexible`*' being set by default, the P1 code is emulated from the C1 code.*[7] *The P1 code is required by* gLAB *for GPS L1 cycle-slip detection and carrier pre-alignment.*

3. **Multipath (PC)** (optional)

   Repeat the multipath analysis of exercise 2 using files `htv13450.04o, htv13460.04o, htv13470.04o` collected by the permanent receiver HTV1, and files `galb3450.04o, galb13460.04o, galb13470.04o` collected by the permanent receiver GALB. The associated broadcast orbit files are `brdc3450.04n, brdc3460.04n, brdc3470.04n`.

   Analyse the satellite PRN03 in the time interval $46\,000 < t < 63\,000$.

4. **Multipath (MB)** (optional)

   Applying a similar approach as in the previous exercise, and taking into account equations (4.19) in Volume I (or equation (4.20) of session 4.2), plot the multipath effect in the Melbourne–Wübbena combination.

   After generating file `galb3460.meas`, execute:

```
gawk 'BEGIN{s12=154/120} {print $6,$4,
      (s12*$14-$16)/(s12-1)-(s12*$13+$15)/(s12+1),$7}' galb3450.meas > galb3450.MB
```

(Similarly for the other two files `galb3460.meas` and `galb3470.meas`.)

5. **Multipath (C1)**

   The code multipath can also be depicted by plotting the combination P1–L1 (with a sampling rate of 1 Hz, it is also possible to follow its evolution). The RINEX files `gage2710.98o`, gage2720.98o and `gage2730.98o` contain observations at 1 Hz collected over the same period of time on three consecutive days.[8].

   Using `gLAB`, read the RINEX files, plot the combination P1–L1 and identify the effect of multipath.

   As in previous exercises, read the RINEX files by executing:

```
gLAB_linux -input:cfg meas.cfg -input:obs gage2710.98o > gage271.meas
```

(Similarly for files `gage2720.98o` and `gage2730.98o`.)

   Produce the plots by executing:

```
graph.py -f gage271.meas -c '($6==14)' -x4 -y'($11-$14)' -l "DoY 271"
         -f gage272.meas -c '($6==14)' -x4 -y'($11-$14)' -l "DoY 272"
         -f gage273.meas -c '($6==14)' -x4 -y'($11-$14)' -l "DoY 273"
```

   (a) Compare the plots corresponding to the three days. Are the patterns similar?

[7] As explained in `gLAB_linux -help`, in `Flexible` mode `gLAB` identifies C1 and P1 codes when one of them is missing. On the other hand, in this `Flexible` mode `gLAB` does not correct DCBs between P1 and C1 codes. This correction is only applied in `Strict` mode, with the required `P1C1YYMM.DCB` and `GPS_Receiver_Types` files. For more details on `Flexible` and `Strict` P1–C1 DCBs, see exercise 3 of session 5.2.

[8] These files were collected with the single-frequency receiver Lassen-SK8 (Trimble). This low-cost receiver provides codes and 'truncated' L1 carriers

(b) Repeat the same plots, but shift the plot of the second day by $3^{m}\,56^{s} = 236^{s}$ and the plot of the third day by $2 \times (3^{m}\,56^{s}) = 472^{s}$.
Execute:

```
graph.py -f gage271.meas -c '($6==14)' -x4 -y'($11-$14)' -l "DoY 271"
         -f gage272.meas -c '($6==14)' -x'($4+236)' -y'($11-$14)' -l "DoY 272"
         -f gage273.meas -c '($6==14)' -x'($4+472)' -y'($11-$14)' -l "DoY 273"
```

(c) What is the source of the observed drift?

(d) Why can it be affirmed that the oscillations seen are basically due to the effect of the C1 code multipath?

6. Ionospheric refraction (GPS, A/S off)

Plot the *ionospheric combination* L1–L2 for the satellite PRN28 (see equation (4.5) in Volume I, or equation (4.8) of session 4.2).
Execute for instance:

```
graph.py -f casa.meas -c '($6==28)' -x4 -y'($12-$14)'
```

What is the physical meaning of this combination?

(a) Repeat for P1–P2. Why does this combination show a different sign from the previous one? Does it make sense that the graph cuts the x-axis? What factors might affect the ionospheric refraction (geometry, local time, etc.)?[9]

(b) Overlap the combinations L1–L2 and P2–P1[10] for satellite PRN28 on the same graph. Add the satellite's elevation to the plot.
Execute for instance:

```
graph.py -f casa.meas -c '($6==28)' -x4 -y'($12-$14)' -l "L1-L2"
             -f casa.meas -c '($6==28)' -x4 -y'($13-$11)' -l "P2-P1"
                -f casa.meas -c '($6==28)' -x4 -y'($7/10)' -l "Elev/10"
```

Taking into account the previous plots, which combination shows a greater level of noise, L1–L2 or P1–P2?

(c) Plot the combination (L1–L2) and (P2–P1) on the same graph. Why does the dispersion increase at the beginning and end of the arcs?

7. Ionospheric refraction (GPS, A/S on)

Using the same procedure as before, use the RINEX files `coco0090.97o` and `brdc0090.97n` with `gLAB` to generate a measurements file with the contents of Table 4.1.

[9] An animation showing the time evolution of vertical ionospheric delays (actually the TEC) on a planetary scale is given in `upci00178.tec0.anim.gif`. This movie can be seen with any Internet browser (Firefox, Internet Explorer, etc.). It has been generated from the UPC IONEX files provided on the IGS server *ftp://cddis.gsfc.nasa.gov/gps/products/ionex/*. Note that the L1–L2 or P2–P1 combination patterns correspond to the 'Slant' delay Slant Total Electron Content (STEC), in the direction of the satellite–receiver ray, see equations (4.2) and (4.3), Volume I. The STEC can be computed by multiplying the TEC by the slant factor, or by *mapping function* (see Fig. 5.13, Volume I). A crude approximation of the mapping (flat ionosphere) for angles significantly larger than zero is given by $FO \simeq 1/sen(elev)$, where $elev$ is the satellite's elevation referred to the user's horizon.

[10] Note that the ionospheric refraction has opposite sign for code and carrier phase measurements, see equations (4.2) and (4.3) in Volume I.

```
gLAB_linux -input:cfg meas.cfg -input:obs coco0090.97o
                    -input:nav brdc0090.97n > coco.meas
```

(a) As in the previous exercise, plot L1–L2 and P2–P1 for the satellite PRN15 on the same graph. Explain the dispersion pattern of the points seen in the figure[11] for code measurements. Why are they larger at lower elevations?

(b) Represent L1–L2 and P2–P1 for satellite PRN01 (for instance) on the same plot. Compare the dispersion of the values obtained for P2–P1 with those of the previous exercise (for 18 October 1995, with A/S off). How significant are the differences found?

(c) Discuss the shape of the elevation curve seen for satellites PRN01 and PRN15.
(*Hint: The COCO receiver is closer to the equator).*

8. Ionospheric refraction (GPS, A/S on)

Using the file `coco.meas` of the previous exercise, draw the following plots and justify why the curves have the same shape. Taking into account equations (4.19) and Table 4.2 in Volume I, justify also the factors used in the different combinations.

```
graph.py -f coco.meas -c '($6==01)' -x4 -y'($14-$16)*1.55' -l "L1-L2"
         -f coco.meas -c '($6==01)' -x4 -y'($11-$14)*0.50' -l "P1-L1"
         -f coco.meas -c '($6==01)' -x4 -y'($15-$16)*0.30' -l "P2-L2"
         -f coco.meas -c '($6==01)' -x4 -y'($7/10)' -l "Elev/10"
```

9. Ionospheric refraction (Glonass)

Using the same procedure as before, read the RINEX files `dlft1810.09o` and `iac15382.sp3` with `gLAB` to generate a measurement file with the contents of Table 4.1. Select the Glonass satellites.

```
gLAB_linux -input:cfg meas.cfg -input:obs dlft1810.09o
      -input:sp3 iac15382.sp3 --pre:prealign | grep GLO > dlft_g.meas
```

Note: The option `--pre:prealign` *must be applied for Glonass RINEX measurement files to avoid the pre-alignment of carriers with their code pseudoranges. This is because the current version of* `gLAB` *does not implement reading of the Glonass RINEX navigation files where the frequency channel associated with the orbit slot number (indicated as the PRN in the RINEX files) is given. This parameter is needed to compute the signal wavelength and convert cycles into metres, for correct pre-alignment.*

(a) Plot the P1–C1 and C1–C2 measurement combinations and interpret the results:

```
graph.py -f dlft_g.meas -c '($6==02)' -x4 -y'($11-$13)' -l "C1-P1"
          -f dlft_g.meas -c '($6==02)' -x4 -y'($11-$15)' -l "C1-C2"
           -f dlft_g.meas -c '($6==02)' -x4 -y'($7/10)' -l "Elev/10"
                                                  --yn -10 --yx 10
```

[11] The data file `coco0090.97o` was collected by a ROGUE SNR-8000 receiver. While the Coarse/Acquisition (C/A) code values are original measurements from C/A code, the P2 code values of this file are 'are indirectly obtained' from the C/A code and the cross-correlation of the encrypted P1 and P2 codes (Y1 and Y2). See exercise 3 in session 5.2.

(b) As commented earlier (see Table 4.1), the carrier phase measurements for Glonass satellites are given in cycles, thus they must be converted to metres using the proper frequency.

As can be seen in the RINEX navigation file `dlft1810.09g`, the frequency channel number associated with the Glonass satellite PRN02 (i.e. Orbit Slot Number 02) is $k = 1$. Taking this information into account, justify that the next sentence provides, for the satellite PRN02, the following combination of measurements expressed in metres of delay: `sec, P2-P1, L1-L2, P1-L1, P2-L2, elev`.

See equations (4.19) and Table 4.2 in Volume I:

```
cat dlft_g.meas | gawk 'BEGIN {k=1;c=299792458;f1=9/16*(2848+k)*1e6;
                  f2=7/16*(2848+k)*1e6;l1=c/f1;l2=c/f2} {if ($6==02)
                        {print $4,$15-$13,$14*l1-$16*l2,$13-$14*l1,
                                        $15-$16*l2,$7}}' > glo_02.meas
```

Repeat exercise 8 with the file `glo_02.meas`. Justify the factors used in the plot.

```
graph.py -f glo_02.meas -x1 -y'($6/10)' -l "Elev/10"
               -f glo_02.meas -x1 -y'($3*1.53)' -l "L1-L2"
                    -f glo_02.meas -x1 -y'($4*0.50)' -l "P1-L1"
                         -f glo_02.meas -x1 -y'($5*0.30)' -l "P2-L2"
                                        -t "GLO 02" --yn -17 --yx 10
```

(c) (Optional) Repeat the computations and plots for Glonass satellite PRN18.

Note that, according to the navigation file `dlft1810.09g`, the frequency channel number associated with Orbit Slot Number 18 (i.e. PRN18 in the RINEX file) is $k = -3$.

10. Solar flare

On 28 October 2003, an intense solar eruption (a solar flare) was detected around 11:00 UT in an active region which had produced one of the largest sunspots ever seen by the SOlar Helioscopic Observatory (SOHO). It appeared as a bright flash in the SOHO ultraviolet images (see *http://www.youtube.com/watch?v=tAyJHLf_NCw*). This sudden enhancement of the solar radiation in the X-ray and extreme ultraviolet bands produced a sudden increase in the ionospheric electron density on the daylight hemisphere.

Analyse the effect of the solar flare on the STEC measurements of four permanent IGS receivers (ANKR, ASC1, KOUR, QAQ1), covering a wide range of longitude and latitude. Use the files `ankr3010.03o`, `asc13010.03o`, `kour3010.03o` and `qaq13010.03o`.

Execute:

```
gLAB_linux -input:cfg meas.cfg -input:obs ankr3010.03o >ankr3010.03.meas
gLAB_linux -input:cfg meas.cfg -input:obs asc13010.03o >asc13010.03.meas
gLAB_linux -input:cfg meas.cfg -input:obs kour3010.03o >kour3010.03.meas
gLAB_linux -input:cfg meas.cfg -input:obs qaq13010.03o >qaq13010.03.meas
```

Plot the ionospheric combination LI=L1–L2 to assess the effect of the flare on the GPS measurements:

```
graph.py -f ankr3010.03.meas -x4 -y'($14-$16)' -l "ankr"
         -f asc13010.03.meas -x4 -y'($14-$16)' -l "asc1"
         -f kour3010.03.meas -x4 -y'($14-$16)' -l "kour"
         -f qaq13010.03.meas -x4 -y'($14-$16)' -l "qaq1"
                  --xn 38500 --xx 40500 --yn -20 --yx 20
```

11. Halloween storm

Associated with the solar flare analysed in the previous exercise, a coronal mass ejection occurred, sending a large particle cloud to impact Earth's magnetosphere about 19 hours later, on 29 October. Subsequent impacts were still occurring several hours after this. The material interacted with Earth's magnetosphere and a Storm Enhancement Density (SED) appeared in North America and later affected northern latitudes in Europe. Extra large gradients of the TEC associated with this phenomenon were also produced, degrading the GPS positioning performance ([HJS et al., 2005]).

The evolution of the TEC on 30 October 2003 (i.e. day 303 of year 2003) can be seen in the movie `TEC_2003oct30_anim.gif`.

The measurement files `garl3010.03o`, `garl3020.03o`, `garl3030.03o`, `garl3040.03o`, `garl3050.03o` and `garl3060.03o` were collected by the permanent receiver GARL in Empire, NV, USA ($\varphi = 40.42°, \lambda = -119.36°$) from 28 October to 2 November 2003. Using these files, plot the STEC for all satellites in view and discuss the range of any variations. Analyse, in particular, satellite PRN04 and calculate the maximum rate of STEC variation in mm/s of L1 delay. Add the elevation of satellite PRN04 to the plot.

The associated RINEX navigation files are `brdc3010.03n`, `brdc3020.03n`, `brdc3030.03n`, `brdc3040.03n`, `brdc3050.03n` and `brdc3060.03n`.

Complete the following steps:

1. Read the RINEX files:

```
gLAB_linux -input:cfg meas.cfg -input:obs garl3010.03o
              -input:nav brdc3010.03n > garl3010.03.meas
```

Repeat for the other files `garl3020.03o, garl3030.03o, garl3040.03o, garl3050.03o, garl3060.03o`

2. Plot the results:

   (a) Plot the STECs for all days from 28 October to 1 November 2003. Merge all files into a common file `garl.meas`, where the 'seconds' (the fourth field) are given with respect to 28 October (day 301), and convert the seconds to hours:[12]

[12] This is done by the instructions `d=($3-301)*86400;$4=$4+d` and `$4=$4/3600` in the next sentence.

```
cat garl3010.03.meas garl3020.03.meas garl3030.03.meas
    garl3040.03.meas garl3050.03.meas garl3060.03.meas |
    gawk '{d=($3-301)*86400;$4=$4+d;$4=$4/3600;print $0}' > garl.meas
```

Produce the plot:

```
graph.py -f garl.meas -x4 -y'($15-$11)' -l "All sat."
            -f garl.meas -c'($6==04)' -x4 -y7 -l "PRN04:ELEV"
              -f garl.meas -c'($6==04)' -x4 -y'($15-$11)' -l "PRN04"
                                    --xn 0 --xx 144 --yn -10 --yx 70
```

(b) Plot the STECs from 22:00 on 30 October to 06:00 on 31 October.

Merge the files `garl3030.03.meas` and `garl3040.03.meas` into a common file `garl.meas` as in the previous case. Now take the time relative to 00:00 on 30 October:

```
cat garl3030.03.meas garl3040.03.meas |
    gawk '{d=($3-303)*86400;$4=$4+d;$4=$4/3600;print $0}' > garl.meas
```

Produce the plot:

```
graph.py -f garl.meas -x4 -y'($15-$11)' -l "All sat."
            -f garl.meas -c'($6==04)' -x4 -y7 -l "PRN04:ELEV"
              -f garl.meas -c'($6==04)' -x4 -y'($15-$11)' -l "PRN04"
                                    --xn 22 --xx 30 --yn -10 --yx 70
```

12. Halloween storm (North America)

The data file `amc23030.03o` was collected by the permanent receiver AMC2 at Colorado Springs in the USA ($\varphi = 38.8°, \lambda = -104.52°$) on 30 October 2003. Plot the STEC in the time interval $55\,000 < t < 86\,400$ seconds for the satellites PRN13, 28 and 29 and discuss the results. Include the elevations of the satellites in the same plot.

The associated broadcast orbit and clock file is `brdc3030.03n`.

Execute:

```
gLAB_linux -input:cfg meas.cfg -input:obs amc23030.03o
                               -input:nav brdc3030.03n > amc23030.03.meas
```

```
graph.py -f amc23030.03.meas -x4 -y'($15-$13)' -l "All sat."
         -f amc23030.03.meas -c'($6==28)' -x4 -y7 -l "PRN28:  ELEV"
         -f amc23030.03.meas -c'($6==28)' -x4 -y'($15-$13)' -so -l "PRN28"
         -f amc23030.03.meas -c'($6==29)' -x4 -y7 -l "PRN29:  ELEV"
         -f amc23030.03.meas -c'($6==29)' -x4 -y'($15-$13)' -so -l "PRN29"
         -f amc23030.03.meas -c'($6==13)' -x4 -y7 -l "PRN13:  ELEV"
         -f amc23030.03.meas -c'($6==13)' -x4 -y'($15-$13)' -so -l "PRN13"
                                    --xn 55000 --xx 95000 --yn 0 --yx 85
```

13. Halloween storm (Europe)

The measurement files `brus3010.03o`, `brus3020.03o`, `brus3030.03o` and `brus3040.03o` were collected by the permanent receiver BRUS in Brussels, Belgium ($\varphi = 50.8°, \lambda = 4.36°$), from 28 to 31 October 2003. Plot the STEC in the time interval $18 < t < 24$ hours for the satellite PRN15 and discuss the results. Include the elevations of the satellite in the plot.

The broadcast orbit and clock files are `brdc3010.03n`, `brdc3020.03n`, `brdc3030.03n` and `brdc3040.03n`.

Execute:

```
gLAB_linux -input:cfg meas.cfg
  -input:obs brus3010.03o -input:nav brdc3010.03n |
    gawk '{if ($6==15) print $4/3600,$14-$16,$7}' > brus1
```

Repeat the same procedure for the other files `brus3020.03o`, `brus3030.03o` and `brus3040.03o`.

```
graph.py -f brus1 -l "28 Oct" -f brus2 -l "29 Oct"
                -f brus3 -l "30 Oct" -f brus4 -l "31 Oct"
                    -f brus4 -x1 -y'($3/10)' -l "Elev/10"
                          --xn 18 --xx 24 --yn -15 --yx 9
```

Discuss the shapes of the STEC patterns seen on the different days.

14. Propagation of medium-scale travelling ionospheric disturbances

A Travelling Ionospheric Disturbance (TID) is understood as a plasma density fluctuation that propagates through the ionosphere at an open range of velocity and frequency. The trend of such fluctuations can be seen from the geometry-free combination of GPS carrier measurements $LI = L_1 - L_2$.

Some authors distinguish between Large-Scale TIDs (LSTIDs) with a period greater than 1 h and moving faster than 0.3 km/s, and Medium-Scale Travelling Ionospheric Disturbances (MSTIDs) with shorter periods (from 10 min to 1 h) and moving slower (0.05–0.3 km/s). LSTIDs seem to be related to geomagnetic disturbances (i.e. aurora, ionospheric storms, etc.). The origin of MSTIDs seems to be related more to meteorological phenomena such as neutral winds, eclipses, or the solar terminator that produces Atmospheric Gravity Waves (AGW)s manifested as TIDs at ionospheric heights, due to the collision between neutral and ionised molecules.

In [HJS, 2006], a quite simple method is given to depict MSTIDs. It consists of 'detrending' the geometry-free combination of GPS carrier measurements from the dependence on diurnal variation and elevation angle, by applying the equation

$$\delta LI(t) = LI(t) - (LI(t+\tau) + LI(t-\tau))/2 \tag{4.1}$$

where a τ value of 300 s is sufficient to retain the variation of LI (i.e. STEC). With equation (4.1), the detrending is done simply by subtracting from each value an average value of the previous and a posteriori measurements (i.e. the curvature of the LI temporal evolution).

It must be stressed that this detrending procedure can be used in real time with a single receiver, so it is suitable for identifying these ionospheric perturbations in navigation applications.

The propagation of an MSTID will be depicted as follows using the measurements from three stations, SODB, MHCB and MONB, separated by a few tens of kilometres. The aim is to reproduce figure 10 in the paper by [HJS, 2006].

Complete the following steps:

(a) Read the files `mhcb2910.01o`, `monb2910.01o`, `sodb2910.01o`, and generate the corresponding `MEAS` measurement files.

Execute for instance:

```
gLAB_linux -input:cfg meas.cfg -input:obs mhcb2910.01o > mhcb.meas
gLAB_linux -input:cfg meas.cfg -input:obs monb2910.01o > monb.meas
gLAB_linux -input:cfg meas.cfg -input:obs sodb2910.01o > sodb.meas
```

(b) The MSTID is affecting a region sounded by the PRN14 GPS signal. Thus, this satellite's measurements will be selected for this study.

Execute for instance:

```
gawk '{if ($6==14) print $0}' mhcb.meas > mhcb_14.meas
gawk '{if ($6==14) print $0}' monb.meas > monb_14.meas
gawk '{if ($6==14) print $0}' sodb.meas > sodb_14.meas
```

(c) The next sentence applies the detrending of equation (4.1) to the geometry-free combination LI= L1–L2 of carrier measurements from previous files.

Execute:

```
gawk '{for (i=0;i<21;i++) {t[i]=t[i+1];l[i]=l[i+1]};
  t[21]=$4;l[21]=$14-$16;if (NR>21){tt=t[0]*t[10]*t[20];
       if (tt!=0) print t[10],(l[10]-(l[0]+l[20])/2)}}'
                              mhcb_14.meas > mhcb_dLi.meas
```

Apply this also to the other two data files `monb_14.meas` and `sodb_14.meas`.

(d) Plot the results for the time interval $55\,500 < t < 57\,000\,\mathrm{s}$.

Execute for instance:

```
graph.py -f mhcb_dLi.meas -l "mhcb:PRN14" -s.-
         -f monb_dLi.meas -l "monb:PRN14" -s.-
         -f sodb_dLi.meas -l "sodb:PRN14" -s.-
         --xn 55500 --xx 57000 --yn -0.05 --yx 0.05
```

(e) Taking into account the station coordinates and baselines given in the following table, discuss how the perturbation is propagating and calculate its propagation velocity:

Station	Latitude (deg.)	Longitude (deg.)	Distance to SODB (km)
SODB	36.98	238.07	0.0
MHCB	37.16	238.36	31.7
MONB	37.30	238.13	35.7

15. Carrier smoothing of code and divergence of the ionosphere

Noisy code can be smoothed with the precise (but ambiguous) carrier measurements. This carrier smoothing can be done in real time by applying the Hatch filter of equation (4.23) in Volume I. The smoothing depends on the filter length, that is N in equation (4.23). The more the filter length is used, the more smoothed the code is, but, with single-frequency measurements, a higher code–carrier divergence error is induced by the ionosphere. This is because the ionospheric refraction has opposite signs on the code and carrier, so its effect is twice the difference of the code and carrier, see equation (4.19) in Volume I. This double ionospheric refraction is propagated forward through the filter and its time variation produces a bias, see section 4.2.3.1 in Volume I.

The error induced by the code–carrier divergence of the ionosphere on the single-frequency smoothed codes is assessed as follows for different filter lengths. The performance of a divergence-free code smoothing is also analysed. In this last case a combination of two-frequency carriers is used to build up a new carrier with the same ionospheric delay (the same sign) as the code, to avoid the divergence of the ionosphere when smoothing the code.

The results of this exercise will reproduce Fig. 4.7 of section 4.2.3.1.2 in Volume I.

Complete the following steps:

(a) Read the RINEX file `UPC33510.08O` and generate the corresponding `MEAS` measurement file. Select satellite PRN03 (to reproduce the bottom left plot of Fig. 4.7 of Volume I).

Execute for instance:

```
gLAB_linux -input:cfg meas.cfg -input:obs UPC33510.08O |
                    gawk '{if ($6==3) print $0}' > upc3.meas
```

(b) Compute the C1 code multipath and receiver noise using the expression (see also equation (4.24))

$$\mathcal{M}_{C_1} \simeq C_1 - L_1 - 3.09\,(L_1 - L_2) \tag{4.2}$$

Execute for instance:

```
gawk '{print $4,$11-$14-3.09*($14-$16)}' upc3.meas > upc3.C1
```

(c) Apply the Hatch filter (equation (4.23) in Volume I) to smooth the code, using a filter length of $N = 60$ samples (as the measurements are at 1 Hz, this means 60 s of smoothing). Then, apply the previous expression (4.2) to depict the multipath and noise of the smoothed code.

These two computations can be done by a single command, executing for instance:

```
gawk 'BEGIN{Ts=60}{if (NR>Ts){n=Ts}else{n=NR};
          C1s=$11/n+(n-1)/n*(C1s+($14-L1p));L1p=$14;
print $4,C1s-$14-3.09*($14-$16)}' upc3.meas > upc3.C1S_60
```

(d) Using a combination of the two-frequency carriers, build up a new carrier with the same ionospheric delay (the same sign) as the code as follows:

$$L_{1_{DFree}} \simeq L_1 + 2 \times 1.55\,(L_1 - L_2) \tag{4.3}$$

Then, apply the Hatch filter to smooth code C_1 using this new carrier $L_{1_{DFree}}$. Finally, depict the code noise and multipath of this Divergence Free (DFree) smoothed code.

All these computations can be done by a single command, executing for instance:

```
gawk 'BEGIN{Ts=60}{if (NR>Ts){n=Ts}else{n=NR};
       L1f=$14+3.09*($14-$16);
       C1s=$11/n+(n-1)/n*(C1s+(L1f-L1fp));L1fp=L1f;
         print $4,C1s-L1f}' upc3.meas > upc3.C1DFreeS_60
```

(e) Plot the STEC for satellite PRN03.

Execute for instance:

```
graph.py -f upc3.meas -x4 -y'1.545*($14-$16)+12.5' -s.-
  --cl r -l "1.546*(L1-L2)" --xn 35000 --xx 40000
  --yn 1 --yx 3 --xl "time (s)" --yl "metres of L1 delay"
```

(f) Plot the previous results.

Execute for instance:

```
graph.py -f upc3.C1 -s- -l "C1:  PRN14"
          -f upc3.C1S_60 -s.- -l "C1 smoothed"
           -f upc3.C1DFreeS_60 -s- -l "C1 DFree smoothed"
                   --xn 35000 --xx 40000 --yn 17 --yx 25
```

(g) Repeat the previous computations and plots, but using a 360 s filter length.

(h) Repeat the previous computations and plots, but using a 3 600 s filter length.

(i) Discuss the results.

16. Carrier smoothing of the Halloween storm

Repeat the previous exercise using the RINEX file `amc23030.03o_1Hz` collected for station AMC2 on 30 October 2003. Select the time interval $56\,500 < t < 78\,160$ GPS seconds of day and satellite PRN13.

17. Ionospheric error in the single-frequency smoothed measurement

Assume that the STEC varies linearly in time,

$$I_1(t) = I_{0_1} + \dot{I}_1\, t$$

Show that the bias induced by the code–carrier divergence of the ionosphere, when the Hatch filter reaches steady state, is given by

$$bias_{I_1} = 2\left(\langle I_1\rangle_{(t)} - I_1(t)\right) = -2\tau\, \dot{I}_1$$

where τ is the smoothing time constant.

Solution:
1. Let $f(t) \equiv I_1(t)$ and $y(t) \equiv \langle I_1\rangle_{(t)}$ in equation (4.28) of Volume I. The (averaging) filter, $\langle x\rangle_k = [(k-1)/k]\langle x\rangle_{k-1} + (1/k)\, x_k$, can be implemented as

$$y(t+\Delta T) = \frac{\tau - \Delta T}{\tau} y(t) + \frac{\Delta T}{\tau} f(t+\Delta T)$$

Rearranging the previous equation yields

$$\frac{y(t+\Delta T) - y(t)}{\Delta t} + \frac{1}{\tau} y(t) = \frac{1}{\tau} f(t+\Delta T)$$

Assuming that the time constant is significantly larger than the sampling period ($\Delta T \ll \tau$), and taking the limit when $\Delta T \longrightarrow 0$, then the following differential equation can be found:

$$y' + \frac{1}{\tau} y = \frac{1}{\tau} f(t)$$

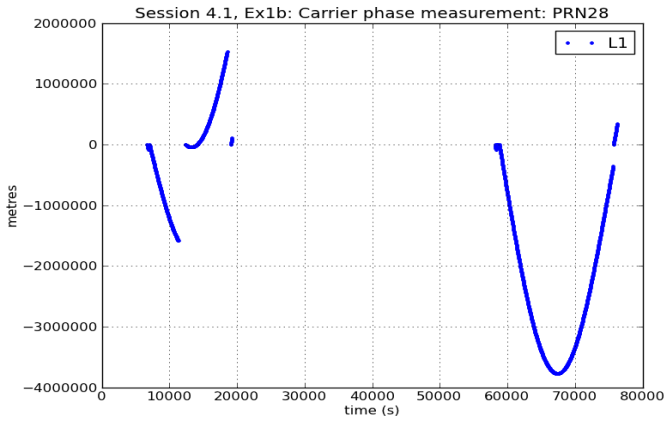
2. Take $f(t) \equiv I_1(t) = I_{0_1} + \dot{I}_1\, t$. Then the solution $y(t) \equiv \langle I_1\rangle_{(t)}$ of the previous differential equation, satisfying the initial condition $y(0) = I_{0_1}$, is

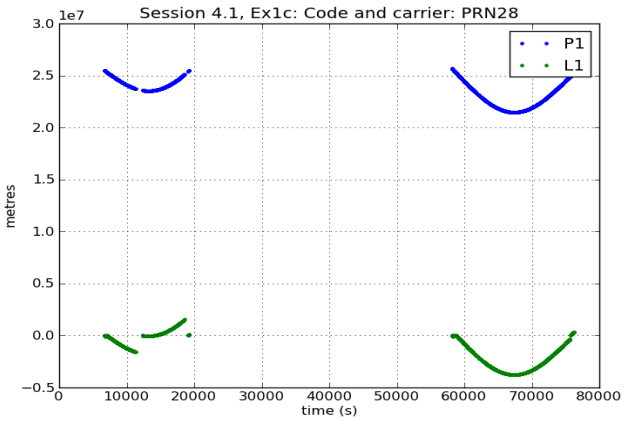
$$\langle I_1\rangle_{(t)} = I_1(t) - \tau\, \dot{I}_1\, (1 - \mathrm{e}^{-t/\tau})$$

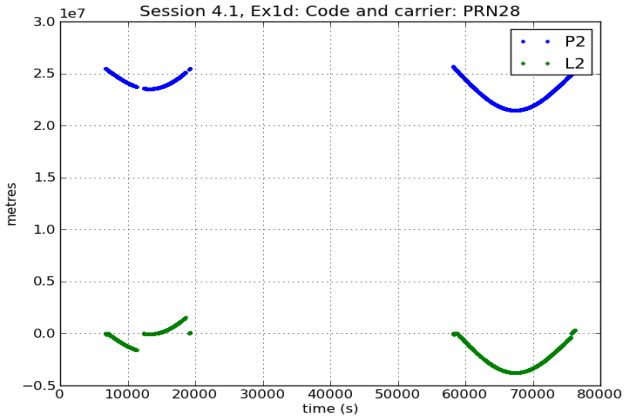
Thus, the bias induced by the code–carrier divergence of the ionosphere is

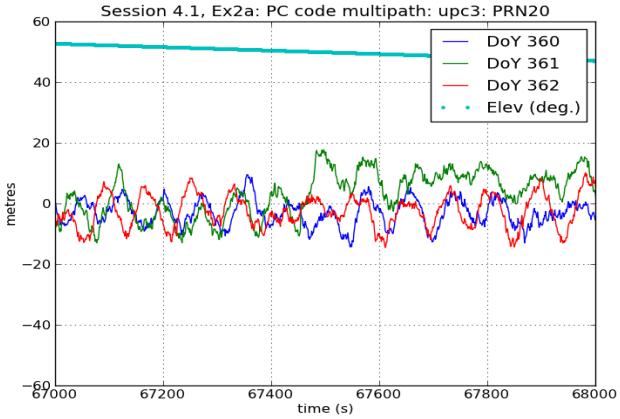
$$bias_{I_1} = 2\left(\langle I_1\rangle_{(t)} - I_1(t)\right) \xrightarrow[t\to\infty]{} -2\tau\, \dot{I}_1$$

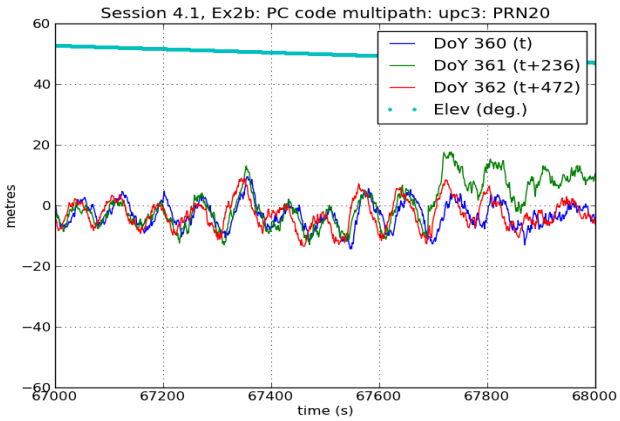
Graphs Session 4.1

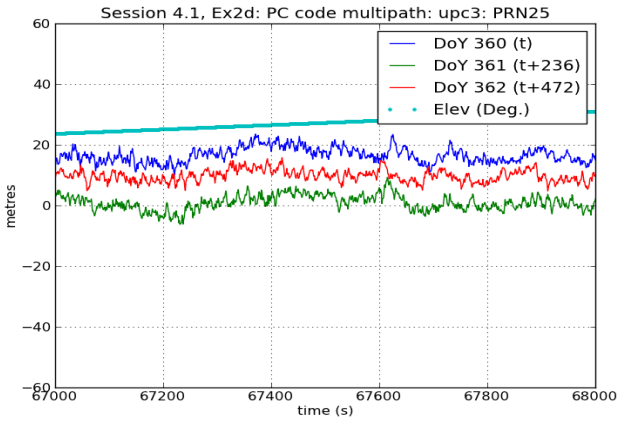


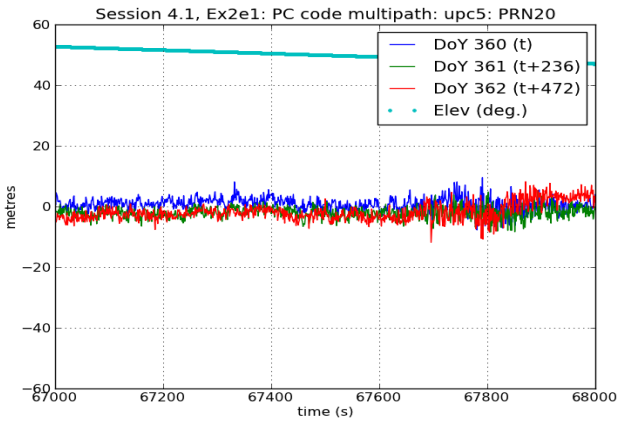


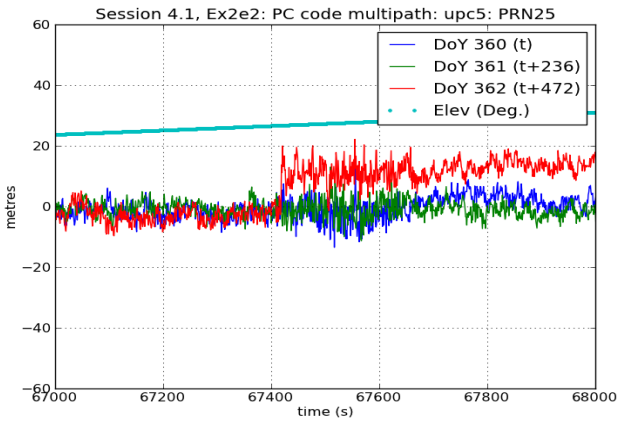


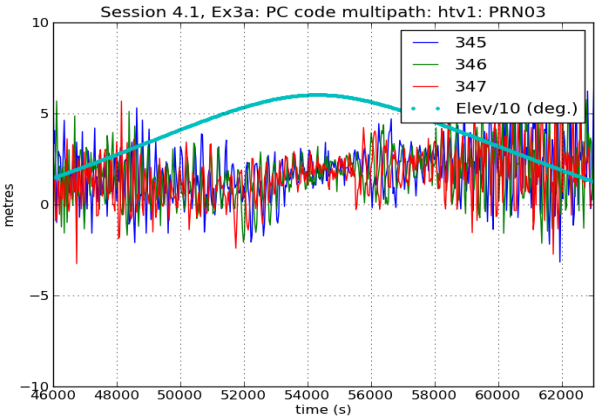


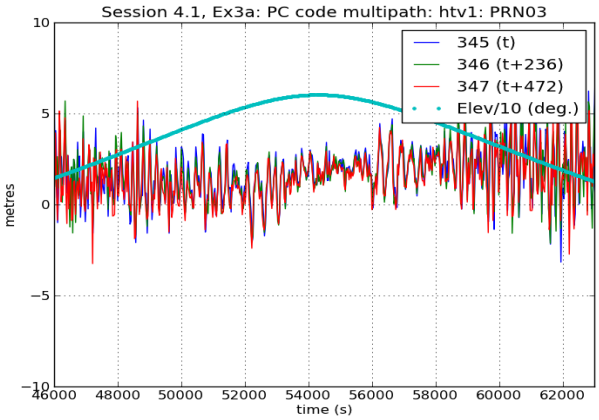


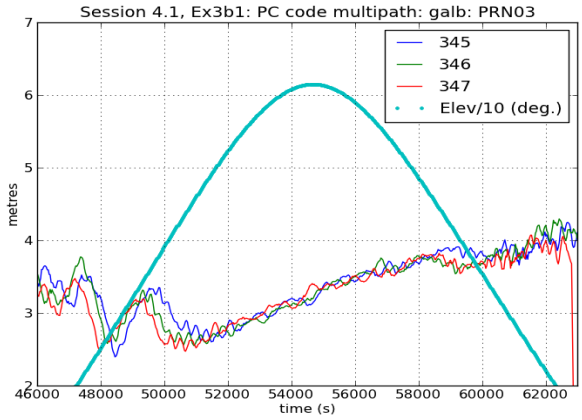


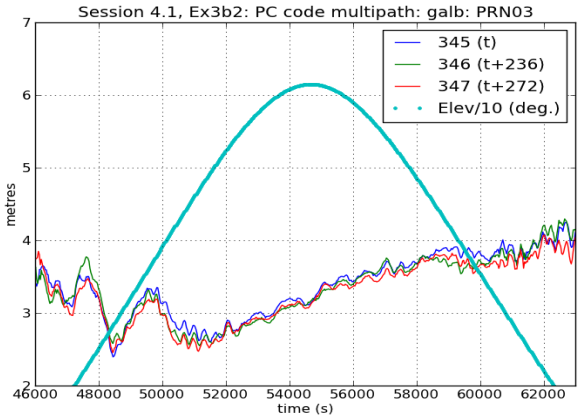


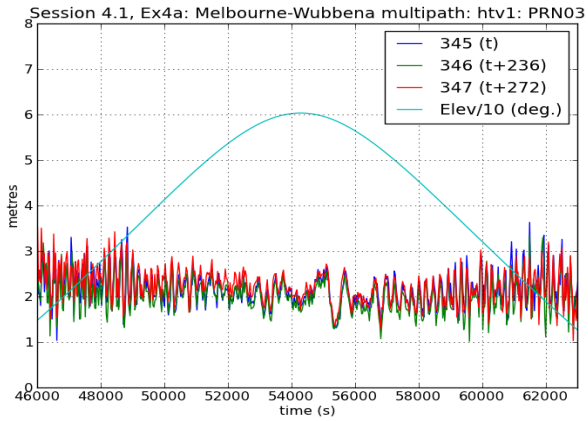


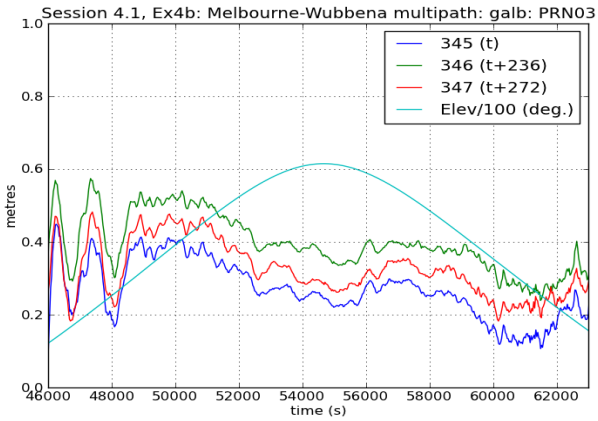

Session 4.1, Ex3a: PC code multipath: htv1: PRN03
345
346
347
Elev/10 (deg.)
metres
time (s)

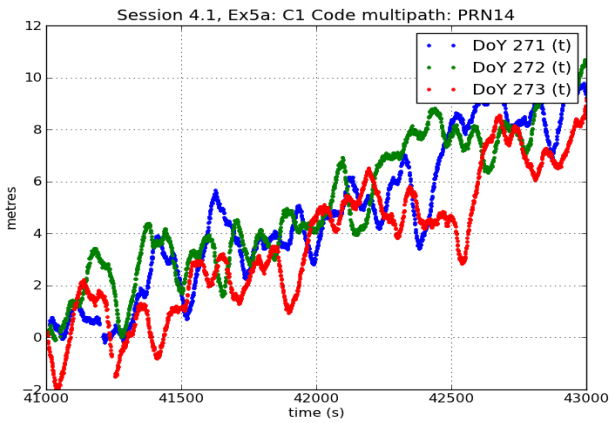

Session 4.1, Ex3a: PC code multipath: htv1: PRN03
345 (t)
346 (t+236)
347 (t+472)
Elev/10 (deg.)
metres
time (s)

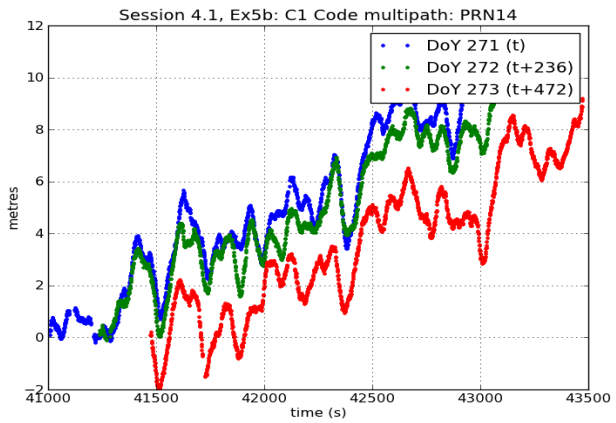

Session 4.1, Ex3b1: PC code multipath: galb: PRN03
345
346
347
Elev/10 (deg.)
metres
time (s)

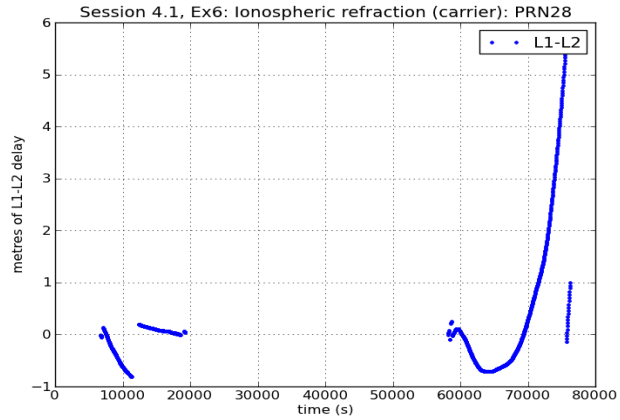

Session 4.1, Ex3b2: PC code multipath: galb: PRN03
345 (t)
346 (t+236)
347 (t+272)
Elev/10 (deg.)
metres
time (s)

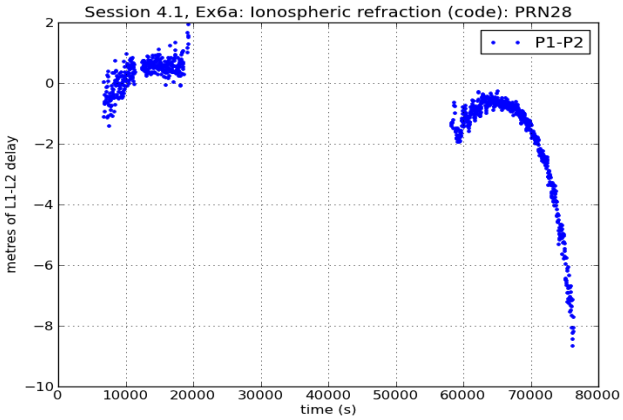

Session 4.1, Ex4a: Melbourne-Wubbena multipath: htv1: PRN03
345 (t)
346 (t+236)
347 (t+272)
Elev/10 (deg.)
metres
time (s)

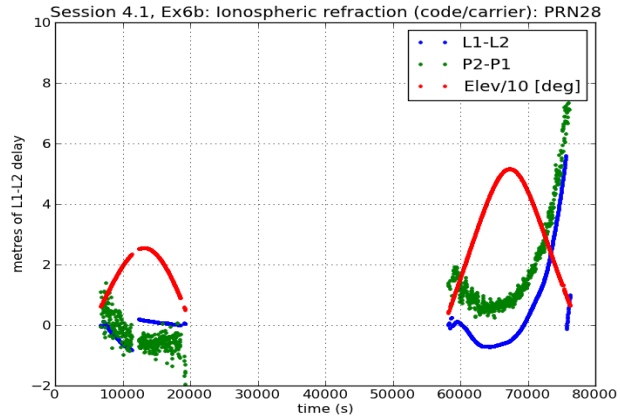

Session 4.1, Ex4b: Melbourne-Wubbena multipath: galb: PRN03
345 (t)
346 (t+236)
347 (t+272)
Elev/100 (deg.)
metres
time (s)

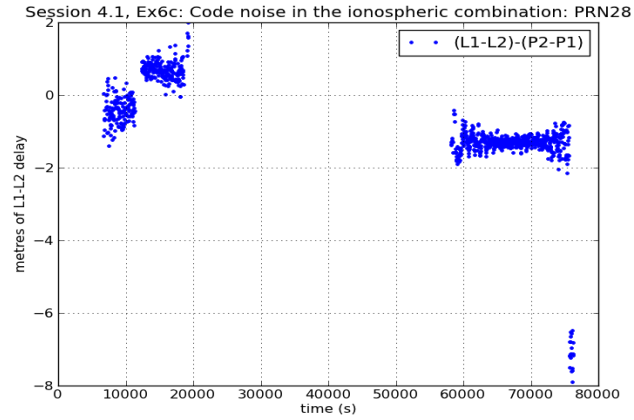

Session 4.1, Ex5a: C1 Code multipath: PRN14
DoY 271 (t)
DoY 272 (t)
DoY 273 (t)
metres
time (s)

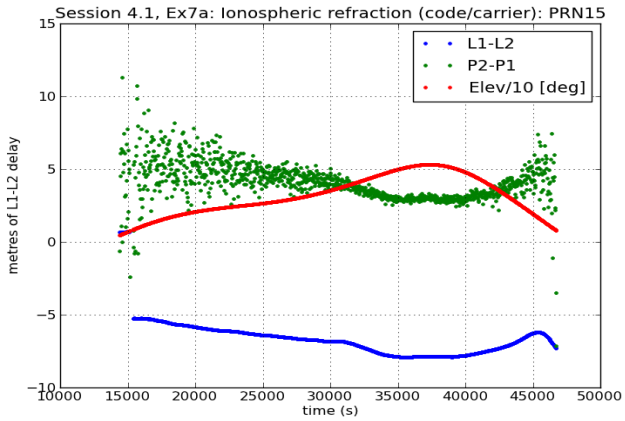

Session 4.1, Ex5b: C1 Code multipath: PRN14
DoY 271 (t)
DoY 272 (t+236)
DoY 273 (t+472)
metres
time (s)

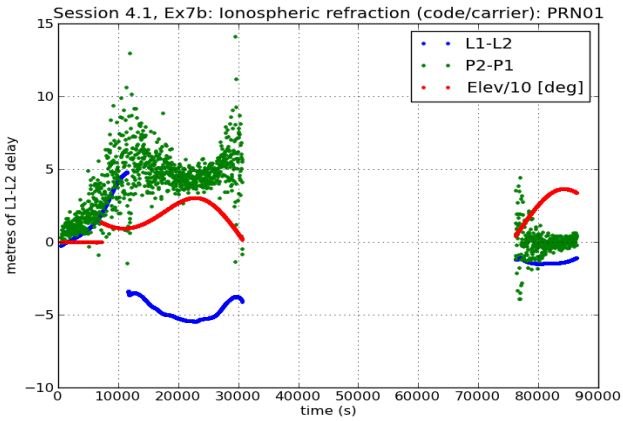

Session 4.1, Ex6: Ionospheric refraction (carrier): PRN28
L1-L2
metres of L1-L2 delay
time (s)

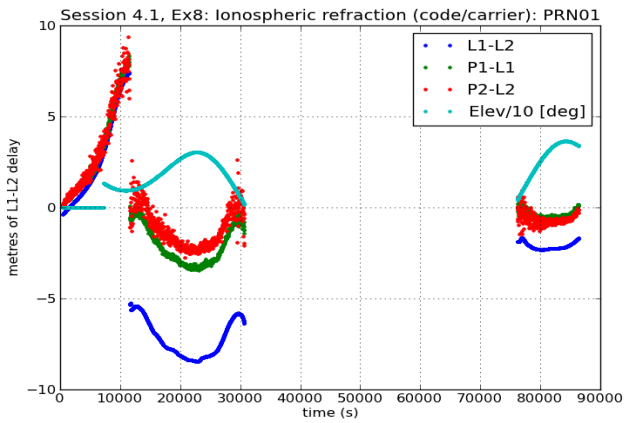

Session 4.1, Ex6a: Ionospheric refraction (code): PRN28
P1-P2
metres of L1-L2 delay
time (s)

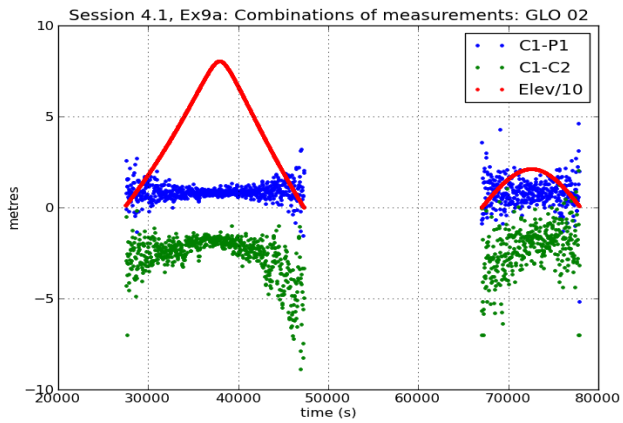

Session 4.1, Ex6b: Ionospheric refraction (code/carrier): PRN28
L1-L2
P2-P1
Elev/10 [deg]
metres of L1-L2 delay
time (s)

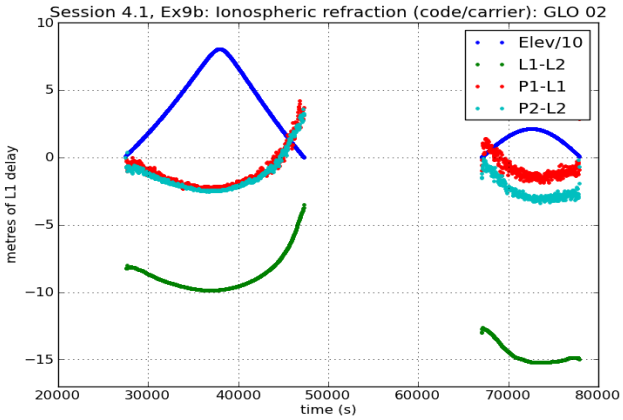

Session 4.1, Ex6c: Code noise in the ionospheric combination: PRN28
(L1-L2)-(P2-P1)
metres of L1-L2 delay
time (s)

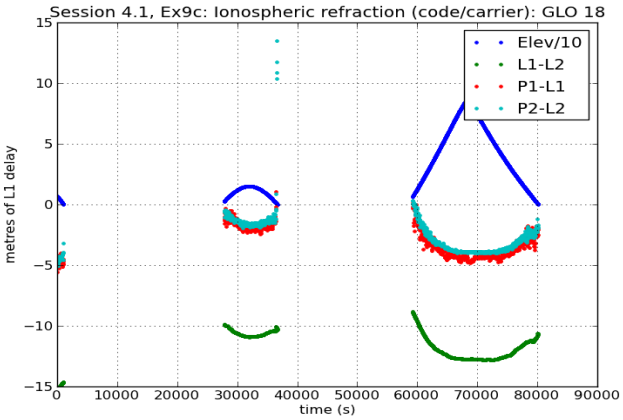

Session 4.1, Ex7a: Ionospheric refraction (code/carrier): PRN15
L1-L2
P2-P1
Elev/10 [deg]
metres of L1-L2 delay
time (s)

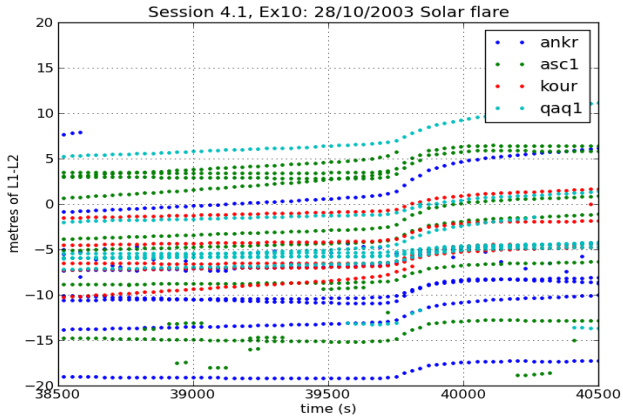

Session 4.1, Ex7b: Ionospheric refraction (code/carrier): PRN01
L1-L2
P2-P1
Elev/10 [deg]
metres of L1-L2 delay
time (s)

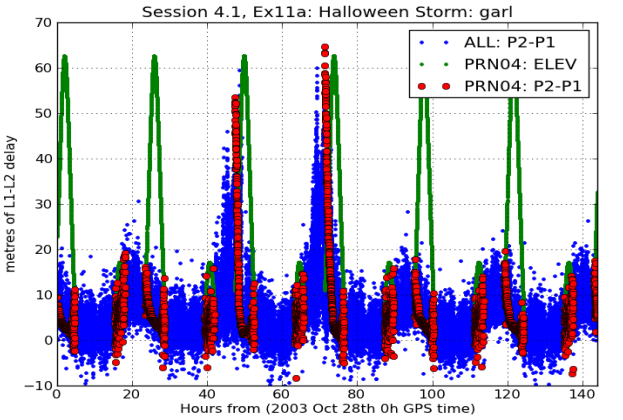

Session 4.1, Ex8: Ionospheric refraction (code/carrier): PRN01
L1-L2
P1-L1
P2-L2
Elev/10 [deg]
metres of L1-L2 delay
time (s)

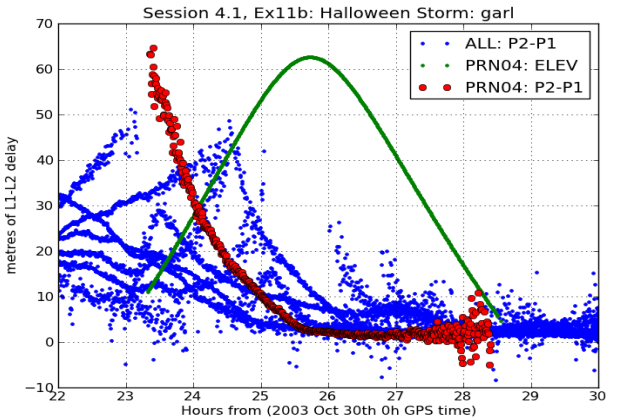

Session 4.1, Ex9a: Combinations of measurements: GLO 02
C1-P1
C1-C2
Elev/10
metres
time (s)

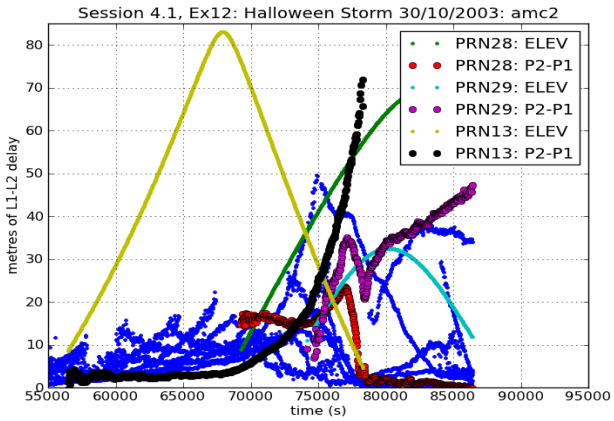

Session 4.1, Ex9b: Ionospheric refraction (code/carrier): GLO 02
Elev/10
L1-L2
P1-L1
P2-L2
metres of L1 delay
time (s)

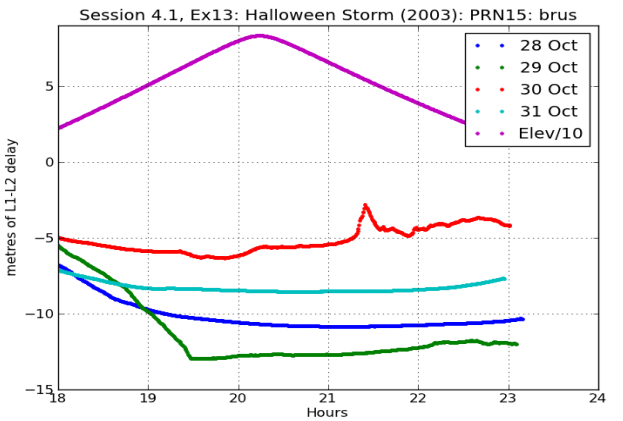

Session 4.1, Ex9c: Ionospheric refraction (code/carrier): GLO 18
Elev/10
L1-L2
P1-L1
P2-L2
metres of L1 delay
time (s)

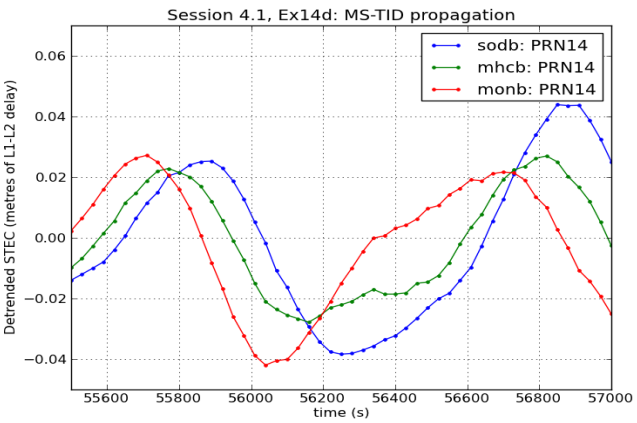

Session 4.1, Ex10: 28/10/2003 Solar flare
ankr
asc1
kour
qaq1
metres of L1-L2
time (s)

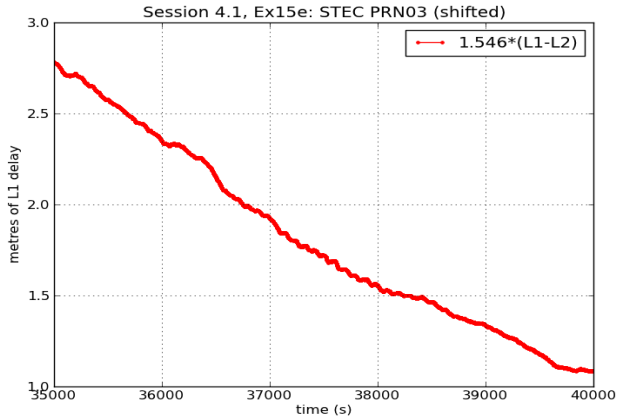

Session 4.1, Ex11a: Halloween Storm: garl
ALL: P2-P1
PRN04: ELEV
PRN04: P2-P1
metres of L1-L2 delay
Hours from (2003 Oct 28th 0h GPS time)

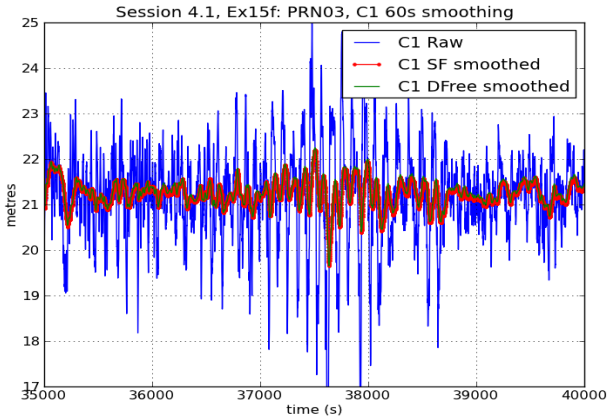

Session 4.1, Ex11b: Halloween Storm: garl
ALL: P2-P1
PRN04: ELEV
PRN04: P2-P1
metres of L1-L2 delay
Hours from (2003 Oct 30th 0h GPS time)

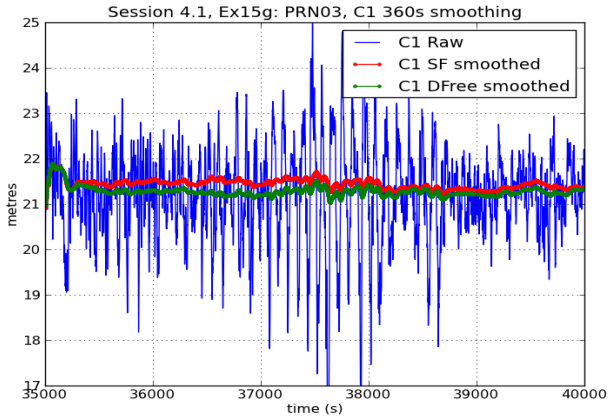

Session 4.1, Ex12: Halloween Storm 30/10/2003: amc2
PRN28: ELEV
PRN28: P2-P1
PRN29: ELEV
PRN29: P2-P1
PRN13: ELEV
PRN13: P2-P1
metres of L1-L2 delay
time (s)

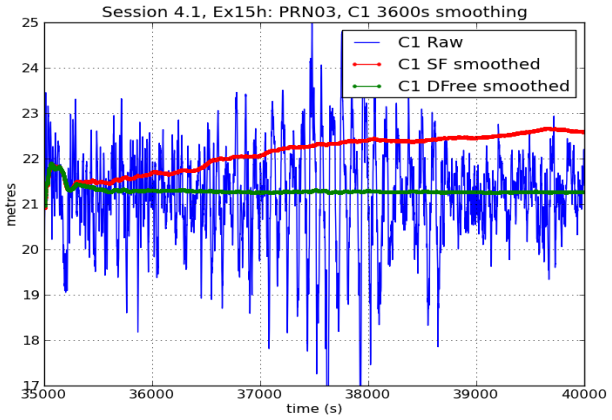

Session 4.1, Ex13: Halloween Storm (2003): PRN15: brus
28 Oct
29 Oct
30 Oct
31 Oct
Elev/10
metres of L1-L2 delay
Hours

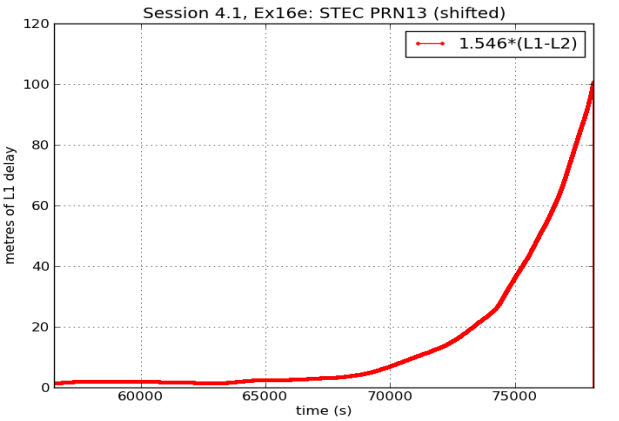

Session 4.1, Ex14d: MS-TID propagation
sodb: PRN14
mhcb: PRN14
monb: PRN14
Detrended STEC (metres of L1-L2 delay)
time (s)

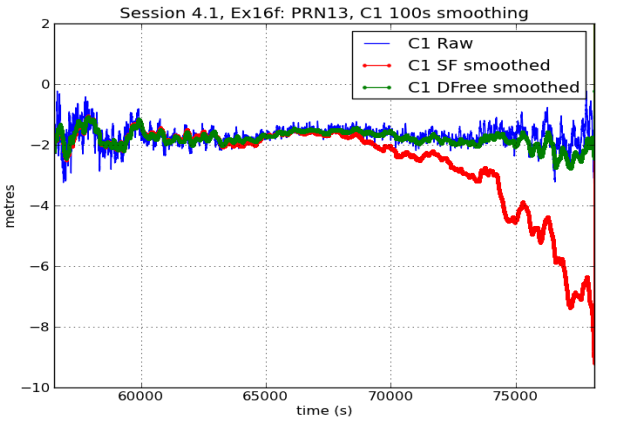

Session 4.1, Ex15e: STEC PRN03 (shifted)
1.546*(L1-L2)
metres of L1 delay
time (s)


Session 4.1, Ex15f: PRN03, C1 60s smoothing
C1 Raw
C1 SF smoothed
C1 DFree smoothed
metres
time (s)


Session 4.1, Ex15g: PRN03, C1 360s smoothing
C1 Raw
C1 SF smoothed
C1 DFree smoothed
metres
time (s)


Session 4.1, Ex15h: PRN03, C1 3600s smoothing
C1 Raw
C1 SF smoothed
C1 DFree smoothed
metres
time (s)


Session 4.1, Ex16e: STEC PRN13 (shifted)
1.546*(L1-L2)
metres of L1 delay
time (s)


Session 4.1, Ex16f: PRN13, C1 100s smoothing
C1 Raw
C1 SF smoothed
C1 DFree smoothed
metres
time (s)

Session 4.2. Code–Carrier Measurements and Combinations of Three-Frequency Signals

Objectives

To analyse graphically the three-frequency (Galileo and GPS) code and phase measurements and their combinations by comparing the different signals. To study their characteristics and properties: differential code biases, ionospheric refraction, multipath, receiver noise. To analyse the noise in the different combinations of signals.

Files to use
`gien327sw.09o`, `gcal327sw.09o`, `gcal330sw.09o`, `meas.cfg`, `orb15591.sp3`, `orb15594.sp3`, `l5dt1260.09o`, `l5dt1260.09n`

Programs to use
`gLAB_linux`, `graph.py`

Development

Session files:

Copy and uncompress these session files in the working directory:

```
cp ~/GNSS/PROG/SES42/* .
cp ~/GNSS/FILES/SES42/* .
gzip -d *.gz *.Z
```

1. Reading the RINEX files and skyplot
The RINEX-3.00 file `gien327sw.09o` contains Galileo (E1, E7, E8) and GPS (L1, L2) measurements collected by a permanent receiver. The associated orbits and clocks are given in the `orb15591.sp3` file.

(a) Following the procedure applied in session 4.1, generate a measurement file (MEAS) from such data files.
Execute for instance:

```
gLAB_linux -input:cfg meas.cfg -input:obs gien327sw.09o
                  -input:sp3 orb15591.sp3 > gien327.09.meas
```

i. How many Galileo satellites are tracked in this file? Which ones?
ii. What Galileo and GPS measurements are available? What are the associated frequencies?

(b) Produce a skyplot to visualise the satellite tracks and identify the nearest GPS satellite to the Galileo satellite PRN16, and the farthest one.

Execute for instance:

Decimate the data every 5 minutes, to avoid a big file:

```
gawk '{if ($4%300==0) print $4,$5,$6,$7,$8}' gien327.09.meas
                                                > gien327.09.tmp
```

Map the elevation and azimuth coordinates into a skyplot view:

```
gawk '{print $1,$2,$3,sin($5*3.14/180)*(90-$4)/90,
    cos($5*3.14/180)*(90-$4)/90}' gien327.09.tmp > gien327.09.elaz
```

Draw the skyplot. Select the satellites GAL (PRN16) and GPS (PRN04, 20):

```
graph.py -f gien327.09.elaz -x4 -y5 -s.  -l "All sat"
         -f gien327.09.elaz -x4 -y5 -so -c '($3==04)' -l "GPS 04"
         -f gien327.09.elaz -x4 -y5 -so -c '($3==20)' -l "GPS 20"
         -f gien327.09.elaz -x4 -y5 -so -c '($3==16)' -l "GAL 16"
                      --xn -1 --xx 1 --yn -1 --yx 1 -t "Sky plot"
```

2. Measurement file contents and DCBs

(a) Taking into account the MEAS file description of Table 4.1, verify the following field contents in the generated file `gien327.09.meas`:

Galileo measurements:

```
MEAS YY DoY sec GAL PRN el Az N. list C1B L1B C1C L1C C7Q L7Q C8Q L8Q
  1   2  3   4   5   6   7  8  9  10   11  12  13  14  15  16  17  18
```

GPS measurements:

```
MEAS YY DoY sec GPS PRN el Az N. list C1C L1C C1P L1P C2P L2P
  1   2  3   4   5   6   7  8  9  10   11  12  13  14  15  16
```

(b) Plot the DCBs between the C1B and C1C code measurements for the Galileo satellite PRN16 and between the C1C and C1P codes for GPS PRN04 and PRN20.

Execute for instance:

Computing the DCB:

```
gawk '{if ($5=="GAL" && $6==16) {print $4,$11-$13}}'
                  gien327.09.meas > gien327C1BC1C_16.dat
gawk '{if ($6==20) {print $4,$11-$13}}' gien327.09.meas
                                     > gien327C1CP1P_20.dat
gawk '{if ($6==04) {print $4,$11-$13}}' gien327.09.meas
                                     > gien327C1CP1P_04.dat
```

Plotting the results:

```
graph.py -f gien327C1BC1C_16.dat -l "GAL16:C1B-C1C"
             -f gien327C1CP1P_04.dat -l "GPS04:C1C-C1P"
             -f gien327C1CP1P_20.dat -l "GPS20:C1C-C1P"
                                          --yn -2 --yx 3
```

Compare the noise of Galileo and GPS measurements seen in the plot. Do Galileo and GPS data show similar noise levels?

3. Ionospheric refraction measured from Galileo and GPS data

Three ionospheric combinations can be computed from the Galileo measurements of file `gien327sw.09o` at frequencies E1, E7 and E8.[13] They corresponds to the three pairs [E1, E7], [E1, E8] and [E7, E8].

(a) Taking into account the equations and tables of section 4.1 in Volume I, verify the following expressions for computing the ionospheric refraction, in f_1 delay units, from the three possible combinations of code C1, C7, C8 and carrier E1, E7, E8 measurements:

Note: 'L' instead of 'E' will always be used in the formulae for GPS or Galileo carrier measurements to simplify the notation. Moreover, 'C' will always be used to indicate the code measurements, whether they correspond to the P or C codes. Finally, for historical reasons, combinations of codes will be written as PI and those of carriers as LI.

$$\begin{aligned} PI_{1_{[1,7]}} &= \tfrac{1}{\gamma_{1,7}-1}(C_7 - C_1) \simeq 1.42\,(C_7 - C_1) \\ LI_{1_{[1,7]}} &= \tfrac{1}{\gamma_{1,7}-1}(L_1 - L_7) \simeq 1.42\,(L_1 - L_7) \end{aligned} \tag{4.4}$$

$$\begin{aligned} PI_{1_{[1,8]}} &= \tfrac{1}{\gamma_{1,8}-1}(C_8 - C_1) \simeq 1.34\,(C_8 - C_1) \\ LI_{1_{[1,8]}} &= \tfrac{1}{\gamma_{1,8}-1}(L_1 - L_8) \simeq 1.34\,(L_1 - L_8) \end{aligned} \tag{4.5}$$

$$\begin{aligned} PI_{1_{[7,8]}} &= \tfrac{1/\gamma_{1,7}}{\gamma_{7,8}-1}(C_8 - C_7) \simeq 22.65\,(C_8 - C_7) \\ LI_{1_{[7,8]}} &= \tfrac{1/\gamma_{1,7}}{\gamma_{7,8}-1}(L_7 - L_8) \simeq 22.65\,(L_7 - L_8) \end{aligned} \tag{4.6}$$

where $\gamma_{i,j} = (f_i/f_j)^2$ is given in Table 4.2, Volume I.

Assuming uncorrelated measurements with equal σ noise, show that

$$\begin{aligned} &\sigma_{PI_{1_{[1,7]}}} \simeq 2.0\,\sigma_C, \quad \sigma_{PI_{1_{[1,8]}}} \simeq 1.9\,\sigma_C, \quad \sigma_{PI_{1_{[7,8]}}} \simeq 32.0\,\sigma_C \\ &\sigma_{LI_{1_{[1,7]}}} \simeq 2.0\,\sigma_L, \quad \sigma_{LI_{1_{[1,8]}}} \simeq 1.9\,\sigma_L, \quad \sigma_{LI_{1_{[7,8]}}} \simeq 32.0\,\sigma_L \end{aligned} \tag{4.7}$$

Which ionospheric combinations seem more suitable for computing the ionospheric refraction? Why?

(b) As in the previous case, verify the following expressions to compute the ionospheric refraction, in f_1 delay units, from GPS code C1, C2 and carrier L1, L2 measurements:

$$\begin{aligned} PI_{1_{[1,2]}} &= \tfrac{1}{\gamma_{1,2}-1}(C_2 - C_1) \simeq 1.55\,(C_2 - C_1) \\ LI_{1_{[1,2]}} &= \tfrac{1}{\gamma_{1,2}-1}(L_1 - L_2) \simeq 1.55\,(L_1 - L_2) \end{aligned} \tag{4.8}$$

Assuming uncorrelated measurements with equal σ noise, show that

$$\sigma_{PI_{1_{[1,2]}}} \simeq 2.2\ \sigma_C, \quad \sigma_{LI_{1_{[1,2]}}} \simeq 2.2\ \sigma_L \tag{4.9}$$

[13] This notation is from RINEX files for Galileo signals: E1→ f_{E1}, E5→ f_{E5a}, E6→ f_{E6}, E7→ f_{E5b}, E8→ f_{E5}.

(c) Plot the ionospheric refraction for the Galileo satellite PRN16 computed from the three possible pairs of combinations of code measurements. Compare and discuss the measurement noise seen.

Execute for instance:

Ionospheric combinations of code measurement computation (Galileo):

```
gawk 'BEGIN{g18=(154/116.5)^2;g17=(154/118)^2;g78=(118/116.5)^2}
        {if ($5=="GAL" && $6==16) {print $4,1/(g17-1)*($15-$11),
                1/(g18-1)*($17-$11),1/g17*1/(g78-1)*($17-$15)}}'
                                    gien327.09.meas >gien327Pi_16.dat
```

Plotting results:

```
graph.py -f gien327Pi_16.dat -x1 -y'($2+351)' -l "C7-C1"
                 -f gien327Pi_16.dat -x1 -y'($3+385)' -l "C8-C1"
                           -f gien327Pi_16.dat -x1 -y4 -l "C8-C7"
```

(d) Compare the ionospheric refraction of Galileo satellite PRN16 computed from the C7–C1 code combination with the ionospheric refraction of GPS satellites PRN04 and PRN20 computed from the P2–P1 code combination. Give the results in metres of delay at frequency f_1.

Execute for instance:

Ionospheric combination of code measurement computation (GPS):

```
gawk 'BEGIN{g12=(154/120)^2}
            {if ($6==20) {print $4,1/(g12-1)*($15-$13)}}'
                                    gien327.09.meas >gien327Pi_20.dat

gawk 'BEGIN{g12=(154/120)^2}
            {if ($6==04) {print $4,1/(g12-1)*($15-$13)}}'
                                   gien327.09.meas > gien327Pi_04.dat
```

Plotting results:

```
graph.py -f gien327Pi_04.dat -x1 -y2 -l "GPS04:P2-P1"
         -f gien327Pi_20.dat -x1 -y'($2-5)' -l "GPS20:P2-P1"
         -f gien327Pi_16.dat -x1 -y'($2+367)' -l "GAL16:C7-C1"
```

(e) Repeat the plots of section 3c, but using the Galileo carrier phase measurements E1, E7, E8 instead of the code ones.

Execute for instance:

Ionospheric combinations of carrier measurement computation (Galileo):

```
gawk 'BEGIN{g18=(154/116.5)^2;g17=(154/118)^2;g78=(118/116.5)^2}
        {if ($5=="GAL" && $6==16) {print $4,1/(g17-1)*($14-$16),
                1/(g18-1)*($12-$16),1/g17*1/(g78-1)*($16-$18)}}'
                                    gien327.09.meas >gien327Li_16.dat
```

Plotting results:

```
graph.py -f gien327Li_16.dat -x1 -y'($2-385)' -l "E1-E7"
        -f gien327Li_16.dat -x1 -y'($3-362.5)' -l "E1-E8"
        -f gien327Li_16.dat -x1 -y'($4-1.65)' -l "E7-E8"
        --yn -1 --yx 2.5 --xl "time (s)" --yl "metres L1 delay"
```

(f) Repeat the plots of section (3d), but using the carrier GPS phase measurements L1, L2 instead of the code ones.
Execute for instance:

Ionospheric combination of carrier measurement computation (GPS):

```
gawk 'BEGIN{g12=(154/120)^2}
           {if ($6==20) {print $4,1/(g12-1)*($12-$16)}}'
                              gien327.09.meas >gien327Li_20.dat

gawk 'BEGIN{g12=(154/120)^2}
           {if ($6==04) {print $4,1/(g12-1)*($12-$16)}}'
                             gien327.09.meas > gien327Li_04.dat
```

Plotting results:

```
graph.py -f gien327Li_04.dat -x1 -y'($2-6)' -l "GPS04:L1-L2"
        -f gien327Li_20.dat -x1 -y'($2-11)' -l "GPS20:L1-L2"
        -f gien327Li_16.dat -x1 -y'($2-385)' -l "GAL16:E1-E7"
        --yn -1 --yx 1.5 --xl "time (s)" --yl "metres L1 delay"
```

4. Ionosphere-free combinations for Galileo and GPS

The three-frequency Galileo measurements E1, E7 and E8 allow three ionosphere-free combinations to be computed from the three pairs of signals [E1,E7], [E1,E8] and [E7,E8].

(a) Taking into account the equations and tables of section 4.1 in Volume I, verify the following expressions for computing the ionosphere combination of code (C1, C7, C8) and carrier (L1, L7, L8) measurements:

$$\begin{aligned} PC_{[1,7]} &= \frac{f_1^2 C_1 - f_7^2 C_7}{f_1^2 - f_7^2} = \frac{\gamma_{1,7} C_1 - C_7}{\gamma_{1,7} - 1} \simeq C_1 - 1.42\,(C_7 - C_1) \\ LC_{[1,7]} &= \frac{f_1^2 L_1 - f_7^2 L_7}{f_1^2 - f_7^2} = \frac{\gamma_{1,7} L_1 - L_7}{\gamma_{1,7} - 1} \simeq L_1 + 1.42\,(L_1 - L_7) \end{aligned} \tag{4.10}$$

$$\begin{aligned} PC_{[1,8]} &= \frac{f_1^2 C_1 - f_8^2 C_8}{f_1^2 - f_8^2} = \frac{\gamma_{1,8} C_1 - C_8}{\gamma_{1,8} - 1} \simeq C_1 - 1.34\,(C_8 - C_1) \\ LC_{[1,8]} &= \frac{f_1^2 L_1 - f_8^2 L_8}{f_1^2 - f_8^2} = \frac{\gamma_{1,8} L_1 - L_8}{\gamma_{1,8} - 1} \simeq L_1 + 1.34\,(L_1 - L_8) \end{aligned} \tag{4.11}$$

$$\begin{aligned} PC_{[7,8]} &= \frac{f_7^2 C_7 - f_8^2 C_8}{f_7^2 - f_8^2} = \frac{\gamma_{7,8} C_7 - C_8}{\gamma_{7,8} - 1} \simeq C_7 - 39.58\,(C_7 - C_8) \\ LC_{[7,8]} &= \frac{f_7^2 L_7 - f_8^2 L_8}{f_7^2 - f_8^2} = \frac{\gamma_{7,8} L_7 - L_8}{\gamma_{7,8} - 1} \simeq L_7 + 39.58\,(L_8 - L_7) \end{aligned} \tag{4.12}$$

where $\gamma_{i,j} = (f_i/f_j)^2$ is given in Table 4.2, Volume I.

Assuming uncorrelated measurements with equal σ noise, show that

$$\begin{aligned} \sigma_{PC_{[1,7]}} &\simeq 2.81\,\sigma_C, \quad \sigma_{PC_{[1,8]}} \simeq 2.69\,\sigma_C, \quad \sigma_{PC_{[7,8]}} \simeq 55.28\,\sigma_C \\ \sigma_{LC_{[1,7]}} &\simeq 2.81\,\sigma_L, \quad \sigma_{LC_{[1,8]}} \simeq 2.69\,\sigma_L, \quad \sigma_{LC_{[7,8]}} \simeq 55.28\,\sigma_L \end{aligned} \tag{4.13}$$

Which combinations seem to be more suitable for navigation? Why?

(b) As in the previous case, verify the following expressions to compute the ionosphere-free combinations of GPS code C1, C2 and carrier L1, L2 measurements:

$$\begin{aligned} PC_{[1,2]} &= \frac{f_1^2 C_1 - f_2^2 C_2}{f_1^2 - f_2^2} = \frac{\gamma_{1,2} C_1 - C_2}{\gamma_{1,2} - 1} \simeq C_1 - 1.55\,(C_2 - C_1) \\ LC_{[1,2]} &= \frac{f_1^2 L_1 - f_2^2 L_2}{f_1^2 - f_2^2} = \frac{\gamma_{1,2} L_1 - L_2}{\gamma_{1,2} - 1} \simeq L_1 + 1.55\,(L_1 - L_2) \end{aligned} \tag{4.14}$$

Assuming uncorrelated measurements with equal σ noise, show that

$$\sigma_{PC_{[1,2]}} \simeq 2.98\,\sigma_C, \quad \sigma_{LC_{[1,2]}} \simeq 2.98\,\sigma_L \tag{4.15}$$

(c) Plot the differences of code and carrier ionosphere-free combinations PC–LC for the Galileo satellite computed from the three possible pairs of combinations of code measurements. Discuss the measurement noise patterns seen.

Execute for instance:

Ionospheric combinations of code measurement computation (Galileo):

```
gawk 'BEGIN{g18=(154/116.5)^2;g17=(154/118)^2;g78=(118/116.5)^2}
            {if ($5=="GAL" && $6==16) {print $4,(g17*($11-$12)-
          ($15-$16))/(g17-1),(g18*($11-$12)-($17-$18))/(g18-1),
           (g78*($15-$16)-($17-$18))/(g78-1)}}' gien327.09.meas
                                              > gien327PcLc_16.dat
```

Plotting results:

```
graph.py -f gien327PcLc_16.dat -x1 -y'($2+40)' -l "PC-LC[17]"
           -f gien327PcLc_16.dat -x1 -y'($3-40)' -l "PC-LC[18]"
                  -f gien327PcLc_16.dat -x1 -y4 -l "PC-LC[78]"
                --yn -80 --yx 80 --xl "time (s)" --yl "metres"
```

(d) Compare in a plot the differences of PC–LC for Galileo satellite PRN16, from f_1 and f_7 frequency signals, with PC–LC of GPS satellites PRN04 and PRN20 computed from frequencies f_1 and f_2. Discuss the measurement noise patterns seen.

Execute for instance:

Ionospheric combinations of code measurement computation (GPS):

```
gawk 'BEGIN{g12=(154/120)^2} {if ($6==20)
                  {print $4,(g12*($13-$15)-($12-$16))/(g12-1)}}'
                           gien327.09.meas > gien327PcLc_20.dat

gawk 'BEGIN{g12=(154/120)^2} {if ($6==04)
                  {print $4,(g12*($13-$15)-($12-$16))/(g12-1)}}'
                           gien327.09.meas > gien327PcLc_04.dat
```

Plotting results:

```
graph.py -f gien327PcLc_04.dat -x1 -y2 -l "GPS04:PC-LC[12]"
      -f gien327PcLc_20.dat -x1 -y'($2+11)' -l "GPS20:PC-LC[12]"
      -f gien327PcLc_16.dat -x1 -y'($2+20)' -l "GAL16:PC-LC[17]"
      --yn 0 --yx 30 --xl "time (s)" --yl "metres"
```

5. Melbourne–Wübbena combinations for Galileo and GPS

Three Melbourne–Wübbena combinations can be computed from the measurement file `gien327sw.09o`, using the three pairs of signals [E1,E7], [E1,E8] and [E7,E8].

(a) Taking into account the equations and tables of section 4.1 in Volume I, verify the following expressions for computing the Melbourne–Wübbena combinations of code (C1, C7, C8) and carrier (L1, L7, L8) measurements:

$$MW_{[1,7]} = \frac{f_1 L_1 - f_7 L_7}{f_1 - f_7} - \frac{f_1 C_1 + f_7 C_7}{f_1 + f_7} = \frac{\sqrt{\gamma_{1,7}} L_1 - L_7}{\sqrt{\gamma_{1,7}} - 1} - \frac{\sqrt{\gamma_{1,7}} C_1 + C_7}{\sqrt{\gamma_{1,7}} + 1}$$
$$\simeq 4.28 L_1 - 3.28 L_7 - (0.57 C_1 + 0.43 C_7) \qquad (4.16)$$

$$MW_{[1,8]} = \frac{f_1 L_1 - f_8 L_8}{f_1 - f_8} - \frac{f_1 C_1 + f_8 C_8}{f_1 + f_8} = \frac{\sqrt{\gamma_{1,8}} L_1 - L_8}{\sqrt{\gamma_{1,8}} - 1} - \frac{\sqrt{\gamma_{1,8}} C_1 + C_8}{\sqrt{\gamma_{1,8}} + 1}$$
$$\simeq 4.11 L_1 - 3.11 L_8 - (0.57 C_1 + 0.43 C_8) \qquad (4.17)$$

$$MW_{[7,8]} = \frac{f_7 L_7 - f_8 L_8}{f_7 - f_8} - \frac{f_7 C_7 + f_8 C_8}{f_7 + f_8} = \frac{\sqrt{\gamma_{7,8}} L_7 - L_8}{\sqrt{\gamma_{7,8}} - 1} - \frac{\sqrt{\gamma_{7,8}} C_7 + C_8}{\sqrt{\gamma_{7,8}} + 1}$$
$$\simeq 78.67 L_7 - 77.67 L_8 - (0.50 C_7 + 0.50 C_8) \qquad (4.18)$$

where $\gamma_{i,j} = (f_i/f_j)^2$ is given in Table 4.2, Volume I.

Assuming uncorrelated measurements with equal σ noise, show that

$$\sigma_{MW_{[1,7]}} \simeq \sigma_{MW_{[1,8]}} \simeq 0.71\,\sigma_C, \qquad \sigma_{MW_{[7,8]}} \simeq 0.76\,\sigma_C. \qquad (4.19)$$

where σ_C indicates the code measurement noise (notice that the σ_C noise is about two orders of magnitude higher than carrier noise σ_L). Discuss the advantages of these combinations to detect cycle slips, to fix ambiguities and other possible applications.

(b) As in the previous case, verify the following expressions for computing the Melbourne–Wübbena combinations with GPS code C1, C2 and carrier L1, L2 measurements:

$$MW_{[1,2]} = \frac{f_1 L_1 - f_2 L_2}{f_1 - f_2} - \frac{f_1 C_1 + f_2 C_2}{f_1 + f_2} = \frac{\sqrt{\gamma_{1,2}} L_1 - L_2}{\sqrt{\gamma_{1,2}} - 1} - \frac{\sqrt{\gamma_{1,2}} C_1 + C_2}{\sqrt{\gamma_{1,2}} + 1}$$

$$\simeq 4.53 L_1 - 3.53 L_2 - (0.56 C_1 + 0.44 C_2) \qquad (4.20)$$

Assuming uncorrelated measurements with equal σ noise, show that

$$\sigma_{MW_{[1,2]}} \simeq 0.71\, \sigma_C \qquad (4.21)$$

(c) Plot the Melbourne–Wübbena combination for the Galileo satellites computed from the three possible pairs of combinations of code measurements. Discuss the measurement noise patterns seen.

Execute for instance:

Computation of Melbourne–Wübbena combination (Galileo):

```
gawk 'BEGIN{s17=154/118;s18=154/116.5;s78=118/116.5}
     {if ($5=="GAL" && $6==16)
      {print $4,(s17*$12-$16)/(s17-1)-(s17*$11+$15)/(s17+1),
                (s18*$12-$18)/(s18-1)-(s18*$11+$17)/(s18+1),
                (s78*$16-$18)/(s78-1)-(s78*$15+$17)/(s78+1)}}'
                                gien327.09.meas >gien327MW_16.dat
```

Plotting results:

```
graph.py -f gien327MW_16.dat -x1 -y'($2-1007)' -l "MW[17]"
             -f gien327MW_16.dat -x1 -y'($3-957.5)' -l "MW[18]"
                 -f gien327MW_16.dat -x1 -y'($4-5)' -l "MW[78]"
                   --yn -4 --yx 4 --xl "time (s)" --yl "metres"
```

(d) Plot the Melbourne–Wübbena combination for Galileo satellite PRN16, from the f_1 and f_7 frequency signals, for the GPS satellites PRN04 and PRN20 computed from frequencies f_1 and f_2. Discuss the measurement noise patterns seen.

Execute for instance:[14]

Computation of Melbourne–Wübbena combination (GPS):

```
gawk 'BEGIN{c=299792458;f0=10.23e6;f1=154*f0;f2=120*f0}
            {if ($6==04) {Pw12=(f1*$13+f2*$15)/(f1+f2);
             Lw12=(f1*$12-f2*$16)/(f1-f2);print $4,Lw12-Pw12}}'
                                gien327.09.meas > gien327MW_04.dat

gawk 'BEGIN{c=299792458;f0=10.23e6;f1=154*f0;f2=120*f0}
            {if ($6==20) {Pw12=(f1*$13+f2*$15)/(f1+f2);
             Lw12=(f1*$12-f2*$16)/(f1-f2);print $4,Lw12-Pw12}}'
                                gien327.09.meas > gien327MW_20.dat
```

[14] Note that the expressions used here to compute the Melbourne–Wübbena combination are different than in the previous case, but equivalent; see equation (4.20).

Plotting results:

```
graph.py -f gien327MW_04.dat -x1 -y'($2-18)' -l "GPS04:MW[12]"
         -f gien327MW_20.dat  -x1 -y'($2-32)' -l "GPS20:MW[12]"
          -f gien327MW_16.dat  -x1 -y'($2-1005)' -l "GAL16:MW[17]"
                      --yn -7 --yx 5 --xl "time (s)" --yl "metres"
```

(e) Using expressions (4.19) and Table 4.2 in Volume I, compute the wavelength for the wide-lane combinations used in the previous exercises and repeat the plots, but using wide-lane cycles instead of metres.

Execute for instance:

```
gawk 'BEGIN {c=299792458; f0=10.23e6; f1=154*f0; f2=120*f0;
                      f7=118*f0; f8=116.5*f0; lambda_w12=c/(f1-f2);
  lambda_w17=c/(f1-f7); lambda_w18=c/(f1-f8); lambda_w78=c/(f7-f8);
            print lambda_w12, lambda_w17, lambda_w18, lambda_w78 }'
```

Repeat the previous plots, after dividing the values by their corresponding wavelengths.

6. Measurement noise and multipath for Galileo and GPS data

The code measurement noise and its multipath can be assessed by subtracting a combination of carriers at different frequencies from the code, in order to remove the ionospheric delay and the other non-frequency-dependent effects such as the geometric range, clocks and tropospheric delay. Different expressions to perform the computation are given as follows:

(a) Taking into account the equations and tables of section 4.1 in Volume I, verify the following expressions for depicting the code (C1, C7, C8) measurement noise and multipath:

$$\begin{aligned}
\mathcal{M}_{C1} &: C_1 - L_1 - \tfrac{2}{\gamma_{1,7}-1}(L_1 - L_7) \simeq C_1 - L_1 - 2.84\,(L_1 - L_7) \\
\mathcal{M}_{C7} &: C_7 - L_7 - \tfrac{2\gamma_{1,7}}{\gamma_{1,7}-1}(L_1 - L_7) \simeq C_7 - L_7 - 4.84\,(L_1 - L_7) \\
\mathcal{M}_{C8} &: C_8 - L_8 - \tfrac{2\gamma_{1,8}}{\gamma_{1,7}-1}(L_1 - L_7) \simeq C_8 - L_7 - 4.96\,(L_1 - L_7)
\end{aligned} \tag{4.22}$$

$$\begin{aligned}
\mathcal{M}_{C1} &: C_1 - L_1 - \tfrac{2/\gamma_{1,7}}{\gamma_{7,8}-1}(L_7 - L_8) \simeq C_1 - L_1 - 45.31\,(L_7 - L_8) \\
\mathcal{M}_{C7} &: C_7 - L_7 - \tfrac{2}{\gamma_{7,8}-1}(L_7 - L_8) \simeq C_7 - L_7 - 77.17\,(L_7 - L_8) \\
\mathcal{M}_{C8} &: C_8 - L_8 - \tfrac{2\gamma_{7,8}}{\gamma_{7,8}-1}(L_7 - L_8) \simeq C_8 - L_8 - 79.17\,(L_7 - L_8)
\end{aligned} \tag{4.23}$$

(b) Using expressions (4.22) produce a plot to visualise the measurement noise and multipath for each (C1, C7, C8) code. Add the satellite's elevation to the plot.

Execute for instance:

Code measurement noise and multipath computation:

```
gawk 'BEGIN{g17=(154/118)^2;g18=(154/116.5)^2}
      {if ($5=="GAL" && $6==16)
         {print $4,$11-$12-2/(g17-1)*($12-$16),
              $15-$16-2*g17/(g17-1)*($12-$16),
                      $17-$18-2*g18/(g17-1)*($12-$16),$7}}'
                       gien327.09.meas > gien327P178_16.dat
```

Plotting results:

```
graph.py -f gien327P178_16.dat -x1 -y'($2+768)' -l "[C1B]"
         -f gien327P178_16.dat -x1 -y'($3+1312)' -l "[C7Q]"
         -f gien327P178_16.dat -x1 -y'($4+1342)' -l "[C8Q]"
         -f gien327P178_16.dat -x1 -y'($5/10)' -l "Elev/10"
               --yn -6 --yx 8 --xl "time (s)" --yl "metres"
```

(c) Repeat the previous plots, but using expressions (4.23). Are there differences in the plots? Why?

Execute for instance:

Code measurement noise and multipath computation:

```
gawk 'BEGIN{g17=(154/118)^2;g78=(118/116.5)^2}
      {if ($5=="GAL" && $6==16)
         {print $4,$11-$12-2/g17/(g78-1)*($16-$18),
              $15-$16-2/(g78-1)*($16-$18),
                      $17-$18-2*g78/(g78-1)*($16-$18),$7}}'
                       gien327.09.meas >gien327P178b_16.dat
```

Plotting results:

```
graph.py -f gien327P178b_16.dat -x1 -y2 -l "[C1B]"
         -f gien327P178b_16.dat -x1 -y'($3+5.2)' -l "[C7Q]"
                  -f gien327P178b_16.dat -x1 -y4 -l "[C8Q]"
        -f gien327P178b_16.dat -x1 -y'($5/10)' -l "Elev/10"
               --yn -6 --yx 8 --xl "time (s)" --yl "metres"
```

(d) Verify the following expression for computing the code C1 measurement noise and multipath for the GPS measurements:

$$\mathcal{M}_{C1} : C_1 - L_1 - \frac{2}{\gamma_{1,2}-1}(L_1 - L_2) \simeq C_1 - L_1 - 3.09\,(L_1 - L_2) \quad (4.24)$$

(e) Compare the multipath and noise of the Galileo PRN16 C1B code and GPS satellites PRN04 and PRN20 C1P and C1C codes.

Execute for instance:

Combination computation of code and phase measurements (GPS sat.):

```
gawk 'BEGIN{g12=(154/120)^2}{if ($6==04) {print $4,$13-$12-
          2/(g12-1)*($12-$16),$11-$12-2/(g12-1)*($12-$16)}}'
                              gien327.09.meas >gien327P12_04.dat

gawk 'BEGIN{g12=(154/120)^2}{if ($6==20) {print $4,$13-$12-
            2/(g12-1)*($12-$16),$11-$12-2/(g12-1)*($12-$16)}}'
                              gien327.09.meas >gien327P12_20.dat
```

Plotting results GPS [C1P], Galileo [C1B]:

```
graph.py -f gien327P12_04.dat -x1 -y'($2+15.5)' -l "GPS04[C1P]"
          -f gien327P12_20.dat -x1 -y'($2+23)' -l "GPS20[C1P]"
         -f gien327P178_16.dat -x1 -y'($2+769)' -l "GAL16[C1B]"
        --yn -4 --yx 7 --xl "time (s)" --yl "metres"
```

Plotting results GPS [C1C], Galileo [C1B]:

```
graph.py -f gien327P12_04.dat -x1 -y'($3+15.5)' -l "GPS04[C1C]"
          -f gien327P12_20.dat -x1 -y'($3+22)' -l "GPS20[C1C]"
         -f gien327P178_16.dat -x1 -y'($2+769)' -l "GAL16[C1B]"
         --yn -4 --yx 7 --xl "time (s)" --yl "metres"
```

(f) Compare the multipath and the noise of the Galileo PRN16 C8Q (AltBOC) code with the Galileo PRN16 C1B and the GPS satellite PRN04 C1C codes.
Execute for instance:

Plotting results GPS [C1C], Galileo C8Q (AltBOC) and [C1B]:

```
graph.py -f gien327P12_04.dat -x1 -y'($3+14)' -l "GPS04[C1C]"
         -f gien327P178b_16.dat -x1 -y'($2+2)' -l "GAL16[C1B]"
         -f gien327P178b_16.dat -x1 -y'($4+2)' -l "GAL16 [C8Q]"
         -f gien327P178_16.dat -x1 -y'($5/10)' -l "Elev/10"
         --yn -4 --yx 8 --xl "time (s)" --yl "metres"
```

Which signal has lower code multipath and noise? Why? What is the level of noise of the Galileo PRN16 C8Q (AltBOC) signal compared with the Galileo C1B and GPS C1C?

7. Other examples with Galileo E1, E7, E5 signals

Repeat all the previous exercises using the file gcal327sw.09o with Galileo measurements at frequencies E1, E7 and E5.[15] The corresponding orbit and clock files orb15591.sp3 as in the previous exercises can be used.

(a) Read the RINEX files.
Execute for instance:

```
gLAB_linux -input:cfg meas.cfg -input:obs gcal327sw.09o
              -input:sp3 orb15591.sp3 > gcal327.09.meas
```

[15]This notation is from RINEX files for Galileo signals: E1→ f_{E1}, E5→ f_{E5a}, E6→ f_{E6}, E7→ f_{E5b}, E8→ f_{E5}.

(b) Produce the skyplot.

Execute for instance:

Decimate the data every 5 minutes, to avoid a big file:

```
gawk '{if ($4%300==0) print $4,$5,$6,$7,$8}' gcal327.09.meas
                                                    > gcal327.09.tmp
```

Map the elevation and azimuth coordinates to the skyplot view:

```
gawk '{print $1,$2,$3,sin($5*3.14/180)*(90-$4)/90,
       cos($5*3.14/180)*(90-$4)/90}' gcal327.09.tmp > gcal327.09.elaz
```

Plot the results for satellites GAL (PRN16) and GPS (PRN04, 20)

```
graph.py -f gcal327.09.elaz -x4 -y5 -s.  -l "All sat"
          -f gcal327.09.elaz -x4 -y5 -so -c '($3==02)' -l "GPS 02"
           -f gcal327.09.elaz -x4 -y5 -so -c '($3==04)' -l "GPS 04"
            -f gcal327.09.elaz -x4 -y5 -so -c '($3==16)' -l "GAL 16"
                          --xn -1 --xx 1 --yn -1 --yx 1 -t "Sky plot"
```

(c) Taking into account the MEAS file description of Table 4.1, verify the following field contents in the generated file `gcal327.09.meas`:

Galileo measurements:

```
MEAS YY DoY sec GAL PRN el Az N. list C1B L1B C1C L1C C5Q L5Q C7I L7I
  1   2  3   4   5   6   7  8  9  10   11  12  13  14  15  16  17  18
```

GPS measurements:

```
MEAS YY DoY sec GPS PRN el Az N. list C1C L1C  C2P L2P
  1   2  3   4   5   6   7  8  9  10   11  12  13  14
```

(d) Plot the interfrequency biases between C1B and C1C for the Galileo satellite PRN16.

(e) Verify the following expressions and plot the ionospheric refraction computed from the different combination of signals:

$$\begin{aligned} PI_{1_{[1,7]}} &= \tfrac{1}{\gamma_{1,7}-1}(C_7 - C_1) \simeq 1.42\,(C_7 - C_1) \\ LI_{1_{[1,7]}} &= \tfrac{1}{\gamma_{1,7}-1}(L_1 - L_7) \simeq 1.42\,(L_1 - L_7) \end{aligned} \tag{4.25}$$

$$\begin{aligned} PI_{1_{[1,5]}} &= \tfrac{1}{\gamma_{1,5}-1}(C_5 - C_1) \simeq 1.26\,(C_5 - C_1) \\ LI_{1_{[1,5]}} &= \tfrac{1}{\gamma_{1,5}-1}(L_1 - L_5) \simeq 1.26\,(L_1 - L_5) \end{aligned} \tag{4.26}$$

$$\begin{aligned} PI_{1_{[7,5]}} &= \tfrac{1/\gamma_{1,7}}{\gamma_{7,5}-1}(C_5 - C_7) \simeq 10.55\,(C_5 - C_7) \\ LI_{1_{[7,5]}} &= \tfrac{1/\gamma_{1,7}}{\gamma_{7,5}-1}(L_7 - L_5) \simeq 10.55\,(L_7 - L_5) \end{aligned} \tag{4.27}$$

(f) Verify the following expressions and draw the plots to depict the noise of the ionosphere-free combination code measurements for the different combinations of pairs of signals:

$$\begin{aligned} PC_{[1,7]} &= \tfrac{f_1^2 C_1 - f_7^2 C_7}{f_1^2 - f_7^2} = \tfrac{\gamma_{1,7} C_1 - C_7}{\gamma_{1,7}-1} \simeq C_1 - 1.42\,(C_7 - C_1) \\ LC_{[1,7]} &= \tfrac{f_1^2 L_1 - f_7^2 L_7}{f_1^2 - f_7^2} = \tfrac{\gamma_{1,7} L_1 - L_7}{\gamma_{1,7}-1} \simeq L_1 + 1.42\,(L_1 - L_7) \end{aligned} \tag{4.28}$$

$$\begin{aligned} PC_{[1,5]} &= \tfrac{f_1^2 C_1 - f_5^2 C_5}{f_1^2 - f_5^2} = \tfrac{\gamma_{1,5} C_1 - C_5}{\gamma_{1,5} - 1} \simeq C_1 - 1.26\,(C_5 - C_1) \\ LC_{[1,5]} &= \tfrac{f_1^2 L_1 - f_5^2 L_5}{f_1^2 - f_5^2} = \tfrac{\gamma_{1,5} L_1 - L_5}{\gamma_{1,5} - 1} \simeq L_1 + 1.26\,(L_1 - L_5) \end{aligned} \tag{4.29}$$

$$\begin{aligned} PC_{[7,5]} &= \tfrac{f_7^2 C_7 - f_5^2 C_5}{f_7^2 - f_5^2} = \tfrac{\gamma_{7,5} C_7 - C_5}{\gamma_{7,5} - 1} \simeq C_7 - 18.92\,(C_7 - C_5) \\ LC_{[7,5]} &= \tfrac{f_7^2 L_7 - f_5^2 L_5}{f_7^2 - f_5^2} = \tfrac{\gamma_{7,5} L_7 - L_5}{\gamma_{7,5} - 1} \simeq L_7 + 18.92\,(L_5 - L_7) \end{aligned} \tag{4.30}$$

(g) Verify the following expressions and draw the plots to analyse the performance of the different Melbourne–Wübbena combinations:

$$\begin{aligned} MW_{[1,7]} &= \tfrac{f_1 L_1 - f_7 L_7}{f_1 - f_7} - \tfrac{f_1 C_1 + f_7 C_7}{f_1 + f_7} = \tfrac{\sqrt{\gamma_{1,7}}\, L_1 - L_7}{\sqrt{\gamma_{1,7}} - 1} - \tfrac{\sqrt{\gamma_{1,7}}\, C_1 + C_7}{\sqrt{\gamma_{1,7}} + 1} \\ &\simeq 4.28 L_1 - 3.28 L_7 - (0.57 C_1 + 0.43 C_7) \end{aligned} \tag{4.31}$$

$$\begin{aligned} MW_{[1,5]} &= \tfrac{f_1 L_1 - f_5 L_5}{f_1 - f_5} - \tfrac{f_1 C_1 + f_5 C_5}{f_1 + f_5} = \tfrac{\sqrt{\gamma_{1,5}}\, L_1 - L_5}{\sqrt{\gamma_{1,5}} - 1} - \tfrac{\sqrt{\gamma_{1,5}}\, C_1 + C_5}{\sqrt{\gamma_{1,5}} + 1} \\ &\simeq 3.95 L_1 - 2.95 L_5 - (0.57 C_1 + 0.43 C_5) \end{aligned} \tag{4.32}$$

$$\begin{aligned} MW_{[7,5]} &= \tfrac{f_7 L_7 - f_5 L_5}{f_7 - f_5} - \tfrac{f_7 C_7 + f_5 C_5}{f_7 + f_5} = \tfrac{\sqrt{\gamma_{7,5}}\, L_7 - L_5}{\sqrt{\gamma_{7,5}} - 1} - \tfrac{\sqrt{\gamma_{7,5}}\, C_7 + C_5}{\sqrt{\gamma_{7,5}} + 1} \\ &\simeq 39.33 L_7 - 38.33 L_5 - (0.51 C_7 + 0.49 C_5) \end{aligned} \tag{4.33}$$

(h) Verify the following expressions and draw the plots to visualise the code measurement noise and multipath for the different signals:

$$\begin{aligned} \mathcal{M}_{C1} &: C_1 - L_1 - \tfrac{2}{\gamma_{1,7} - 1}(L_1 - L_7) \simeq C_1 - L_1 - 2.84\,(L_1 - L_7) \\ \mathcal{M}_{C7} &: C_7 - L_7 - \tfrac{2\gamma_{1,7}}{\gamma_{1,7} - 1}(L_1 - L_7) \simeq C_7 - L_7 - 4.84\,(L_1 - L_7) \\ \mathcal{M}_{C5} &: C5 - L5 - \tfrac{2\gamma_{1,5}}{\gamma_{1,7} - 1}(L_1 - L_7) \simeq C_5 - L_7 - 4.52\,(L_1 - L_7) \end{aligned} \tag{4.34}$$

$$\begin{aligned} \mathcal{M}_{C_1} &: C_1 - L_1 - \tfrac{2/\gamma_{1,7}}{\gamma_{7,5} - 1}(L_7 - L_5) \simeq C_1 - L_1 - 22.12\,(L_7 - L_5) \\ \mathcal{M}_{C_7} &: C_7 - L_7 - \tfrac{2}{\gamma_{7,5} - 1}(L_7 - L_5) \simeq C_7 - L_7 - 37.84\,(L_7 - L_5) \\ \mathcal{M}_{C_5} &: C_5 - L_5 - \tfrac{2\gamma_{7,5}}{\gamma_{7,5} - 1}(L_7 - L_5) \simeq C_5 - L_5 - 39.84\,(L_7 - L_5) \end{aligned} \tag{4.35}$$

8. Anomaly investigation

A large noise pattern appeared in the previous exercise when plotting the Galileo measurements of file `gcal327sw.09o` between $75\,000 < t < 76\,000$ seconds of day. This exercise is aimed at analysing this anomaly.

The following procedure is proposed:

(a) Read RINEX file `gcal327sw.09o` with `gLAB`, using the orbit file `orb15591.sp3`, and compile the measurement file `gcal327.09.meas`. (This item can be skipped if it has been done in the previous exercise.)

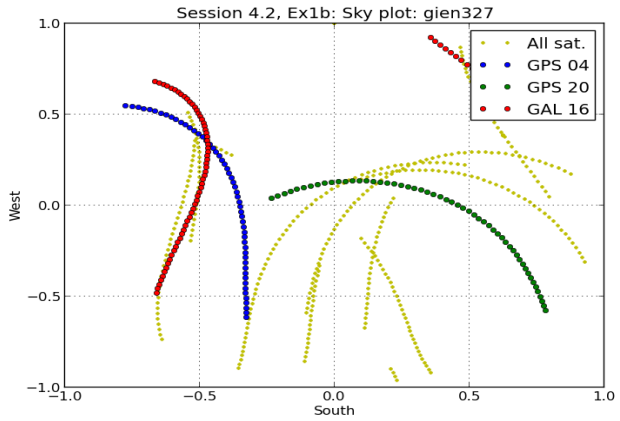
(b) Compute and plot the ionosphere-free combination for the Galileo satellite PRN16. Check the three possible pairs of combinations of frequencies E1, E7, E5 available in the file.

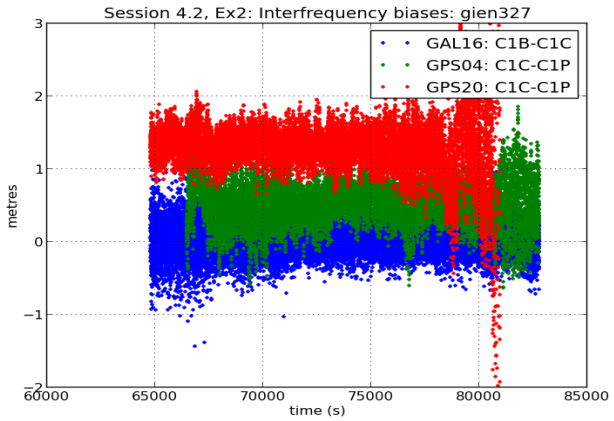
(c) Check if the same anomaly is also seen in GPS satellite measurements close to the Galileo satellite when this event happens. A skyplot will allow identification of the closest GPS satellites.

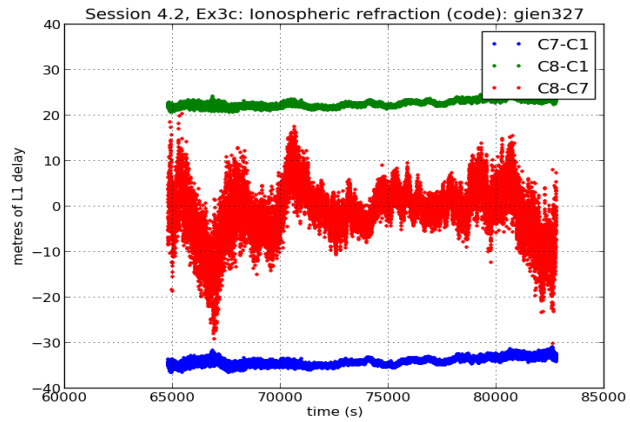
(d) Repeat the previous analysis using the measurement and orbit files of DoY 330 (three days after), `gcal330sw.09o` and `orb15594.sp3`.

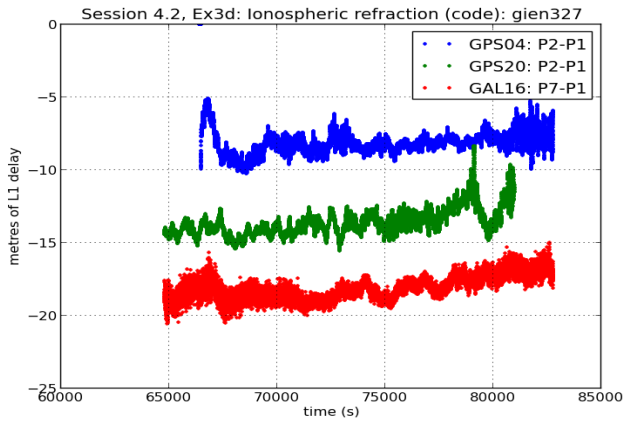
(e) Discuss the possible source of this anomaly.

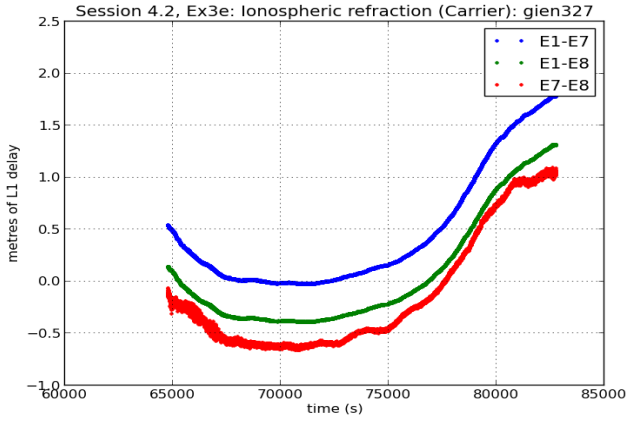
Graphs Session 4.2

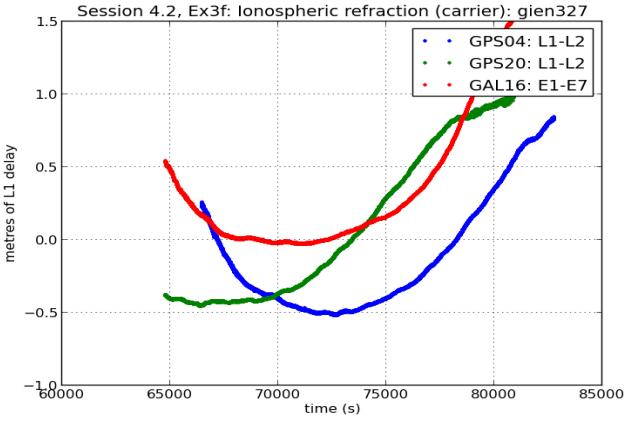


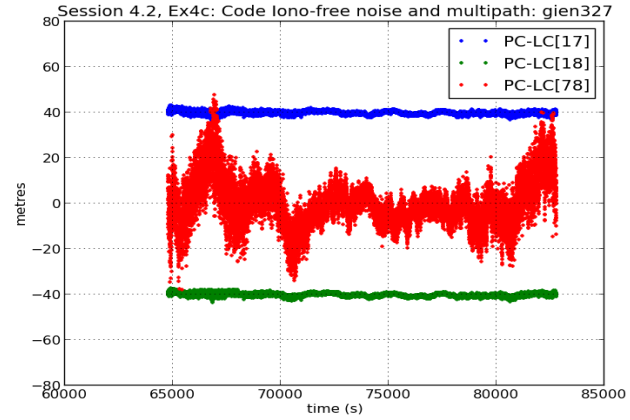


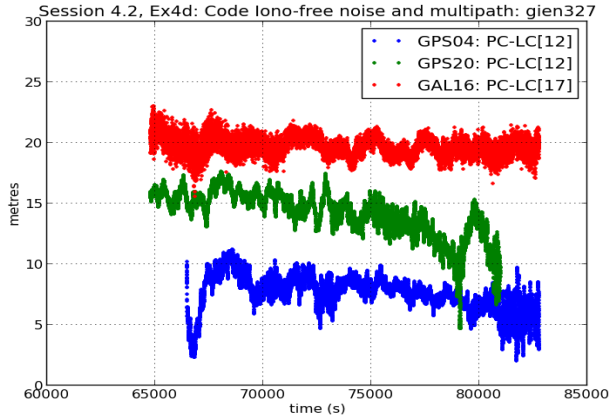


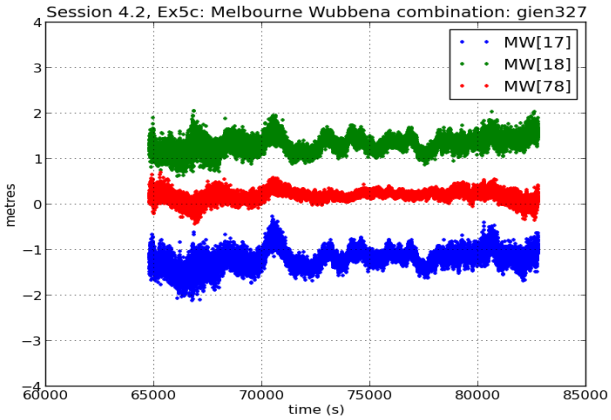


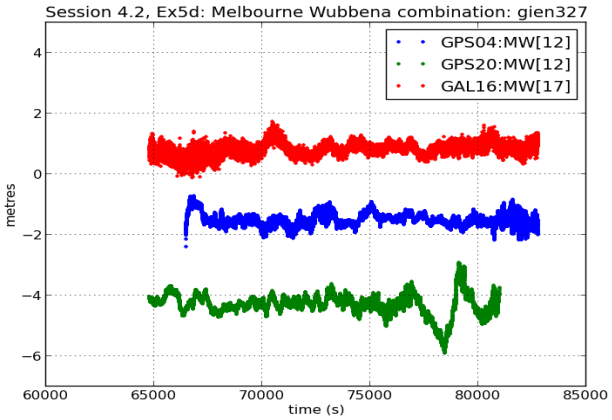


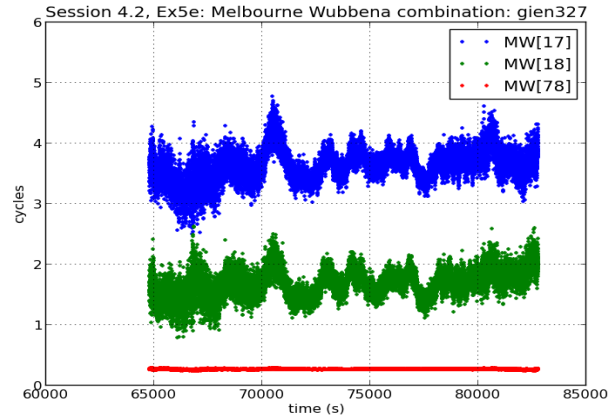


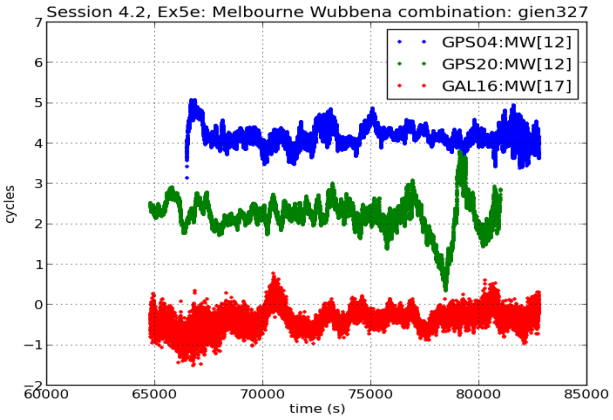


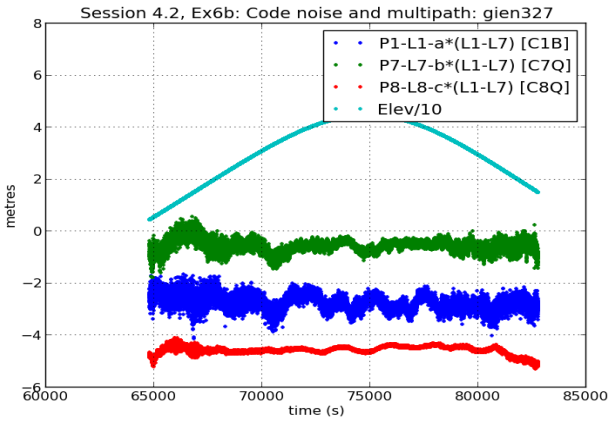


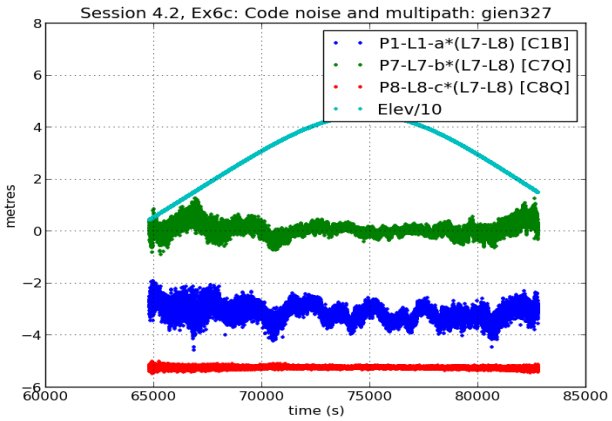

Session 4.2, Ex5c: Melbourne Wubbena combination: gien327
MW[17]
MW[18]
MW[78]
metres
time (s)

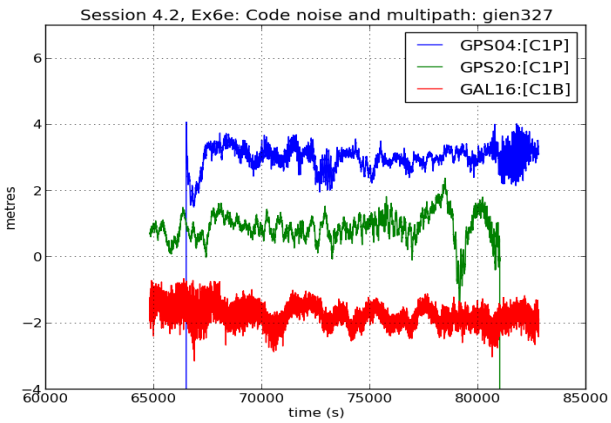

Session 4.2, Ex5d: Melbourne Wubbena combination: gien327
GPS04:MW[12]
GPS20:MW[12]
GAL16:MW[17]
metres
time (s)

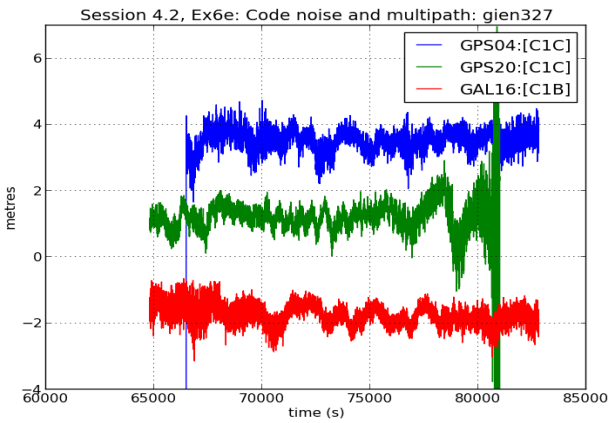

Session 4.2, Ex5e: Melbourne Wubbena combination: gien327
MW[17]
MW[18]
MW[78]
cycles
time (s)

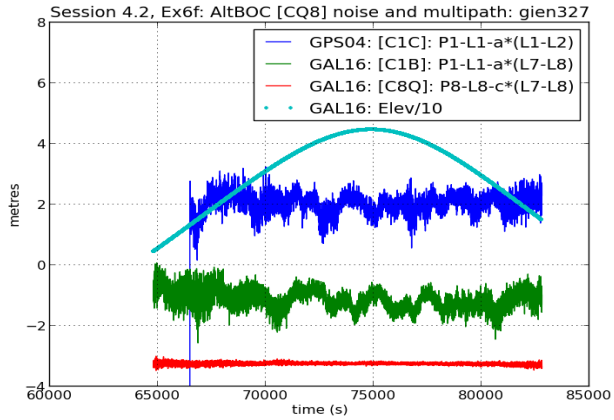

Session 4.2, Ex5e: Melbourne Wubbena combination: gien327
GPS04:MW[12]
GPS20:MW[12]
GAL16:MW[17]
cycles
time (s)

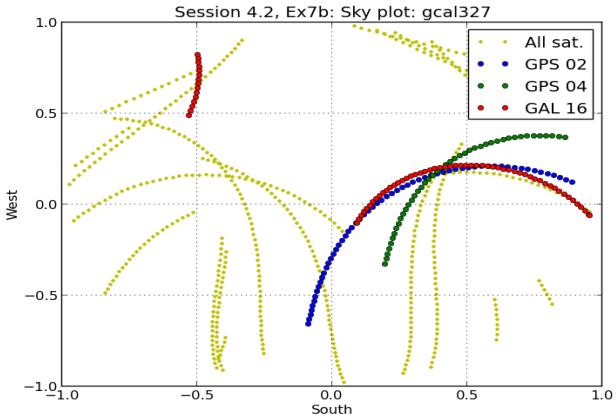

Session 4.2, Ex6b: Code noise and multipath: gien327
P1-L1-a*(L1-L7) [C1B]
P7-L7-b*(L1-L7) [C7Q]
P8-L8-c*(L1-L7) [C8Q]
Elev/10
metres
time (s)

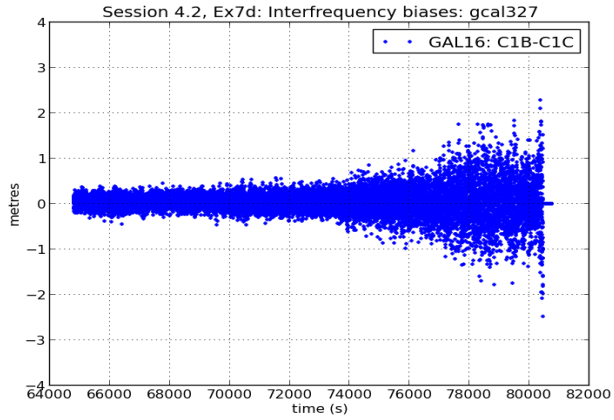

Session 4.2, Ex6c: Code noise and multipath: gien327
P1-L1-a*(L7-L8) [C1B]
P7-L7-b*(L7-L8) [C7Q]
P8-L8-c*(L7-L8) [C8Q]
Elev/10
metres
time (s)

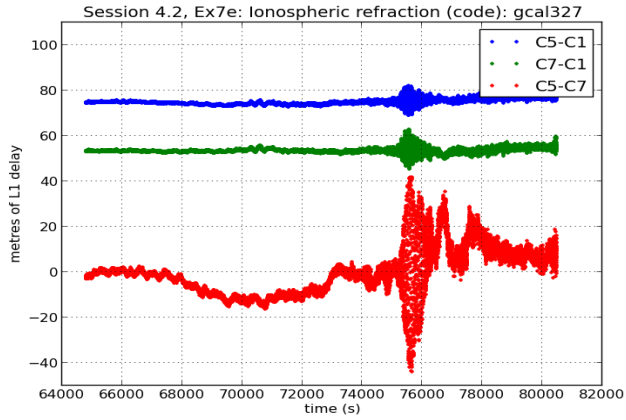

Session 4.2, Ex6e: Code noise and multipath: gien327
GPS04:[C1P]
GPS20:[C1P]
GAL16:[C1B]
metres
time (s)

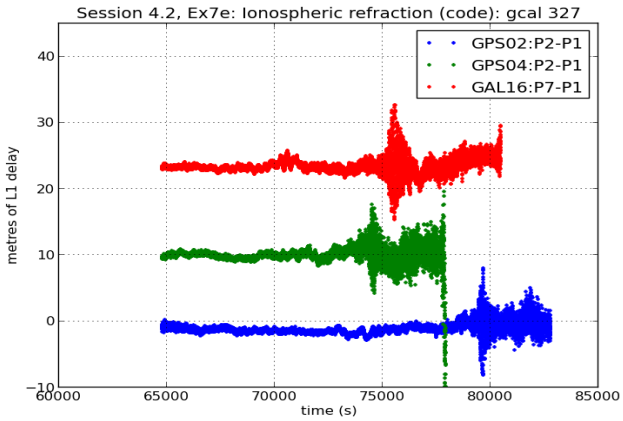

Session 4.2, Ex6e: Code noise and multipath: gien327
GPS04:[C1C]
GPS20:[C1C]
GAL16:[C1B]
metres
time (s)

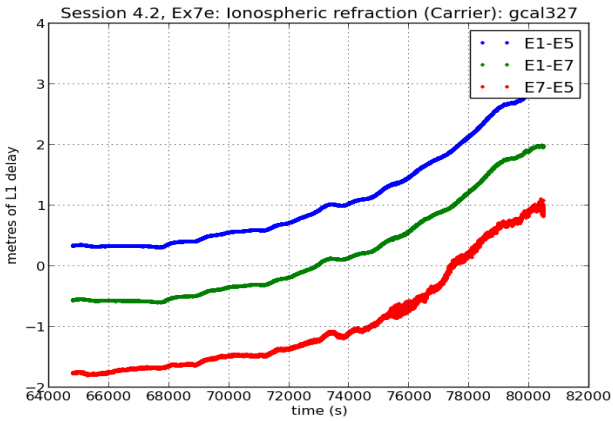

Session 4.2, Ex6f: AltBOC [CQ8] noise and multipath: gien327
GPS04: [C1C]: P1-L1-a*(L1-L2)
GAL16: [C1B]: P1-L1-a*(L7-L8)
GAL16: [C8Q]: P8-L8-c*(L7-L8)
GAL16: Elev/10
metres
time (s)

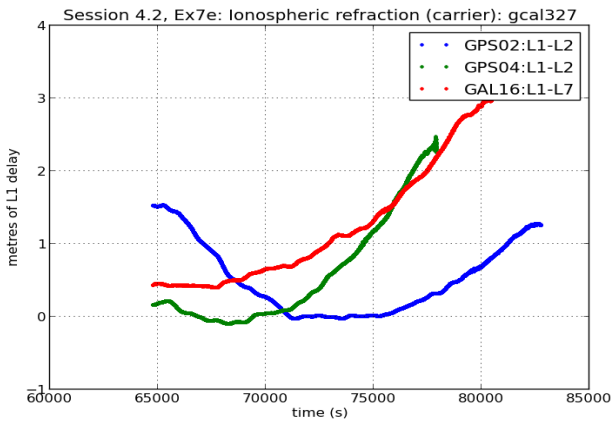

Session 4.2, Ex7b: Sky plot: gcal327
All sat.
GPS 02
GPS 04
GAL 16
West
South

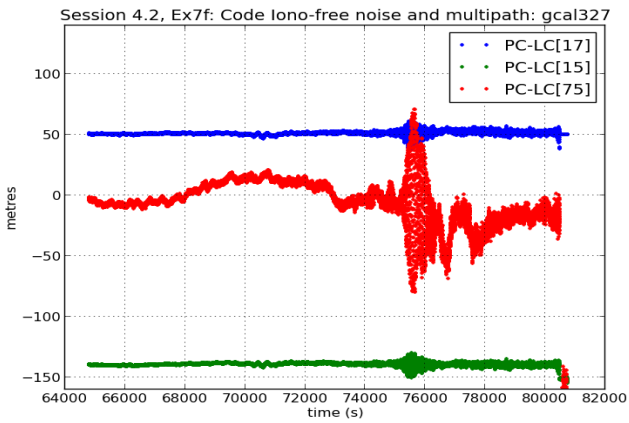

Session 4.2, Ex7d: Interfrequency biases: gcal327
GAL16: C1B-C1C
metres
time (s)

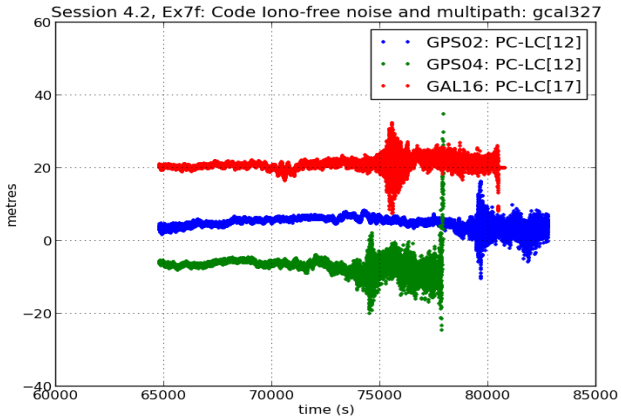

Session 4.2, Ex7e: Ionospheric refraction (code): gcal327
C5-C1
C7-C1
C5-C7
metres of L1 delay
time (s)

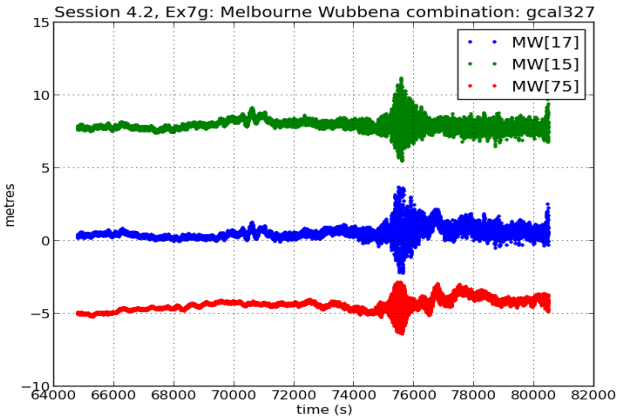

Session 4.2, Ex7e: Ionospheric refraction (code): gcal 327
GPS02:P2-P1
GPS04:P2-P1
GAL16:P7-P1
metres of L1 delay
time (s)

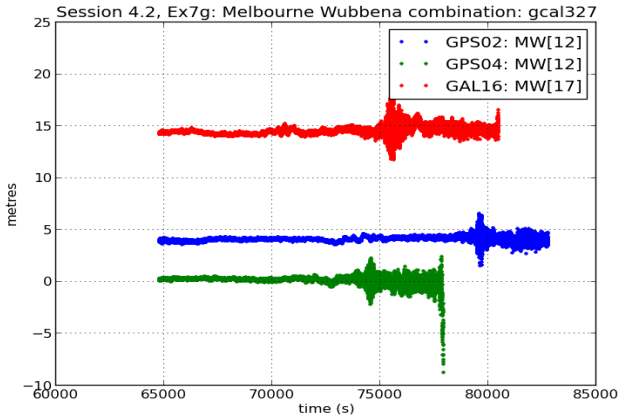

Session 4.2, Ex7e: Ionospheric refraction (Carrier): gcal327
E1-E5
E1-E7
E7-E5
metres of L1 delay
time (s)

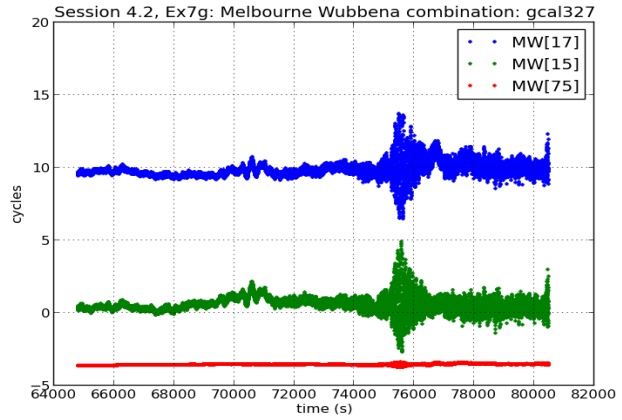

Session 4.2, Ex7e: Ionospheric refraction (carrier): gcal327
GPS02:L1-L2
GPS04:L1-L2
GAL16:L1-L7
metres of L1 delay
time (s)

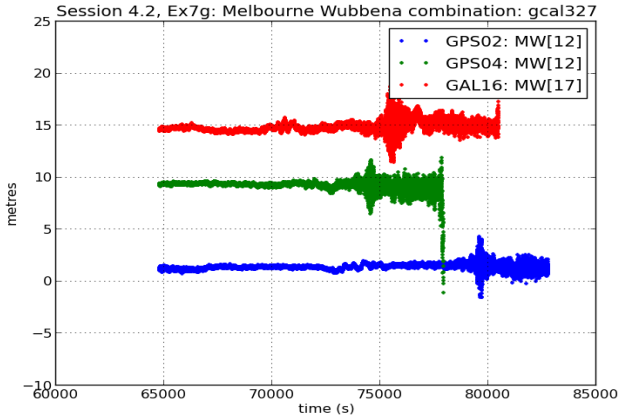

Session 4.2, Ex7f: Code Iono-free noise and multipath: gcal327
PC-LC[17]
PC-LC[15]
PC-LC[75]
metres
time (s)

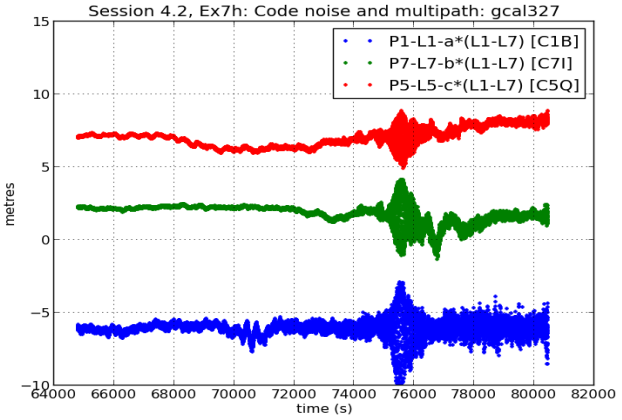

Session 4.2, Ex7f: Code Iono-free noise and multipath: gcal327
GPS02: PC-LC[12]
GPS04: PC-LC[12]
GAL16: PC-LC[17]
metres
time (s)

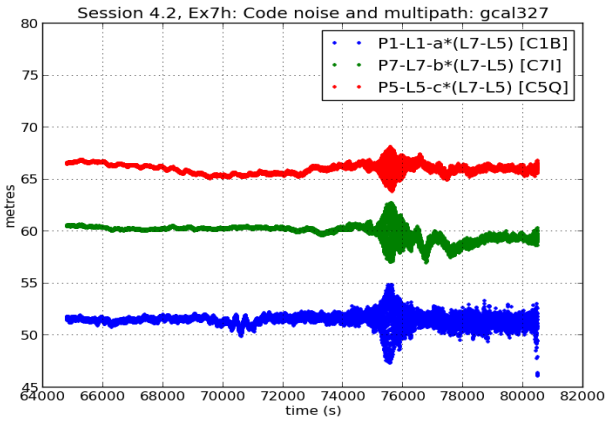

Session 4.2, Ex7g: Melbourne Wubbena combination: gcal327
MW[17]
MW[15]
MW[75]
metres
time (s)

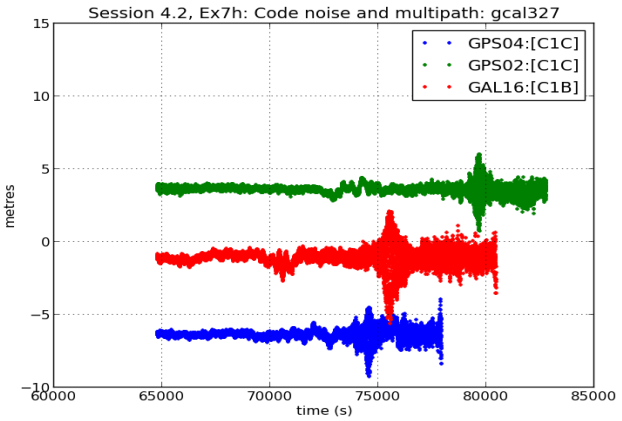

Session 4.2, Ex7g: Melbourne Wubbena combination: gcal327
GPS02: MW[12]
GPS04: MW[12]
GAL16: MW[17]
metres
time (s)

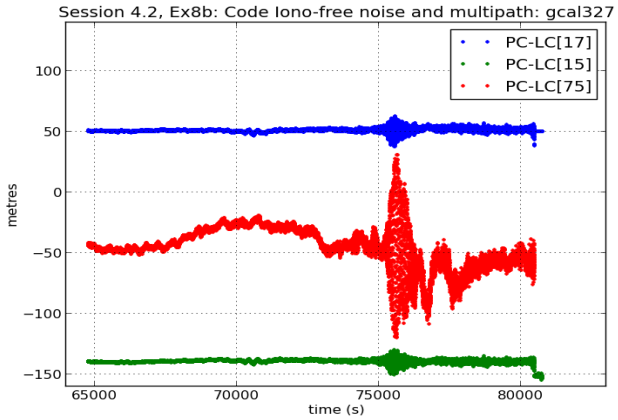

Session 4.2, Ex7g: Melbourne Wubbena combination: gcal327
MW[17]
MW[15]
MW[75]
cycles
time (s)

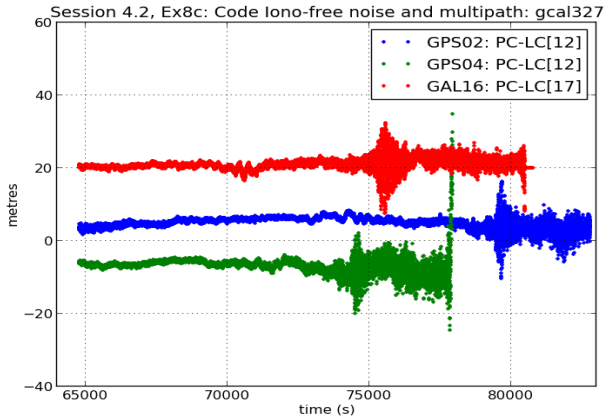

Session 4.2, Ex7g: Melbourne Wubbena combination: gcal327
GPS02: MW[12]
GPS04: MW[12]
GAL16: MW[17]
metres
time (s)

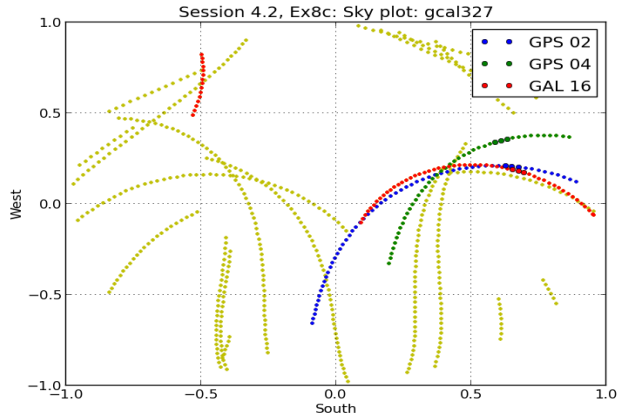

Session 4.2, Ex7h: Code noise and multipath: gcal327
P1-L1-a*(L1-L7) [C1B]
P7-L7-b*(L1-L7) [C7I]
P5-L5-c*(L1-L7) [C5Q]
metres
time (s)

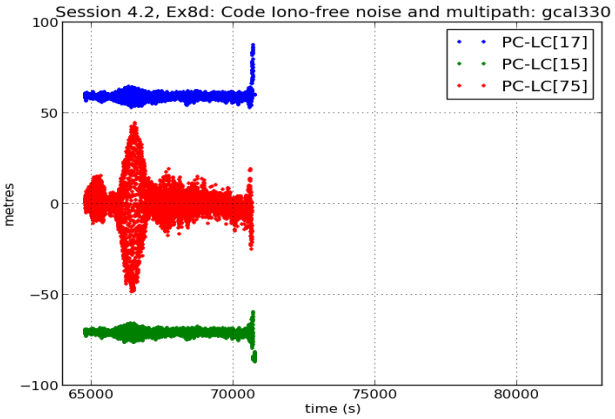

Session 4.2, Ex7h: Code noise and multipath: gcal327
P1-L1-a*(L7-L5) [C1B]
P7-L7-b*(L7-L5) [C7I]
P5-L5-c*(L7-L5) [C5Q]
metres
time (s)

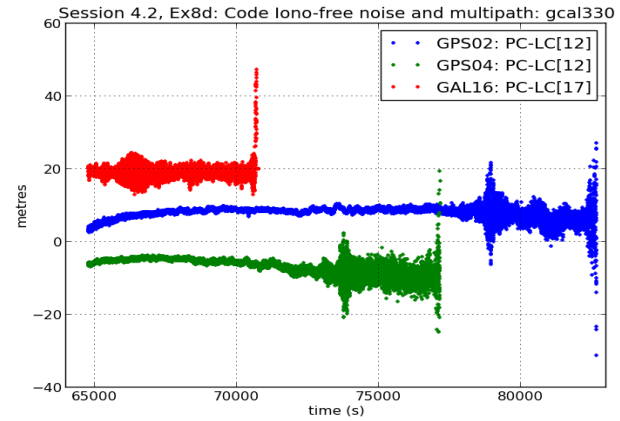

Session 4.2, Ex7h: Code noise and multipath: gcal327
GPS04:[C1C]
GPS02:[C1C]
GAL16:[C1B]
metres
time (s)

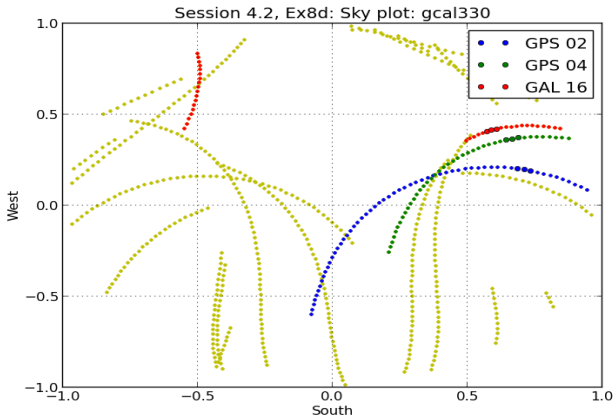

Session 4.2, Ex8b: Code Iono-free noise and multipath: gcal327
PC-LC[17]
PC-LC[15]
PC-LC[75]
metres
time (s)


Session 4.2, Ex8c: Code Iono-free noise and multipath: gcal327
GPS02: PC-LC[12]
GPS04: PC-LC[12]
GAL16: PC-LC[17]
metres
time (s)


Session 4.2, Ex8c: Sky plot: gcal327
GPS 02
GPS 04
GAL 16
West
South


Session 4.2, Ex8d: Code Iono-free noise and multipath: gcal330
PC-LC[17]
PC-LC[15]
PC-LC[75]
metres
time (s)


Session 4.2, Ex8d: Code Iono-free noise and multipath: gcal330
GPS02: PC-LC[12]
GPS04: PC-LC[12]
GAL16: PC-LC[17]
metres
time (s)


Session 4.2, Ex8d: Sky plot: gcal330
GPS 02
GPS 04
GAL 16
West
South

Session 4.3. Combinations of Trios of Carrier and Code Measurements

Objectives

To explore new combinations of signals from three-frequency systems, involving trios of carrier or code measurements: the *first-order ionosphere-free and geometry-free* combination, the *first- and second-order ionosphere-free* combination and the *second-order ionosphere-free and geometry-free* combination. To discuss their physical content, properties and suitability for GNSS applications.

Files to use

`meas.cfg`, `gien327sw.09o`, `gcal327sw.09o`, `orb15591.sp3`, `l5dt1260.09o`, `l5dt1260.09n`

Programs to use

`gLAB_linux`, `graph.py`

Development

Session files: Copy and uncompress these session files in the working directory:

```
cp ~/GNSS/PROG/SES43/* .
cp ~/GNSS/FILES/SES43/* .
gzip -d *.gz *.Z
```

1. Ionosphere-free and geometry-free combination of carriers

Show that there is a unique[16] linear combination of three carriers cancelling both the geometry and the (first-order) ionospheric refraction.

Hint: Let $a_i\,L_i + a_j\,L_j + a_k\,L_k$ be a linear combination of three-carrier[17] *measurements at frequencies f_i, f_j and f_k. The coefficients a_i, a_j, a_k should verify the conditions*

$$\begin{aligned} &\text{Geometry free:} && a_i + a_j + a_k = 0 \\ &\text{Ionosphere free:} && a_i/f_i^2 + a_j/f_j^2 + a_k/f_k^2 = 0 \end{aligned} \tag{4.36}$$

Solution:

$Comb \equiv (\gamma_{ij} - \gamma_{ik})L_i - (1 - \gamma_{ik})L_j + (1 - \gamma_{ij})L_k$, with $\gamma_{ij} = (f_i/f_j)^2$

[16] Except by a scale factor.
[17] Or code measurements, as well.

2. Combinations of trios of three-frequency carriers

Let $I1_i$ and $I2_i$ be the first- and second-order ionospheric effects on the GNSS signals at the frequency f_i, and ρ the geometric range plus all other non-dispersive terms and carrier ambiguity. Taking into account expressions (5.40) to (5.43) of Volume I, answer the following questions:

(a) Show that the carrier phase measurements L_i, L_j, L_k at frequencies f_i, f_j and f_k can be written as:

$$\begin{aligned} L_i &= \rho - I1_i + I2_i \\ L_j &= \rho - \gamma_{i_j}\, I1_i + \gamma_{i_j}^{3/2}\, I2_i \\ L_k &= \rho - \gamma_{i_k}\, I1_i + \gamma_{i_k}^{3/2}\, I2_i \end{aligned} \tag{4.37}$$

with $\gamma_{mn} = (f_m/f_n)^2$.

(b) Consider the trios of Galileo carriers [E1, E7, E8], [E1, E7, E5] and the GPS carriers [L1, L2, L5]. Verify the expressions

Galileo [E1, E7, E8] (Note: $\rho \leftrightarrow LC2$, $I1_1 \leftrightarrow LI1$, $I2_1 \leftrightarrow LI2$)	
$LC2 = 6.964\, L_1 - 78.318\, L_7 + 72.354\, L_8,$	$\sigma_{LC2} = 106.8\, \sigma_L$
$LI1 = 12.110\, L_1 - 182.385\, L_7 + 170.275\, L_8,$	$\sigma_{LI1} = 249.8\, \sigma_L$
$LI2 = 6.146\, L_1 - 104.067\, L_7 + 97.921\, L_8,$	$\sigma_{LI2} = 143.0\, \sigma_L$

(4.38)

Galileo [E1, E7, E5]	
$LC2 = 6.722\, L_1 - 39.311\, L_7 + 33.589\, L_5,$	$\sigma_{LC2} = 52.1\, \sigma_L$
$LI1 = 11.541\, L_1 - 90.587\, L_7 + 79.046\, L_5,$	$\sigma_{LI1} = 120.8\, \sigma_L$
$LI2 = 5.819\, L_1 - 51.277\, L_7 + 45.458\, L_5,$	$\sigma_{LI2} = 68.8\, \sigma_L$

(4.39)

GPS [L1, L2, L5]	
$LC2 = 7.081\, L_1 - 26.130\, L_2 + 20.050\, L_5,$	$\sigma_{LC2} = 33.7\, \sigma_L$
$LI1 = 12.368\, L_1 - 60.215\, L_2 + 47.847\, L_5,$	$\sigma_{LI1} = 77.9\, \sigma_L$
$LI2 = 6.287\, L_1 - 34.084\, L_2 + 27.797\, L_5,$	$\sigma_{LI2} = 44.4\, \sigma_L$

(4.40)

where L_i carriers are assumed uncorrelated and with the same σ_L.

Note that the last equation of the previous sets (4.38) to (4.40) corresponds to the solution found in the previous exercise multiplied by the scale factor $a_1 = 1/[(\gamma_{1_j} - \gamma_{1_k}) - (1 - \gamma_{1_k})\gamma_{1j}^{3/2} + (1 - \gamma_{1_j})\gamma_{1k}^{3/2}]$. This scale factor expresses the combination in units of L_1 delay.

Hint: Let $\mathbf{y} = \mathbf{A}\,\mathbf{x}$ *represent the equation system (4.37), where* $\mathbf{y} = [L_i, L_j, L_k]^T$, $\mathbf{x} = [\rho, I1, I2]^T$ *and* $\mathbf{A}$ *is the associated matrix evaluated for the given frequencies. Then the combinations (4.38) to (4.40) are given by* $\mathbf{A}^{-1}$. *The* σ*s are the square root of the diagonal elements of matrix* $(\mathbf{A}^T\mathbf{A})^{-1}$. *It can be easily computed with Octave or MATLAB, as well.*

3. Combinations of trios of three-frequency codes

Let $I1_i$ and $I2_i$ be the first- and second-order ionospheric effects on the GNSS signals at the frequency f_i, and ρ the geometric range plus all other non-dispersive terms. Taking into account expressions (5.40) to (5.43) of Volume I, answer the following questions:

(a) The code C_i, C_j, C_k at frequencies f_i, f_j and f_k can be written as

$$\begin{aligned} C_i &= \rho + I1_i - 2\,\mathtt{I2_i} \\ C_j &= \rho + \gamma_{i_j}\, I1_i - 2\,\gamma_{i_j}^{3/2}\, I2_i \\ C_k &= \rho + \gamma_{i_k}\, I1_i - 2\,\gamma_{i_k}^{3/2}\, I2_i \end{aligned} \tag{4.41}$$

with $\gamma_{mn} = (f_m/f_n)^2$.

(b) For the trios of Galileo codes [C1, C7, C8], [C1, C7, C5] and the GPS codes [C1, C2, C5], verify the following expressions:

Galileo [C1, C7, C8] (Note: $\rho \leftrightarrow PC2$, $I1_1 \leftrightarrow PI1$, $I2_1 \leftrightarrow PI2$)	
$PC2 = 6.964\,C_1 - 78.318\,C_7 + 72.354\,C_8,$	$\sigma_{PC2} = 106.8\,\sigma_C$
$PI1 = -12.110\,C_1 + 182.385\,C_7 - 170.275\,C_8,$	$\sigma_{PI1} = 249.8\,\sigma_C$
$PI2 = -3.073\,C_1 + 52.033\,C_7 - 48.960\,C_8,$	$\sigma_{PI2} = 71.5\,\sigma_c$

(4.42)

Galileo [C1, C7, C5]	
$PC2 = 6.722\,C_1 - 39.311\,C_7 + 33.589\,C_5,$	$\sigma_{cC2} = 52.1\,\sigma_C$
$PI1 = -11.541\,C_1 + 90.587\,C_7 - 79.046\,C_5,$	$\sigma_{PI1} = 120.8\,\sigma_C$
$PI2 = -2.910\,C_1 + 25.638\,C_7 - 22.729\,C_5,$	$\sigma_{PI2} = 34.4\,\sigma_C$

(4.43)

GPS [C1, C2, C5]	
$PC2 = 7.081\,C_1 - 26.130\,C_2 + 20.050\,C_5,$	$\sigma_{PC2} = 33.7\,\sigma_C$
$PI1 = 12.368\,C_1 - 60.215\,C_2 + 47.847\,C_5,$	$\sigma_{PI1} = 77.9\,\sigma_C$
$PI2 = -3.144\,C_1 + 17.042\,C_2 - 13.899\,C_5,$	$\sigma_{PI2} = 22.2\,\sigma_C$

(4.44)

where the C_i codes are assumed uncorrelated and with the same σ_C noise.

Note that, as expected from a glance at the equation systems (4.37) and (4.41), $PI1 = -LI1$ and $PI2 = -2\,LI2$, with $PC2 = LC2$.

Hint: Proceed as in the previous exercise.

4. First- and second-order ionosphere-free combination analysis

The following questions are devoted to analysing the suitability of the combinations LC2 and PC2, obtained in the previous exercises, for navigation:

(a) Compare the theoretical noise of *first- and second-order* ionosphere-free combination (σ_{LC2} and σ_{PC2}) with the noise of the *first-order* ionosphere-free combination (σ_{LC} and σ_{PC}) of equations (4.13) and (4.15) of session 4.2.

(b) Which combinations are more suitable for navigation: PC and LC given by the expressions (4.10) to (4.12) and (4.14); or PC2 and LC2 given by the first equations of (4.38) to (4.40) and (4.42) to (4.44)?

(c) Produce a plot to compare the code noise of the *first-order* ionosphere-free combination of Galileo signals [C1,C7] and the *first- and second-order* ionosphere-free combination of Galileo signals [C1,C7,C8].

Complete the following steps:

i. Generate the MEAS file:

The RINEX-3.00 file `gien327sw.09o` contains (E1, E7, E8) Galileo measurements for the satellite PRN16 collected by a permanent receiver. The corresponding orbits and clocks are given in the `orb15591.sp3` file.

Execute for instance:

```
gLAB_linux -input:cfg meas.cfg -input:obs gien327sw.09o
                -input:sp3 orb15591.sp3 > gien327.09.meas
```

The generated MEAS file `gien327.09.meas` will contain the following fields:

Galileo PRN16 measurements:

```
MEAS YY DoY sec GAL PRN el Az N. list C1B L1B C1C L1C C7Q L7Q C8Q L8Q
  1   2  3   4   5   6   7  8  9  10   11  12  13  14  15  16  17  18
```

ii. Plot the results.
Execute for instance:

First- and second-order Ionosphere Free (IFree) combination:

```
gawk '{if ($5=="GAL" && $6==16) {print $4,$11-1.42*($15-$11)
       -($12+1.42*($12-$16)),
           6.93*($11-$12)-78.32*($15-$16)+72.35*($17-$18)}}'
                            gien327.09.meas > gien327Pc2Lc2.dat
```

Plotting:

```
graph.py -f gien327Pc2Lc2.dat -x1 -y3 -l "[178]"
                        -f gien327Pc2Lc2.dat -x1 -y2 -l "[17]"
```

(d) (Optional) Repeat the plot for the Galileo signals [C1,C7,C5]. Complete the following steps:

i. Generate the MEAS file:

The RINEX-3.00 file `gcal327sw.09o` contains (E1, E7, E5) Galileo measurements for the satellite PRN16 collected by a permanent receiver. The associated orbits and clocks are given in file `orb15591.sp3`.

Execute for instance:

```
gLAB_linux -input:cfg meas.cfg -input:obs gcal327sw.09o
               -input:sp3 orb15591.sp3 > gcal327.09.meas
```

The generated MEAS file `gcal327.09.meas` will contain the following fields:

```
Galileo PRN16 measurements:
MEAS YY DoY sec GAL PRN el Az N. list C1B L1B C1C L1C C5Q L5Q C7I L7I
  1   2  3   4   5   6   7  8  9  10   11  12  13  14  15  16  17  18
```

ii. Plot the results:

The same plot as in previous exercise, but with `gcal327.09.meas`.

(e) (Optional) Repeat the plot for the GPS [C1,C2,C5]. The data file `l5dt126.09.meas` of the previous exercise can be used:

i. Generate the MEAS file:

The RINEX-2.10 file `l5dt1260.09o` contains (L1, L2, L5) GPS measurements for the satellite PRN01 collected by a permanent receiver. The associated orbits and clocks are given in the broadcast navigation file `l5dt1260.09n`.

Execute for instance:[18]

```
gLAB_linux -input:cfg meas.cfg
    -input:obs gcal327sw.09o -input:obs l5dt1260.09o
    -input:nav l5dt1260.09n | grep GPS > l5dt126.09.meas
```

The generated MEAS file `l5dt126.09.meas` will contain the following fields:

```
GPS PRN01 measurements:
MEAS YY DoY sec GAL PRN el Az N. list C1C xxx C1P L1P xxx xxx C2P L2P C5X L5X
  1   2  3   4   5   6   7  8  9  10   11  12  13  14  15  16  17  18  19  20
```

ii. Plot the results:

The same plot as in previous exercises, but with `l5dt126.09.meas`.

[18] Note that the P1 code is not available in this file but, with the model option '`-model:dcb:p1c1 flexible`' set by default, the P1 code is emulated by the C1 code. As explained in `gLAB_linux -help`, in `Flexible` mode `gLAB` identifies C1 and P1 codes when one of them is missing. The P1 code is required by `gLAB` for GPS L1 carrier pre-alignment.

5. Geometry-free and second-order ionosphere-free combination analysis

The following questions are devoted to analysing the suitability of the combinations LI1 and PI1, obtained in the previous exercises, to estimate the STEC (i.e., the first-order slant ionospheric delay):

(a) Compare the theoretical noise of the *geometry-free and second-order ionosphere-free* combination (σ_{PI1} and σ_{LI1}) with the noise of the *geometry-free* combination (σ_{PI} and σ_{LI}) of equations (4.7) and (4.9).

(b) Which combinations are more suitable for estimating the ionospheric refraction: the PI and LI given by expressions (4.4) to (4.6) and (4.8); or the PI1 and LI1 given by equations (4.38) to (4.40) and (4.42) to (4.44)?

(c) Produce a plot to compare the code noise of the *Geometry-free* combination of Galileo signals [C1,C7] and the *Geometry-free and second-order ionosphere-free* combination of Galileo signals [C1,C7, C8]. Repeat the plot for the carrier measurements. The measurement file `gien327.09.meas` from the previous exercise can be used.

Execute for instance:

Code: Geometry Free (GFree) and first-order IFree combination:

```
gawk '{if ($5=="GAL" && $6==16){print $4,1.42*($15-$11),
     -12.11*$11+182.38*$15-170.27*$17}}' gien327.09.meas
                                          > gien327Pi1.dat
```

Plotting results:

```
graph.py -f gien327Pi1.dat -x1 -y'($3+3268)' -l "PI1"
-f gien327Pi1.dat -x1 -y'($2+384)' -l "PI"
```

Execute for instance:

Carrier: GFree and first-order IFree combination:

```
gawk '{if ($5=="GAL" && $6==16){print $4,1.42*($12-$16),
      12.11*$12-182.38*$16+170.27*$18}}' gien327.09.meas
                                          > gien327Li1.dat
```

Plotting results:

```
graph.py -f gien327Li1.dat -x1 -y'($3-3268)' -l "LI1"
              -f gien327Li1.dat -x1 -y'($2-384)' -l "LI"
```

(d) (Optional) Repeat the plots for the Galileo signals [C1,C7,C5]. The data file `gcal327.09.meas` of the previous exercise can be used.

(e) (Optional) Repeat the plots for the GPS [C1,C2,C5]. The data file `l5dt126.09.meas` of previous exercises can be used. Discuss the possible source of the large pattern seen in the carrier plot. (Note: It is an elevation-dependent pattern. See session 6.1, exercise 2.)

6. Geometry-free and first-order ionosphere-free combination analysis

The following questions are devoted to analysing the suitability of the combination LI2, obtained in the previous exercises, to estimate the second-order ionospheric refraction (i.e., the second-order slant ionospheric delay):

(a) Discuss which errors can affect the *geometry-free and first-order ionosphere-free* combination combination LI2 (e.g. multipath, APCs , etc.).

(b) Determine if the second-order ionospheric refraction can be depicted from the GPS or Galileo carrier phase measurements. Consider an ideal case with no multipath and no APC errors, and take in the last equations of (4.38) to (4.40) $\sigma_L \simeq 2\,\text{mm}$.

Note that, for ground measurements, the effect of I2 is less than 2 cm in L1 [HJS, 2007]; see also exercise 7.

(c) Draw a plot to show the *geometry-free and first-order ionosphere-free* combination of Galileo carrier phase measurements [E1,E7,E8]. The measurement file `gien327.09.meas` can be used.

Execute for instance:

GFree and first-order IFree combination computation:

```
gawk '{if ($5=="GAL" && $6==16) {print $4,6.15*$12-
              104.07*$16+97.92*$18}}' gien327.09.meas
                                        > gien327Li2.dat
```

Plotting results:

```
graph.py -f gien327Li2.dat -x1 -y'($2-1660)' -l "LI2"
```

(d) (Optional) Repeat the plots for the Galileo signals [E1,E7,E5]. The data file `gcal327.09.meas` of the previous exercise can be used.

(e) (Optional) Repeat the plots for the GPS [L1,L2,L5]. The data file `l5dt126.09.meas` of previous exercises can be used. Discuss the possible source of the large pattern seen in the plot.
Hint: Consider the effect of different APC offsets for the three frequencies. They will not cancel in this combination of carriers, and produce an elevation-dependent effect. See exercise 2 and question 7d of session 6.1.

7. Show that the second-order ionospheric delay on the L1 carrier produced by 10 TECUs of STEC[19] is less than 2 mm. Which values would reach $I2_{L1}$ in the Halloween ionospheric storm (30 October 2003) analysed in exercise 11 of session 4.1?

Hint: Consider the expression (see section 5.4.1 in Volume I)

$$I2_{L1} = -\frac{7527\,c}{2\,f_1^3}\int_{rcv}^{sat} N_e B\cos\theta\,dl \simeq -\frac{7527\,c\,B_0\cos\theta}{2\,f_1^3}\int_{rcv}^{sat} N_e\,dl \quad (4.45)$$

[19] That is, $\int_{rcv}^{sat} N_e\,dl = 10\,\text{TECUs} = 10^{17}\text{e}^-/\text{m}^2$, see equation (5.38) in Volume I.

where the value $B_0 \simeq 4 \cdot 10^{-5}$ *T can be taken for the magnetic field magnitude (at the intersection point of the satellite–receiver ray with an ionospheric layer assumed to be about* 400 *km in height). Note that* $f_1 = 1575.420 \cdot 10^6$ *Hz.*

8. The previous expressions (4.38) to (4.40) and (4.42) to (4.44) give the I2 effect in L1 delay units. Determine the following relation between the I2 delay in the L1 carrier and in the (first-order) ionosphere-free combination $LC_{[12]}$:

$$I2_{LC_{[12]}} = -\frac{\gamma_{12}}{1+\sqrt{\gamma_{12}}} I2_{L1} \tag{4.46}$$

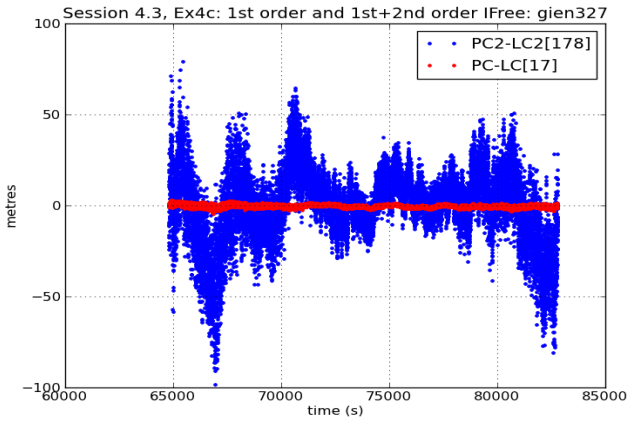
9. Assuming that the STEC can be estimated from smoothed code PI (see equation (4.4)), after removing the DCBs,[20] with an accuracy better than 5 TECUs, and assuming a negligible error in the magnetic field B value,[21] show that the second-order I2 effect can be calculated from (4.45) with an error less than 1 mm of delay in the L1 carrier.

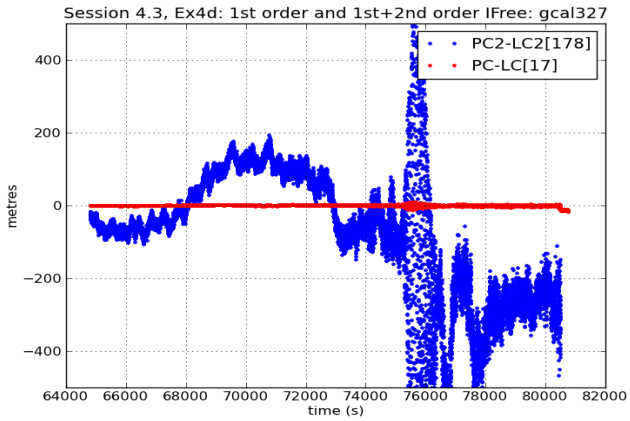
10. Taking into account the previous results, and considering high-accuracy geodetic positioning, determine what is more suitable to remove this effect on the (first-order) ionosphere-free combination LC: that is, to use a model or to remove this effect from a linear combination of three-frequency signals?

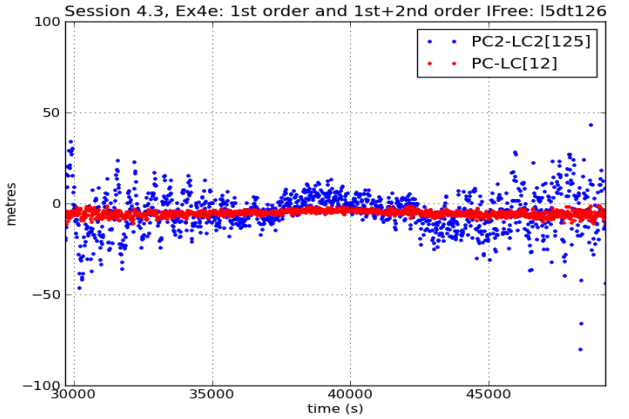
[20] The DCBs can be obtained from the IONEX files available at the IGS site *ftp://cddis.gsfc.nasa.gov/gps/products/ionex/* .

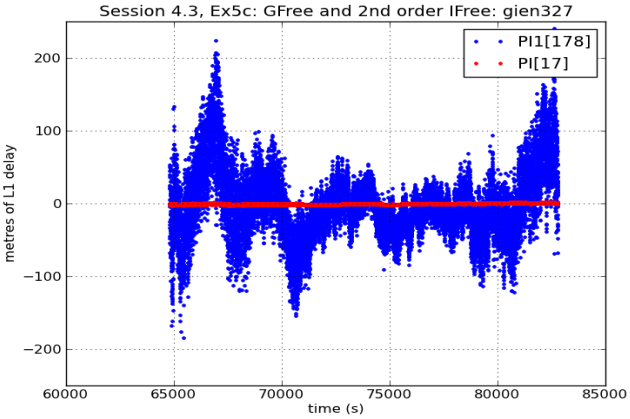
[21] The subroutine International Geomagnetic Reference Field (IGRF) [Tsyganenko, 2003] could be used as well to compute the magnetic field.

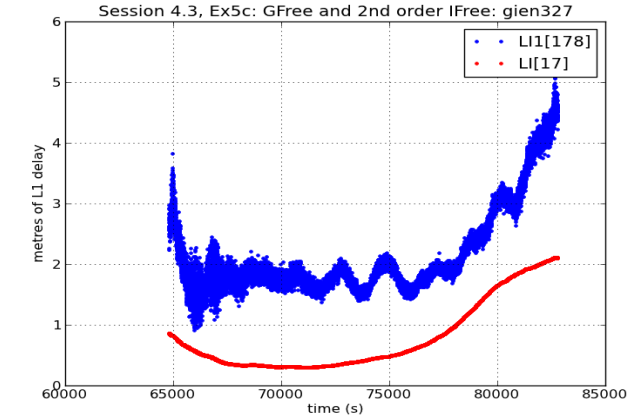
Graphs Session 4.3

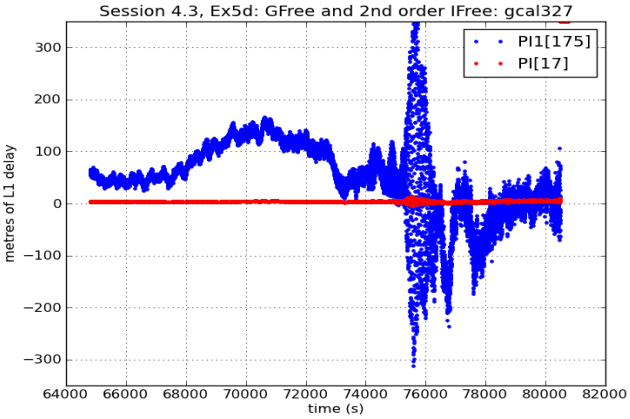


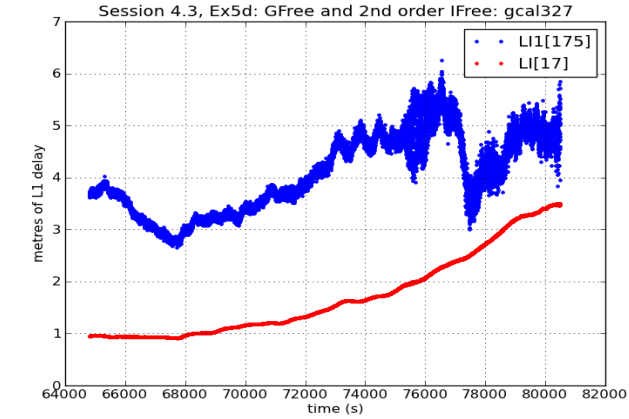


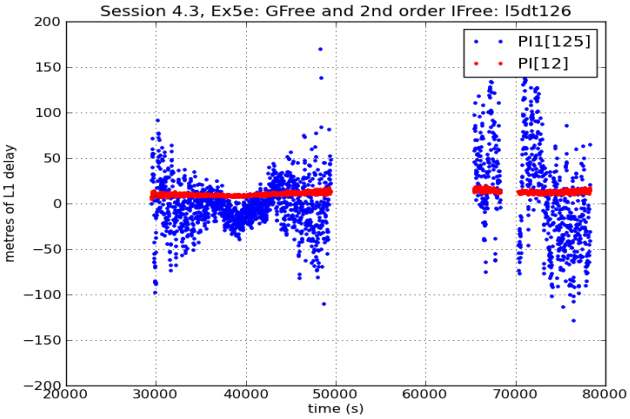


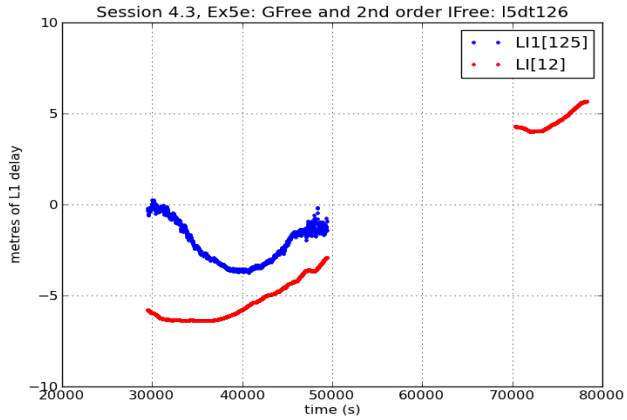


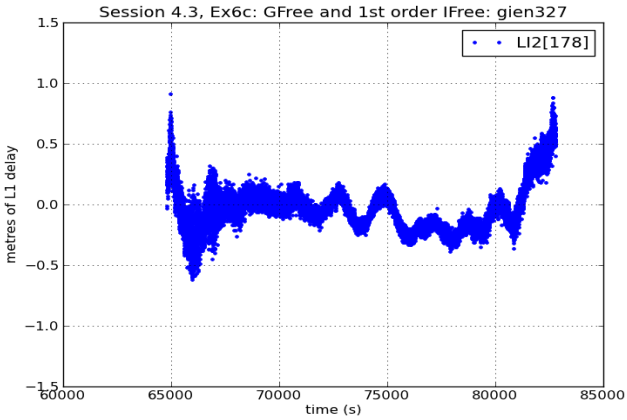


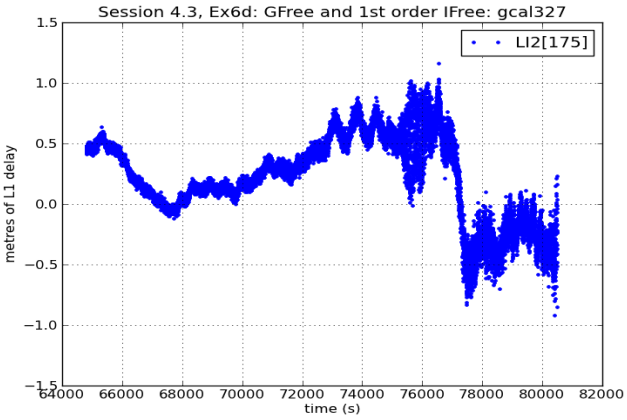


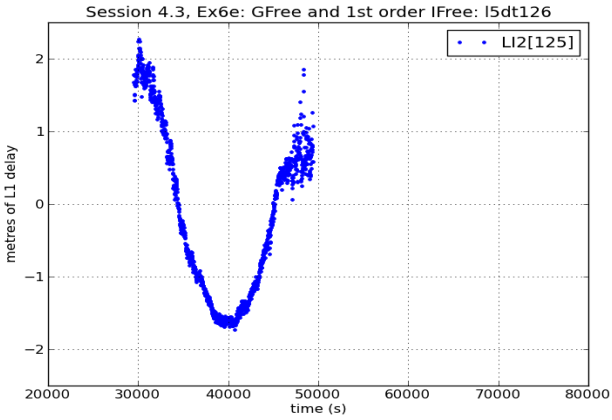





Session 4.3, Ex5e: GFree and 2nd order IFree: l5dt126
LI1[125]
LI[12]
metres of L1 delay
time (s)


Session 4.3, Ex6c: GFree and 1st order IFree: gien327
LI2[178]
metres of L1 delay
time (s)


Session 4.3, Ex6d: GFree and 1st order IFree: gcal327
LI2[175]
metres of L1 delay
time (s)


Session 4.3, Ex6e: GFree and 1st order IFree: l5dt126
LI2[125]
metres of L1 delay
time (s)

5. Measurement Modelling and Positioning

Computation of the measurement modelling and position solutions is treated in detail here in three laboratory sessions.

The first session is devoted to describing in depth the gLAB configuration for both Standard Point Positioning (SPP) and Precise Point Positioning (PPP). The different options used, data files and reference values are checked and discussed in order to show the user how to obtain full traceability of the data processing flow. Using the same data files (measurements, orbits and clocks) and references (ANTEX, ARP, etc.) as in the IGS solution of a Solution (Software/technique) INdependent EXchange format (SINEX) file, the IGS coordinate estimates are reproduced with centimetre-level accuracy, as a driving example to illustrate in detail the gLAB settings for high-accuracy positioning.

The capability of gLAB to process trajectories is illustrated by the kinematic positioning of a roving receiver onboard a Low Earth Orbit (LEO) satellite. The particular context of this exercise is used to discuss different modelling options and measurements for use in this kinematic orbit determination.

The second session focuses on the model component analysis for SPP and PPP. The different model terms (troposphere, ionosphere, satellite clocks, relativistic clock corrections, satellite and receiver antenna phase centres, solid tides, wind-up) are graphically analysed in the range domain. The impact on the position domain of neglecting each one of these components is assessed.

The Differential Code Bias (DCB) P1–C1 handling for the different GPS receiver types is included, as an advanced topic, and the impact of neglecting this DCB correction (for receiver types 1 and 2) on the precise receiver coordinates and clock is evaluated. As a complementary exercise, the P1–C1 DCB values are estimated with a receiver type 3 (Ashtech Z-XII) and the results are compared with IGS precise products.

Hand-made examples of the implementation of algorithms are provided to help software code development. Indeed, computation of the measurement model components and the prefit residuals is illustrated in detail. This is done by directly identifying the different values to take into RINEX files and using simple software routines which implement elemental functions. Once the prefit residuals and satellite coordinates are computed, the navigation equations system is posed and solved by least squares and a Kalman filter using Octave or MATLAB. The results are compared with the internal computations of gLAB.

Error budget assessment in the SPP model is the aim of the last laboratory session. This is done using precise IGS products (receiver coordinates, clock and troposphere estimates, satellite orbits and clocks) as accurate reference values. The Klobuchar model for the ionospheric corrections is roughly assessed using the receiver measurements and the IGS DCBs. In particular, it analyses how range domain errors are transferred to the position domain.

Session 5.1. Standard and Precise Point Positioning with gLAB

Objectives

To describe in detail all aspects involved in the SPP and PPP modes defined in the GUI templates of gLAB. To illustrate how to track the internal configuration used and to compare with precise references from IGS (SINEX files with precise coordinates and Zenith Tropospheric Delay (ZTD) estimation files).

To illustrate gLAB options and capabilities with the kinematic positioning of a LEO satellite as a baseline example.

Files to use
`UPC11490.05O`, `UPC11490.05N`, `roap1810.09o`, `roap1810.09zpd`, `igs15382.sp3`, `igs05_1525.atx`, `brdc0800.07n`, `cod14193.eph`, `igs09P1538.snx`, `igs15382.clk_30s`, `graa0800.07o`, `GRAA_07i_080.sp3`, `cod14193.clk`, `igs05_1402.atx`.

Programs to use
`gLAB_GUI.py`, `graph.py`, `txt2sp3`

Development

Session files:
Copy and uncompress these session files in the working directory:

```
cp ~/GNSS/PROG/SES51/* .
cp ~/GNSS/FILES/SES51/* .
gzip -d *.gz *.Z
```

1. Standard Point Positioning (SPP)

This introductory exercise is devoted to explaining in detail how to compute the SPP solution with gLAB. The GPS SPS solution of a receiver with fixed coordinates will be computed using gLAB as the driving example, as follows.

(a) *Data processing with* gLAB

i. Execute for instance

`gLAB_GUI.py &`

to run the gLAB GUI.

The `[Positioning]` tab is opened by default. Click the `[Input]` button to open the Input Section panel.

ii. Select the **default options for SPP Template**. This is done by clicking the button `SPP Template` at the bottom of the panel.

iii. In the [Input] section, click the Examine button for the RINEX observation file and select the file UPC11490.05O in the WORK directory.[1] Do the same for the navigation file[2] UPC11490.05N. The ANTEX file is not needed for GPS standalone processing.[3]

iv. Click button Run gLAB to compute the solution.

v. In the [Analysis] tab, click the NEU positioning error button and then click Plot to generate the figure.
As shown in the Graphic Details box, this plot is from file gLAB.out and shows column 18 (DSTAN) versus column 4 (SEC) in Plot Nr.1, column 19 (DSTAE) versus column 4 (SEC) in Plot Nr.2 and column 20 (DSTAU) versus column 4 (SEC) in Plot Nr.3.

Note: This plot shows the discrepancies between the computed solution and the 'A priori receiver position' defined in the [Input] section of the [Positioning] tab. By default gLAB uses the "APPROX POSITION XYZ" of the RINEX file.[4]

vi. In the [Positioning] tab, click the button Show Output (at the bottom right corner) to edit the gLAB output file (by default it is named gLAB.out). In the previous plot, the columns 18 (North), 19 (East) and 20 (Up) of rows labelled by 'OUTPUT'[5] have been represented versus column 4 (time). The computed receiver coordinates (X, Y, Z), in the ECEF system, are given in columns 6, 7 and 8.

(b) Compare values of the (X, Y, Z) receiver coordinates of output file gLAB.out with those of the UPC1 RINEX file:

i. The reference position used to compute the previous 'Receiver NEU position error' is defined in the [Input] section as the 'A priori receiver position'. As commented above, the default option is [⊙ Use RINEX Position] (which is provided in the header as "APPROX POSITION XYZ").

Such coordinates can be found by 'greping' the word POSITION in the RINEX file. That is, execute:

```
grep POSITION UPC11490.05O
```

Compare these values with those provided in the gLAB output file (the name gLAB.out is assigned to this file by default). These values are given in the rows labelled 'OUTPUT'.

[1] By default, the browser selects the files '*.??o', with lower or upper case letters. If the required file does not appear in the browser, then select the option "All files" to see what is available in the directory.

[2] In SPP mode, the button [⊙ Broadcast] of the Orbit and Clock Source is selected by default.

[3] In the SPP, the satellite coordinates are computed from the broadcast navigation message, and are relative to the satellite APC of the ionosphere-free combination (see sections 3.3.1 and 5.6.3, Volume I). On the other hand, the computed receiver coordinates will be relative to the receiver APC for the C1 signal.

[4] Four options are available in the A priori receiver position box: [◯ Calculate], [⊙ Use RINEX position], [◯ Specify] and [◯ Use SINEX file].

[5] The description of the different fields of this file can be obtained (with the tool tips option activated) by placing the mouse over the different message print options in the [Output] section of the [Positioning] tab.

Execute:

```
grep OUTPUT gLAB.out|gawk '{print $6,$7,$8}'|more
```

Note: When using the default configuration of `SPP Template`,[6] *the computed coordinates are given relative to the APC for the L1 signal. Nevertheless, although usually the* `"APPROX POSITION XYZ"` *given in the RINEX file corresponds to the Monument Marker (MM) coordinates (see Fig. 5.22, Volume I), the coordinates of the APC have been written (instead of those of the MM) in the* `UPC11490.05O` *RINEX file, to simplify the comparison.*

(c) Plot the discrepancies between the computed position and the approximate coordinates given in the RINEX file, for the horizontal plane.
In the `[Analysis]` tab, click the `Horizontal positioning error` button and then click `Plot` to generate the figure.

2. Checking the default configuration of gLAB for the SPP

As mentioned above, the default configuration for the SPP navigation mode is set by clicking the button `SPP Template` at the bottom of the GUI, in the `[Positioning]` tab. The different options defined by default, as well as the models involved in the `gLAB` processing, are discussed as follows.

(a) `[Input]` section.
The SPP uses the broadcast navigation message, which is provided in `gLAB` by RINEX navigation files. Thus:

i. In the *Orbit and Clock Source* box, the button `[⊙ Broadcast]` is activated by default to upload the broadcast RINEX navigation file.
ii. In the *Ionosphere Source* box, the button `[⊙ Broadcast (same as navigation)]` is activated by default to get the Klobuchar coefficients (α_n and β_n, $n = 0, \ldots, 3$), see equations (5.50) and (5.51), Volume I) from the same RINEX navigation file used for orbits and clocks.
iii. In the *A priori receiver position* box, the button `[⊙ Use RINEX Position]` is activated in order to use the receiver coordinates of the RINEX file to linearise the navigation equations, see equation (6.6) in Volume I. These values will also be used to compute the positioning error.[7]

(b) `[Preprocess]` section:
i. The `[☒ Data Decimation (s)]` button is activated with a default value of `300` seconds. This is done in order to reduce the size of the

[6] In this SPP default configuration, the buttons `□ Receiver Antenna Phase Centre correction` and `□ Receiver Antenna Reference Point correction` of the `[Modelling]` section are switched off.

[7] If no a priori receiver coordinates are known, then the button `[⊙ Calculate]` must be selected. In this case, `gLAB` uses Earth's centre coordinates (0,0,0) as an a priori value to start up the positioning algorithm, and iterates until the solution converges to an stationary solution. This option must be selected for roving receivers (i.e. to compute trajectories). Note that, in this case, (of course) no positioning error relative to the nominal a priori one is provided; that is, fields 9 to 11 and 18 to 20 of the `OUTPUT message` are set to zero (see section `OUTPUT`).

output data file. It saves time for data processing and, mostly, for writing and plotting results.[8]

The *Satellite Options* box, `[Elevation mask (Degrees)]`, is set to `5` to exclude low-elevation satellites, mostly affected by large multipath errors. Conversely, the option `[☐ Discard Eclipsed Satellites]` is switched off by default in the SPP mode, because their effect is dealt with under broadcast navigation data accuracy, and the geometry improves as the number of satellites used increases.

ii. In the *Cycle-slip Detection* box, the `[☒ L1-C1 difference (F1)]` button is activated with its default configuration parameters (which can be checked by clicking the `configure` button). Note that the SPP uses single-frequency data, thus the detection is based on carrier and code (L1–C1) differences. An algorithm based on Example 2 in section 4.3.2 of Volume I is implemented in `gLAB`.[9]

(c) `[Modelling]` section.

Six options are deactivated by default in SPP:

i. The option `[Satellite mass centre to antenna phase centre correction]` is switched off, because the satellite coordinates computed from broadcast orbits (see the equations of section 3.3.1, Volume I) are relative to the satellite APC (of the ionosphere-free combination).[10]

ii. The options `[Receiver antenna phase centre correction]` and `[Receiver antenna reference point correction]` are switched off. This means that the computed receiver coordinates will correspond, directly, to the receiver APC (of the L1 signal in SPP mode), see section 5.4.6.2, Volume I.

iii. The last three modelling options, `[Wind up correction]`, `[Solid tides correction]` and `[Relativistic path range correction]`, are under the code measurement noise level (even smoothed) and are not needed for the SPP mode. That is, the SPP results will not vary when switching on such corrections.

Other comments: The following options are set by default:

i. `[☒ Ionospheric correction]` `Klobuchar` `▼`[11]

[8] The data decimation does not affect the carrier phase cycle-slip detection (see below), because it is performed before decimating the data.

[9] If the buttons `[☐ Geometric-free CP Combination (F1-F2)]` and `[☐ Melbourne-Wübbena (F1-F2)]` detectors are enabled, then all of them contribute to the cycle-slip detection (setting cycle-slip flags), although processing in SPP mode (i.e. single frequency).

[10] This is not the case for the precise orbits computed from the SP3 files, which are referred to the satellite MC, see section 3.3.3, Volume I. Therefore, this option must be activated when using SP3 files, but not with the broadcast orbits.

[11] When positioning with single-frequency measurements (i.e. when the `[⊙ Single frequency]` button is activated in the `[Filter]` section), an ionospheric model is needed to remove the ionospheric refraction. With two- frequency signals the ionosphere-free combination (PC) can be used to cancel out the ionosphere. This is done when the `[⊙ Dual frequency]` button of the `[Filter]` section is activated to use the ionosphere-free combination (PC) of measurements. In this case, when using PC, the `[☐ Ionospheric correction]` and `[☐ P1-P2 correction]` buttons of the `[Modelling]` section must be deselected.

ii. [☒ Tropospheric correction] UNB-3 Nominal ▼
Simple Mapping ▼

The previous ionospheric and tropospheric models are described in sections 5.4.1.2.1 and 5.4.2.1 of Volume I, respectively.

iii. [☒ P1-C1 correction] Flexible ▼

With this option activated, gLAB uses whichever C1 or P1 measurement is in the file. If both are available, P1 is used by default. This is to prevent C1 or P1 missing from the data file (see details in tool tip).

(d) [Filter] section.
The following buttons in the left-hand panel are activated by default for the SPP Template mode:

i. The [⊙ Single Frequency] button is used for single-frequency positioning (e.g. with GPS C1 code). If the [⊙ Dual Frequency] option is selected, then the ionosphere-free combination (PC) (see equation (5.44), Volume I) is used instead.[12]

Warning: when using the ionosphere-free combination (PC), the [□ Ionospheric Correction] *and* [□ P1-P2 Correction] *must be switched off.*

ii. The [⊙ Kinematic] button is activated as a default option for SPP. In this case the Kalman filter is configured according to section 6.1.2.1.2, equations (6.37) Volume I. The values of the different parameters of matrices $\mathbf{\Phi}$ and $\mathbf{Q}$ are defined in the 'Parameters' box on the right. The covariance matrix $\mathbf{P}_0$ to initialise the filter is also defined (see Fig. 6.2 in Volume I).[13] These matrices are assumed to be diagonal and only their diagonal elements can be set (and the same value is used for the three coordinates x, y, z). That is:

Position:	$\Phi_{xx} = \Phi_{yy} = \Phi_{zz} \equiv$ Phi	($= 0$ for Kinematic option)
	$Q_{xx} = Q_{yy} = Q_{zz} \equiv$ Q	($= 10^8$ for Kinematic option)
	$P_{0_{xx}} = P_{0_{yy}} = P_{0_{zz}} \equiv$ Po	($= 10^8$ by default) $[(10\,\mathrm{km})^2]$
Clock:	$\Phi_{dt\,dt} \equiv$ Phi	($= 0$ by default)
	$Q_{dt\,dt} \equiv$ Q	($= 9 \cdot 10^{10}$ by default) $[(1\,\mathrm{ms})^2]$
	$P_{0_{dt\,dt}} \equiv$ Po	($= 9 \cdot 10^{10}$ by default) $[(1\,\mathrm{ms})^2]$

Data sets collected with a receiver at a fixed location (i.e. a static receiver) can be processed by setting the [⊙ Static] option. In this case, the model of section 6.1.2.1.1, equations (6.36) Volume I, is applied, as the coordinate estimates 'time averaged' by the Kalman filter.

[12] The GNSS measurements used are indicated in the *Measurement configuration and noise* box on the left. In the current version of gLAB, use of the ionosphere-free combination (PC) is hard coded for [⊙ Dual Frequency] in the GUI. Nevertheless, different measurements can be used for single-frequency positioning: C1C, C1P, C2C or C2P.

[13] The Kalman filter is initialised with the coordinates and clock values $\hat{\mathbf{x}}_0 = [0, 0, 0, 0]$ and $dt_0 = 0$, being its covariance matrix $\mathbf{P}_0$.

iii. The [□ Estimate Troposphere] option is not selected for the SPP mode.[14]

iv. In the *Measurements* box, the option [⊙ Pseudorange] is selected by default for SPP. Note that [□ Pseudorange Smoothing (epochs)] is deselected by default. This means that the filter will use raw pseudorange measurements (decimated as indicated in the [Preprocess] section) but without smoothing, see section 4.2.3, Volume I. If the [☒ Pseudorange Smoothing (epochs)] option is selected, then the number of epochs (not the time window)[15] must be specified.

Note: The navigation algorithms of section 6.1 Volume I, are applied when the [⊙ Pseudorange] *option is selected (whether the pseudorange is smoothed or not, or the* [⊙ Single Frequency] *or* [⊙ Dual Frequency] *option is used). On the other hand, when the option* [⊙ Pseudorange + Carrier] *is selected, then the code and carrier measurements are combined in the navigation filter,*[16] *according to the equations*[17] *of section 6.2.2 Volume I. The measurements used in the filter are indicated in the '*Measurement configuration and noise*' box.*

v. In the *Measurement configuration and noise* box on the right, the standard deviation of the measurement noise σ_y is defined (i.e. the prefit residuals, see section 6.1.1.2, Volume I). By default, the option [⊙ Fixed StdDev (m)] is set with $\boxed{1}$ m for the C1C code (i.e. C1 GPS code) in SPP mode. The same value $\sigma_{y_i} = \sigma_y \equiv$ StdDev is used for all measurements.

An elevation-dependent function can be defined by selecting the option [⊙ Elevation StdDev (m)], and setting the parameters $\boxed{a}$, $\boxed{b}$ and $\boxed{c}$ of the function $\texttt{StdDev} = a + b\,\mathrm{e}^{-elev/c}$, see equation (6.25), Volume I.

When the [⊙ Pseudorange + Carrier] option is selected, then the PC and LC (with two-frequency data) or C1C and L1P (with single-frequency data) measurements are used.

(e) [Output] section.
All messages, except MEAS, are selected to be output by default. Note that, when processing with 1 Hz data (without decimating), it is recommended to output only [☒ Print INFO Messages], [☒ Print FILTER Messages], [☒ Print OUTPUT Messages] to save storage space as well as writing and plotting time.

[14] This option is only used for PPP. Notice that when this option is selected a new parameter to be estimated (i.e. the wet tropospheric delay) is created in the filter (see the *Parameters* box on the left) among the receiver coordinates and clock. In PPP mode, the mapping of Niell is used by default for the [☒ Tropospheric correction] option in the [Modelling] section. When this option is selected, the model of section 5.4.2.2, Volume I, is applied.

[15] For a RINEX file with 30 s decimation data, 100 samples of smoothing would mean a 3 000 s time window. This can introduce a large ionosphere divergence using a single-frequency (L1, C1) smoother, see Fig. 4.7, Volume I.

[16] If the [☒ Pseudorange Smoothing (epochs)] is set, then the smoothed code is used instead of the raw code measurement.

[17] The tropospheric wet delay estimate is included only when the [☒ Estimate Troposphere] option is selected.

3. Precise Point Positioning (PPP): Static

This exercise is devoted to explaining in detail how to compute the PPP solution using gLAB. The coordinates of an IGS permanent station will be computed to a centimetre level of accuracy with gLAB, as the driving example.

(a) *Data processing with* gLAB

i. Execute

`gLAB_GUI.py &`

to run the gLAB GUI.

The [Positioning] tab is opened by default. Click the [Input] button to open the input section panel.

ii. Select the **default options for PPP Template**. This is done by clicking the button PPP Template at the bottom of the gLAB GUI.

iii. In the [Input] section, click the Examine button for the RINEX observation file and select the file roap1810.09o. Repeat the same for the SP3 file[18] igs15382.sp3 and for the ANTEX file igs05_1525.atx.[19]

iv. Click the button Run gLAB to compute the solution.

v. In the [Analysis] tab, click the NEU positioning error button and then click Plot to generate the figure.

Note: This plot shows the discrepancies between the computed solution and the A priori receiver position *defined in the* [Input] *section. By default it uses the 'APPROX POSITION XYZ' of the RINEX file (see details in the previous exercise).*[20]

vi. In the [Positioning] tab, click the button Show Output (in the bottom right corner) to edit the gLAB output file (by default it is named gLAB.out). The previous plot corresponds to columns 18, 19 and 20 versus column 4 of rows labelled as 'OUTPUT' in this file.[21] Being the button [⊙ Static] activated in the [Filter] section (this is a default option in PPP), the fields 6, 7 and 8 of the last record of the OUTPUT message in this file provide the best estimate[22] of the receiver (X, Y, Z) coordinates in the ECEF system.

[18] The button [⊙ Precise (1 file)] is already selected by default.

[19] The file igs05_1525.atx is the ANTEX file associated with the igs15382.sp3 orbit and clock file. Nevertheless, a newer file can be used.

[20] *Remark:* The receiver coordinates of the RINEX file are not necessarily accurate. They must be taken only as an APPROX POSITION XYZ.

[21] A description of the different fields of this file can be obtained (with the tool tips option activated) by hovering the mouse over the different messages in the [Output] section of the [Positioning] tab.

[22] That is, the best estimate of the parameters with the filter forward (i.e. as in real-time mode).

(b) Repeat the previous processing, but using the IGS estimates of the ROAP receiver coordinates from the SINEX file igs09P1538.snx. The following procedure can be applied.

In the [Positioning] tab, click the button [Input]. In the *A priori receiver position*, click the button [⊙ Use SINEX file] and click Examine to select the igs09P1538.snx file. Then, click the button Run gLAB to compute the solution.

Plot the NEU coordinate estimates relative to the SINEX values and the Zenith Tropospheric Delay (ZTD) estimate, as indicated next. In the [Analysis] tab, click the NEU positioning error button and then click Plot to generate the figure. Do the same for the ZTD.

(c) As commented above, the last row associated to the OUTPUT label in the gLAB.out file provides the final solution for the (X, Y, Z) receiver coordinates. Compare these estimates with those of the IGS SINEX file for the station ROAP.[23] What is the discrepancy between both solutions?

Note that the requested values can be easily 'greped' from such files by executing:

Coordinates computed by gLAB:

```
grep OUTPUT gLAB.out|tail -1|gawk '{print $6,$7,$8}'
```

Coordinates computed by IGS:

```
grep -i ROAP igs09P1538.snx | grep STA
```

4. Tracking the site values used in the computations

The gLAB output file (named by default gLAB.out) provides all the information on the configuration parameters and reference values used internally by gLAB to process the data. This exercise is aimed at illustrating how to perform the tracking of some values used in the internal computations to better assess and understand the output results.

The following procedure is proposed:

i. Repeat the computation of the static PPP solution, using the default configuration of [PPP Template].

At the bottom of the gLAB GUI, click the PPP Template button to set again its associated default configuration.[24] Then, click the button [⊙ Use SINEX file] and select the igs09P1538.snx file. Finally, click the Run gLAB button.[25]

[23] The SINEX format is available at *http://www.iers.org/MainDisp.csl?pid=190-1100110*.

[24] The default configuration uses, in particular, the RINEX "APPROX POSITION XYZ" coordinates as the 'A priori receiver position'.

[25] Input files roap1810.09o, igs15382.sp3 and igs05_1525.atx should remain loaded if the program has not been exited. Otherwise, upload such files again, as indicated in the previous exercise.

ii. Follow the next steps to track the site configuration parameters used:

(a) *A priori receiver position*
The reference position used to compute the previous 'Receiver NEU position error' is defined in the `[Input]` section as the 'A priori receiver position'. The precise coordinates of the SINEX file have been selected.

Find in file `gLAB.out` the '`A priori receiver position`' used by `gLAB`. These values are given in the rows labelled 'INFO PREPROCESSING'.

Execute for instance:

```
grep INFO gLAB.out | grep PREPROC | grep position
```

Note that these coordinates correspond to the MM in Fig. 5.22, Volume I.

Compare the coordinates obtained with those of the SINEX file found in the previous exercise 3c, in order to confirm that the SINEX coordinates have been used as the 'a priori values' by `gLAB`.

(b) *Antenna Reference Point (ARP)*
The ARP position relative to the MM (see Volume I, Fig. 5.22) is given by the 'site eccentricity vector (ΔUp, ΔEast, ΔNorth)', which is selected in `gLAB` by the button `[☒ Receiver antenna reference point correction]` in `[Modelling]` section, `[Positioning]` tab. The default option is `[⊙ Read from RINEX]`. The eccentricity vector is given in the RINEX header as 'ANTENNA: DELTA H/E/N'.

Find the eccentricity vector in the RINEX file `roap1810.09o` (it is assumed that this option was activated during the data processing).

Execute for instance:

```
grep ANTENNA roap1810.09o
```

Compare these values with those provided in the `gLAB` output file `gLAB.out`. These values are given in the rows labelled 'INFO MODELING' (Note that, in this file, the vector is given in N/E/H coordinates.)

Execute for instance:

```
grep INFO gLAB.out | grep MODELING | grep ARP
```

The values used by IGS for estimating the coordinates are given in the SINEX file, see exercise (4e).

(c) *Receiver Antenna Phase Centre*
The receiver APC is selected by the button `[☒ Receiver antenna phase centre correction]` in `[Modelling]` section, `[Positioning]` tab. The default option is `[⊙ Read from ANTEX]`. When this option is activated, `gLAB` reads the receiver antenna type in the RINEX header ('ANT # / TYPE'), and gets from the ANTEX file the APC vector (North, East, Up) for such an antenna model.

Find the receiver antenna model in the RINEX measurement file `roap1810.09o` and get the APC vector for this antenna model from the ANTEX file.[26]

Execute for instance:[27]

Antenna Model (from RINEX header):

```
grep ANT roap1810.09o| grep TYPE
```

Antenna Phase Centre (from ANTEX file):

```
grep -A12 SEN67157596+CR igs05_1525.atx | grep NORTH
```

Check the label `"PCV TYPE / REFANT"` in the first rows of the ANTEX file and check if the APC corrections are absolute (`A`) or relative (`R`), see section 5.6.1, Volume I.

Edit the file with a text editor, or execute for instance:

```
grep "PCV TYPE" igs05_1525.atx
```

Compare the APC correction values of the ANTEX file with those provided in the `gLAB` output file `gLAB.out`. These values are given in the rows labelled '`INFO MODELLING`'.

Execute:

```
grep INFO gLAB.out | grep MODELING | grep PCO
```

The values used by IGS for estimating the coordinates are given in the SINEX file, see exercise (4e).

(d) *Satellite coordinates*
Edit the SP3 file[28] `igs15382.sp3` with the precise orbits and clocks used in this run and answer the following questions: (1) Which coordinate system is used?[29] (2) What APC corrections model[30] is used? (3) What is the reference clock?[31]

(e) *Cross-check with the SINEX file*
Compare the site configuration parameters used by `gLAB` in this run (i.e. the site eccentricity vector from the MM to ARP and APC corrections) with those used by IGS to compute the coordinates of station ROAP as indicated in the SINEX file `igs09P1538.snx`.

[26] `gLAB` uses only the APC vector (North, East, Up). That is, it does not use the elevation- and azimuth-dependent corrections.

[27] In this instruction it is assumed that the RINEX file `roap1810.09o` and the ANTEX file `igs05_1525.atx` are used.

[28] See the SP3 file format in exercise 10 of session 2.2.

[29] IGS05 is the IGS realization of ITRF2005.

[30] For instance, `PCV:IGS05_1525` indicates the `igs05_1525.atx` ANTEX file.

[31] For instance, referred to the IGS Time (IGST), or referred to a station receiver clock.

Execute for instance:

Site eccentricity vector:
`grep ROAP igs09P1538.snx | grep UNE`
Information on ROAP site (including receiver and antenna model):
`grep ROAP igs09P1538.snx`
APC offsets and ANTEX file used:
`grep SEN67157596+CR igs09P1538.snx`

Has gLAB used in this run the same site configuration parameters as those of the SINEX file?[32] If this is the case, what is the level of agreement between both solutions?

5. PPP: kinematic

Repeat the previous computation, but in kinematic mode.

(a) *Data processing with* gLAB

i. To run the gLAB GUI execute: `gLAB_GUI.py &`
The [Positioning] tab is opened by default. Click the [Input] button to open the input section panel.

ii. Select the **default options for PPP Template**. This is done by clicking the button `PPP Template` at the bottom of the gLAB GUI.

iii. **Switch to PPP Kinematic** mode:
In the [Filter] section, in the Receiver Kinematics box, click the button [⊙ Kinematic].

iv. In the [Input] section, click the `Examine` button for the RINEX observation file and select the file roap1810.09o.[33] Do the same for the SP3 file[34] and the ANTEX file, selecting igs15382.sp3 and igs05_1525.atx, respectively.

v. In the [Input] section, in the *A priori receiver position*, click the button [⊙ Use SINEX file] and select igs09P1538.snx as in the previous exercise.

vi. Click the button `Run gLAB` to compute the solution.

(b) *Data analysis: computed navigation solution*

i. In the [Analysis] tab, click `NEU positioning error` and then click `Plot` to generate the receiver NEU position error plot as a function of time.

ii. In the [Analysis] tab, click `Horizontal positioning error` and then click `Plot` to generate the horizontal positioning error plot.

[32] Note that the same site configuration parameters must be used to compare the computed solution of the SINEX and gLAB output.

[33] Input files roap1810.09o, igs15382.sp3 and igs05_1525.atx should remain loaded if the program has not been exited. Otherwise, upload such files again, as indicated in the previous exercise.

[34] The button [⊙ Precise (1 file)] is already selected by default.

iii. In the [Analysis] tab, click Zenith Tropospheric Delay and then click Plot to generate the ZTD plot.

6. Zenith Tropospheric Delay estimation

(a) *Reference values from IGS*
Generate a file with the tropospheric delays to plot.
Execute (in a single command line):[35]

```
grep ROAP roap1810.09zpd | gawk -F\: '{print $3}' |
       gawk '{print $1,$2/1000}' > roap_igs.trp
```

(b) *Data processing and analysis: Forward filter*

i. Following the same procedure as in the previous two exercises, reprocess the RINEX file roap1810.09o in:
   - Static mode (name the output file gLAB.outS).
   - Kinematic mode (name the output file gLAB.outK).

ii. Compare the results from both processing modes (static and kinematic).
In the [Analysis] tab, click Zenith Tropospheric Delay
The last generated output file, namely gLAB.outK, will be selected for plotting. Click the button [⊙ Plot 2], select the other output file (i.e. gLAB.outS) for this plot and set the same 'Condition' as for Plot 1. Click Plot to generate the ZTD plots.

iii. Compare the results with the IGS estimation. Add the plot of IGS ZTD to the previous plots.
In the [Analysis] tab, click the button [⊙ Plot 3], then click the button Examine and select the file roap_igs.trp. In the X column write 1 in the second box. In the Y column write 2 in the second box. Click Plot.

(c) *Data processing and analysis: Forward+backward filter*
gLAB implements a backward filter option which acts as a mirror, processing the data in a reverse way (or backwards) after the initial processing ('forward processing'). This is a very simple way to emulate a raw smoother, because in the backward processing the filter starts from the final solution computed and then takes advantage of the estimated parameters such as coordinates, carrier phase ambiguities and troposphere.
The backward filter solution will be analysed as follows:

i. Repeat the previous processing in kinematic mode, but using the option [☒ backward filtering] in the [Filter] section.

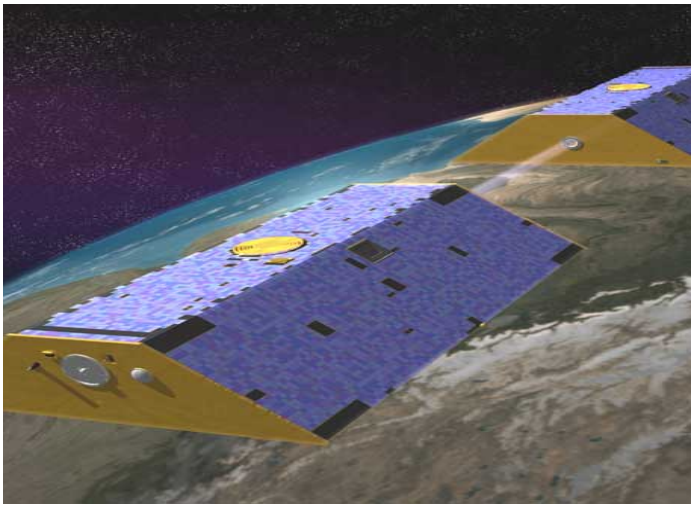
ii. Repeat the previous processing in static mode using this file.

iii. Draw the following plots on the same figure and discuss the results:
(1) Static: forward+backward; (2) Static: forward; and (3) IGS troposphere.

[35] The ZTDs of file roap1810.09zp are in millimetres of delay.

Table 5.1: Description and main characteristics (left) of the GRACE satellite (right).

GRACE SATELLITES (A & B)	
Nominal altitude:	460 km
Orbital period:	1.5 h (approx.)
Mass:	432 kg
Launch date:	17 May 2002
Space agency:	NASA/GFZ
Designed lifetime:	5 years
Receiver pseudorange noise:	40 cm
Receiver carrier phase noise:	8 mm
Receiver GRAPHIC noise:	12 cm
Antenna phase centre:	(0.0, 0.0, −0.414) m



7. Kinematic positioning of an LEO satellite

A GPS receiver onboard an LEO satellite will be positioned kinematically to illustrate some performance results on single- and double-frequency positioning with code or code and carrier data. Indeed, the LEO satellite will be treated as a roving receiver (whether it is on a satellite, or in a car, or on an aircraft, as well) and different positioning approaches and modelling options will be discussed in these exercises.

The file `graa0800.07o` contains code and carrier measurements collected by a receiver onboard the GRACE-A LEO satellite (see Table 5.1).[36] This measurement file will be used to estimate the satellite coordinates in kinematic mode using different approaches. The results will be compared with the precise coordinate file `GRAA_07i_080.sp3`.
Note: These coordinates correspond to the satellite MC.

(a) *Single-frequency positioning with C1 code*
Estimate the coordinates of the GRACE-A LEO satellite in SPP mode with `gLAB`, using the measurement file `graa0800.07o` and the broadcast navigation file `brdc0800.07n`. Assess the accuracy of the estimated coordinates taking as a precise reference the coordinates of file `GRAA_07_080.sp3`.
Complete the following steps:

i. Coordinates computation.
Compute the satellite coordinates using the following configuration of `gLAB`:

- Set the default configuration of `SPP Template` mode.

[36] The Gravity Recovery and Climate Experiment (GRACE) mission is a joint project between National Aeronautics and Space Administration (NASA) and Deutsches Zentrum für Luft- und Raumfahrt (DLR). The primary mission of GRACE is to provide high-accuracy estimates of the global high-resolution models of Earth's gravity field for a period of up to five years. The secondary scientific mission of GRACE is to provide several hundreds of daily Radio Occultation (RO) measures, thus providing vertical information on the ionosphere (see *http://op.gfz-potsdam.de/grace*).

- `[Input]` section:
  - Upload the RINEX measurement file `graa0800.07o` and RINEX navigation file `brdc0800.07n`.
  - In the `A priori receiver position` box, set option `[⊙ Calculate]`.
- `[Preprocess]` section:
  Set `[Data Decimation (s)]` to `30` seconds.
- `[Modelling]` section:
  Uncheck `[□ Tropospheric correction]`.[37]
  Uncheck `[□ Ionospheric correction]`.[38]
- `[Output]` section:
  - Set `gLAB.out` as the output file.
  - Uncheck all messages writing options, except:
    `[☒ Print OUTPUT Messages]`.
  - Set the `[Output file]` as `gLAB.out`.
- Click the button `Run gLAB` to compute the navigation solution.

ii. From output file `gLAB.out`, generate an SP3 format file with the estimated satellite coordinates. The program `txt2sp3` can be used to convert a text columnar format file to SP3 format.

Execute for instance:

(1) Generate the `orbit.txt` file with the time and satellite coordinates (the vehicle number is set to `09`, as in the `GRAA_07_080.sp3` file):

```
grep OUTPUT gLAB.out | grep -v :|
            gawk '{print "09",$2,$3,$4,$6,$7,$8}' > orb.txt
```

(2) Using the program `txt2sp3`, generate the SP3 file (default symbol `"R"` is changed to `"L"`, to match the `GRAA_07_080.sp3` file:

```
cat orb.txt | txt2sp3 | sed 's/R/L/g' > orb.sp3
```

(3) The `orb.sp3` file with the estimated coordinates is compared with the reference coordinates of the `GRAA_07_080.sp3` file:

```
gLAB_linux -input:SP3 GRAA_07_080.sp3 -input:SP3 orb.sp3
      --model:satphasecenter --model:satclocks -pre:dec 0 |
  gawk '{if ($4%60==0) print $0}' | grep SATDIFF >dif.selC1
```

iii. Plot the coordinate discrepancy: (1) 3D error; (2) radial error; (3) along-track error; (4) cross-track error.

[37] The satellite is orbiting at about 450 km in height; that is, over the troposphere.

[38] Orbiting at 450 km in height, the satellite is less affected by the ionosphere than on the ground; nonetheless, some metres of slant delay are expected. On the other hand, as the correction from the Klobuchar model is tuned to ground receivers, its usage could do more harm than good.

Execute for instance:

```
3D error:
graph.py -f dif.selC1 -x4 -y9 --yn -0 --yx 40

Radial error:
graph.py -f dif.selC1 -x4 -y11 --yn -20 --yx 20

Along-track error:
graph.py -f dif.selC1 -x4 -y12 --yn -20 --yx 20

Cross-track error:
graph.py -f dif.selC1 -x4 -y13 --yn -20 --yx 20
```

iv. Zoom in on time interval $43\,000 < t < 67\,000$ and plot the radial, along-track and cross-track error on the same figure.
Execute for instance:

```
graph.py -f dif.selC1 -x4 -y11 -l "Radial error"
         -f dif.selC1 -x4 -y12 -l "Along Track error"
         -f dif.selC1 -x4 -y13 -l "Cross Track error"
               --xn 43000 --xx 67000 --yn -20 --yx 20
```

v. What is the level of accuracy of the estimated coordinates?

(b) *Dual-frequency positioning with PC code*
Repeat the previous computation, but using the ionosphere-free combination of codes (PC).

i. Change the following options from the previous configuration, and compute the coordinates:
- `[Modelling]` section:
Uncheck the `[□ P1-P2 correction]` (i.e. the Total Group Delay (TGD) correction).
- `[Filter]` section: Set the `[⊙ Dual Frequency]` option in the `Available Frequencies` box.
- Click button `Run gLAB` to compute the navigation solution.

ii. As in the previous case, generate the SP3 files from `gLAB.out` and draw the plots.

iii. Has the accuracy improved? Why are the results noisier than in the previous case?

(c) *Dual-frequency positioning with code and carrier data*
Repeat the previous computation, but using the ionosphere-free combination of codes and carriers (PC, LC). Use the precise GPS orbit and clock files `cod14193.eph` and `cod14193.clk`, and the ANTEX file `igs05_1402.atx`.

i. Compute the satellite coordinates using the following configuration of `gLAB`:
- Set the default configuration of the `PPP Template` mode.

- [Input] section:
  – In Orbit and clock Source, select [Precise (2 files)] and upload the SP3 orbits file cod14193.eph and CLK clocks file[39] cod14193.clk.
  – Upload the ANTEX file igs05_1402.atx.
  – In the "A priori receiver position" box, set option [⊙ Calculate].
- [Preprocess] section:
  Set [Data Decimation (s)] to 30 seconds.
- [Modelling] section:
  Uncheck [□ Receiver Antenna phase centre correction].
  Uncheck [□ Receiver Antenna ref. point correction].
  Note: With these options unchecked, the estimated coordinates of GRACE will correspond to the APC in the ionosphere-free combination (PC).
  Uncheck [□ Ionospheric correction].
  Uncheck [□ Tropospheric correction].
  Uncheck [□ P1-P2 correction] (i.e. TGD correction).
  Uncheck [□ Solid Tides correction]
  (because the receiver is not on Earth's surface).
- [Filter] section:
  Uncheck [□ Estimate Troposphere].
  Set [⊙ Kinematic] in the Receiver Kinematics box.
- [Output] section: Uncheck all message writing options, except [☒ Print OUTPUT Messages].
- Click button Run gLAB to compute the navigation solution.

ii. Generate the SP3 files from gLAB.out and draw the plots as in the previous case.

iii. Has the accuracy improved? Why do large error peaks appear?

iv. Why does a mean bias of about half a metre appear in the radial component error?

(d) *Single-frequency positioning with the GRAPHIC combination*
Repeat the previous computation, but using the Group and Phase Ionospheric Calibration (GRAPHIC) combination of single-frequency L1 and C1 carrier and code data.[40] Use precise GPS orbit and clock files cod14193.eph and cod14193.clk, and the ANTEX file igs05_1402.atx.

i. Change the following options from the previous configuration, and compute the coordinates:
- [Filter] section:
  Set the [⊙ Single Frequency] option in the Available frequencies box.[41]

[39] This file contains clocks at 30 s sampling rate.

[40] This combination provides an ambiguous code (due to the carrier ambiguities), but is free from the ionospheric refraction.

[41] Note that the present gLAB architecture requires code measurements when processing carriers. Thus, a large σ is assigned to the code C1 (affected by the ionosphere) to downweight this measurement against the GRAPHIC combination (ionosphere free).

Set [C1C ▼] and [⊙ Fixed StdDev (m)] to [100].

Set [G1C ▼] and [⊙ Fixed StdDev (m)] to [0.5].

- Click button [Run gLAB] to compute the solution.[42]

ii. Generate the SP3 files form `gLAB.out` and draw the plots.

iii. Justify why the accuracy is worse than in the previous case, but better than in cases 7a and 7b.

(e) *Single-frequency positioning with the P1 and L1 code and carrier*
Repeat the previous computation, but using the single-frequency P1 and L1 code and carrier data. Use the precise GPS orbit and clock files `cod14193.eph` and `cod14193.clk`, and the ANTEX file `igs05_1402.atx`.

i. Change the following options from the previous configuration, and compute the coordinates:

- `[Filter]` section:
  Set the `[⊙ Single Frequency]` option in the `Available frequencies` box.

  Set [C1P ▼] and [⊙ Fixed StdDev (m)] to [1].

  Set [L1P ▼] and [⊙ Fixed StdDev (m)] to [0.01].
- `[Modelling]` section: Set `[☒ P1-P2 correction]`.
- `[Input]` section: In the `P1-P2 correction` box, select `DCB Source` [Broadcast (specify) ▼] and upload the RINEX navigation file `brdc0800.07n`.
- Click button [Run gLAB] to compute the solution.[43]

ii. Generate the SP3 files from `gLAB.out` and draw the plots.

iii. Discuss the worsening in accuracy regarding the previous case. Does the ionospheric refraction affect the results?

iv. Would the Klobuchar model be suitable for removing the ionospheric delay for positioning the satellite with carrier phase data?

(f) *Single-frequency positioning with the P1 and L1 code and carrier and applying the Klobuchar model*
Repeat the previous computation, but apply the Klobuchar model and the TGD correction. Use the precise orbit file `cod14193.eph`, the clock file `cod14193.clk` and the ANTEX file `igs05_1402.atx`.

i. Change the following options from the previous configuration, and compute the coordinates.

- `[Modelling]` section: Set `[☒ Ionospheric correction]`.
- `[Input]` section: In the `Ionosphere source` box, select the option `[⊙ Broadcast (specify)]` and upload the RINEX navigation file `brdc0800.07n`.

[42] A `WARNING` will appear when executing `gLAB`. This warning notifies that the single frequency is used for positioning, while two-frequency cycle-slip detectors are used. This is not a problem in this case because the RINEX file contains dual-frequency measurements. Therefore, skip this warning and proceed with the data processing.

[43] The same `WARNING` as in the previous case will appear when executing `gLAB`. Skip this warning and proceed with the data processing.

- Click button `Run gLAB` to compute the solution.[44]

ii. Generate the SP3 files from new `gLAB.out` and draw the plots.

iii. Discuss the worsening in accuracy regarding the previous cases.

iv. Why are the results even worse than in the previous cases?

8. Orbit in space representation with `Google Earth`

Using any of the previous results (the accuracy is not critical here), generate a `kml` file to produce the `Google Earth` view in Fig. 5.1 of the `GRACE-A` satellite in orbit.

Execute for instance:

```
cat Prefix.kml > track.kml

grep OUTPUT gLAB.out | grep -v INFO |
 gawk 'BEGIN{OFS=","}{print $16,$15,$17}' >> track.kml

cat  Postfix.kml >> track.kml
```

Readers aiming to go further into the topics of these exercises can find complementary background in the following tutorials linked to this book:

Tutorial 1 [Sanz et al., 2010] where the ionosphere seen by this LEO satellite is analysed, among other questions.

Tutorial 2 [Sanz et al., 2012a] where radio occultation data from LEO satellites are used to retrieve electron density profiles of the ionosphere (using the Abel transform) and where the atmospheric bending of GNSS signals is analysed.

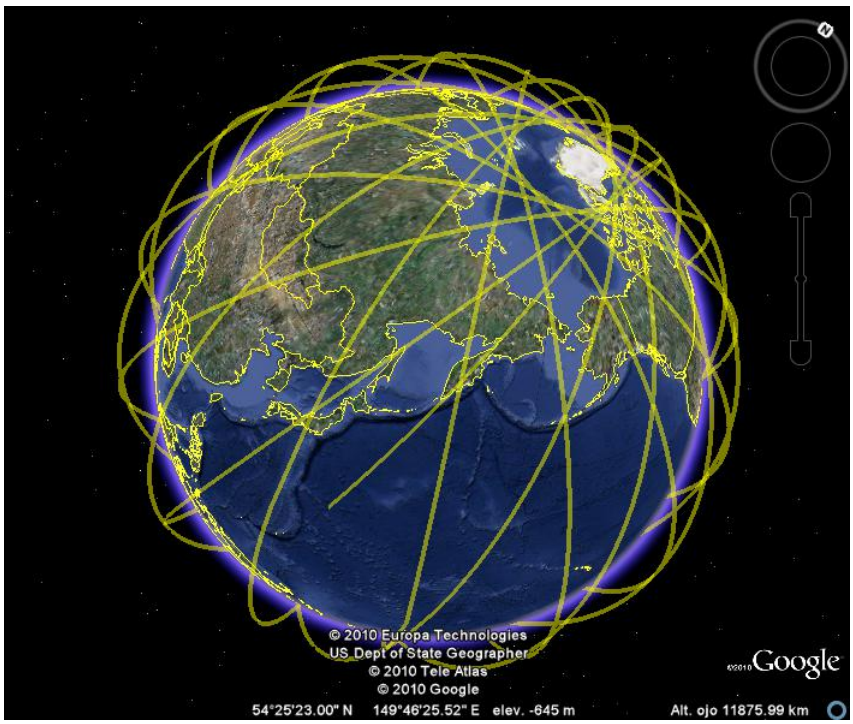
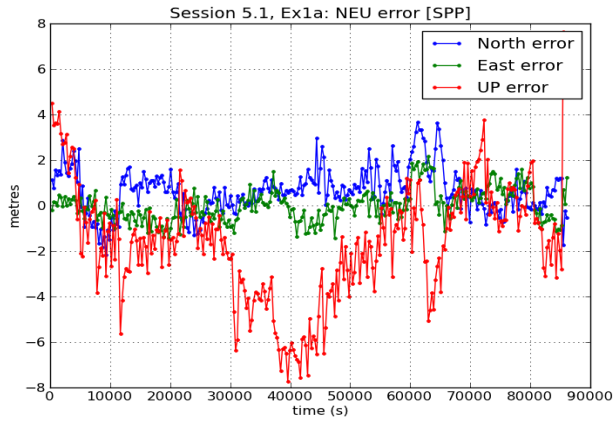
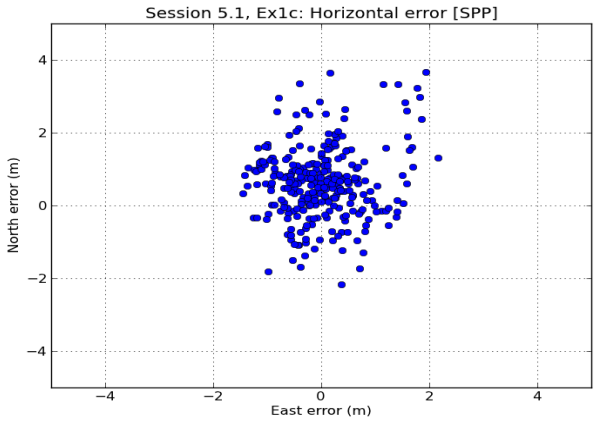


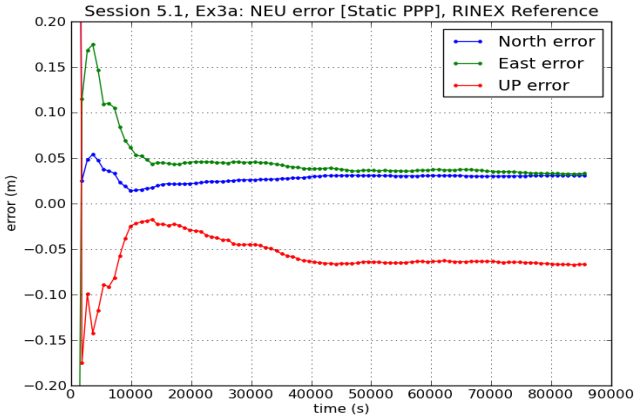
Figure 5.1: GRACE-A orbit view with Google Earth.

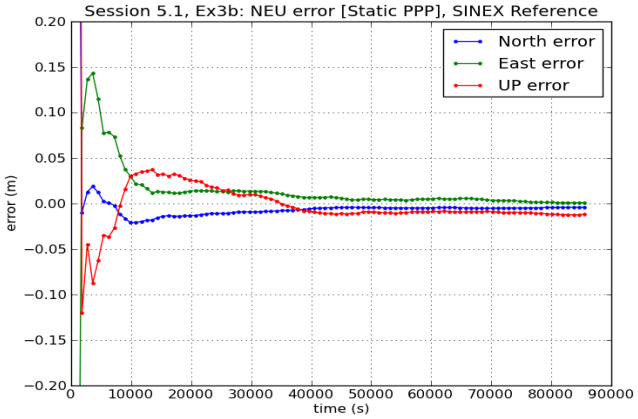
[44] Again, the same `WARNING` as in the previous two cases will appear when executing `gLAB`. Skip this warning and proceed with the data processing.

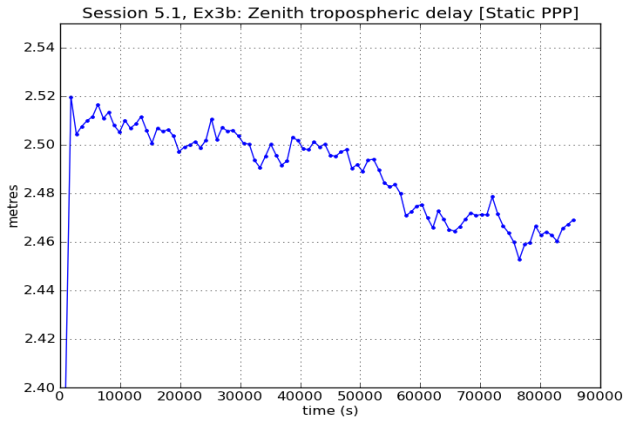
Graphs Session 5.1

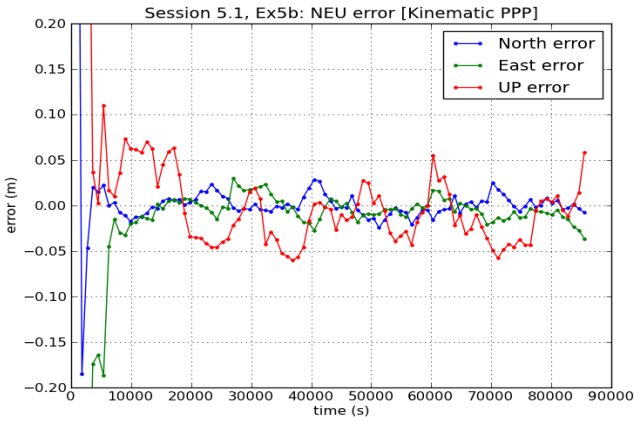


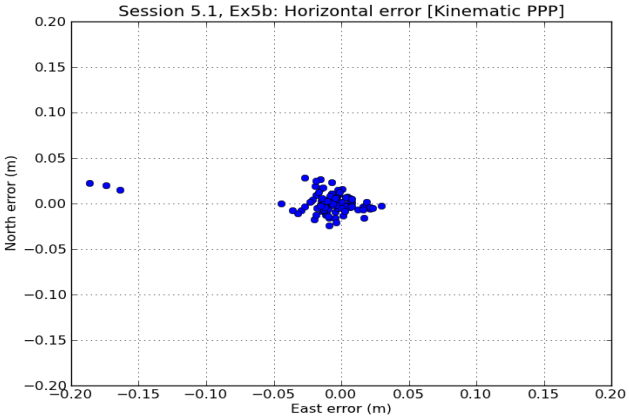


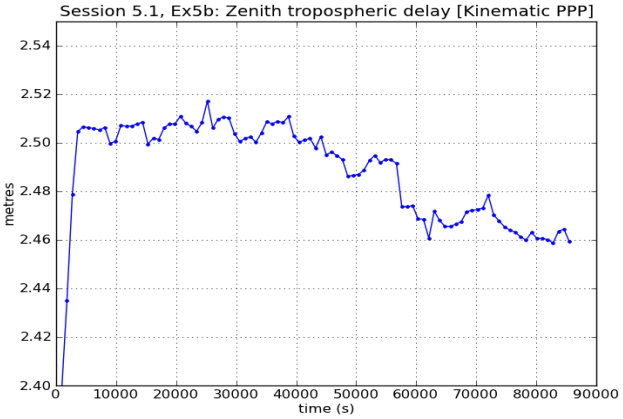


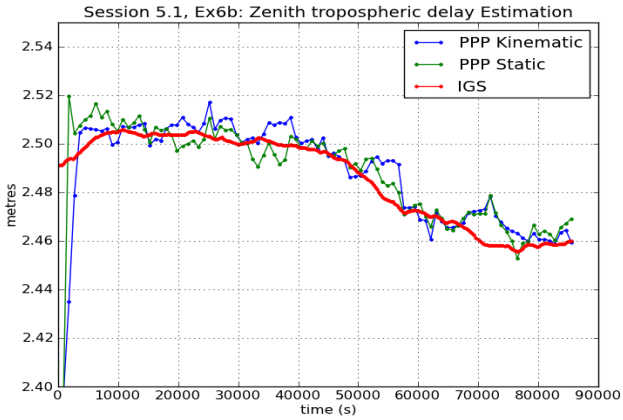


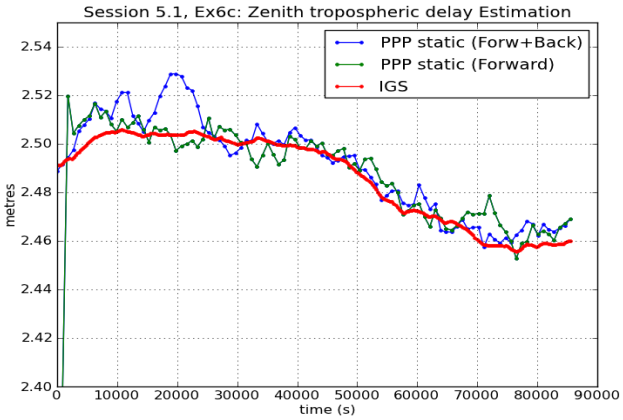


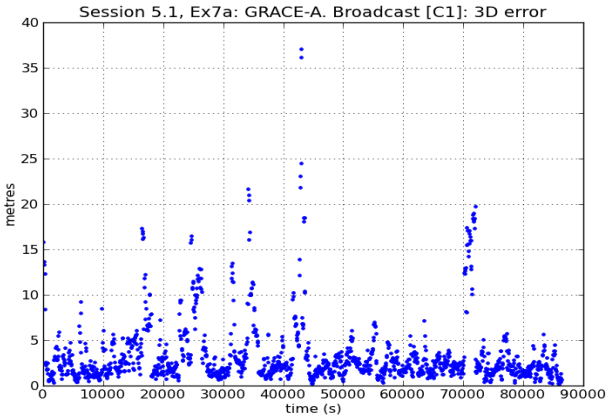


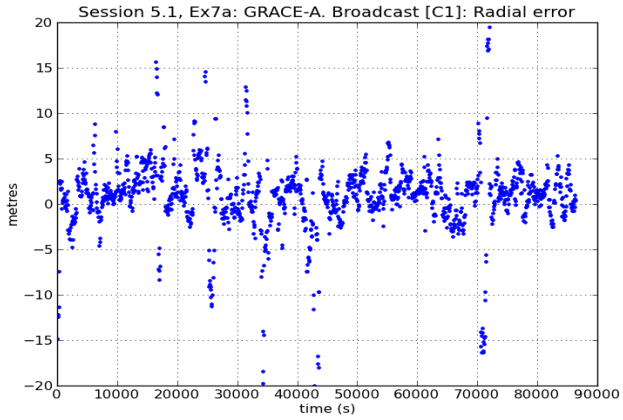


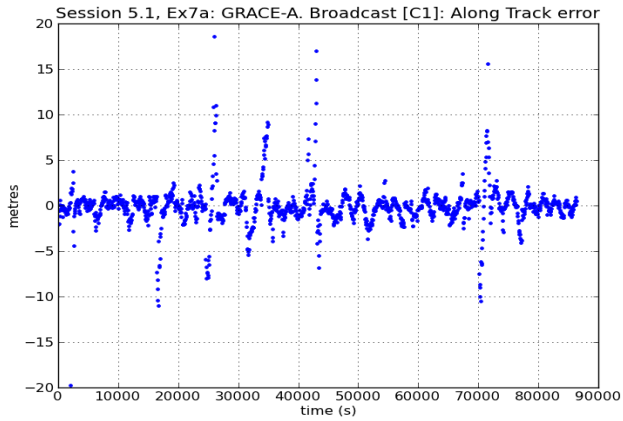


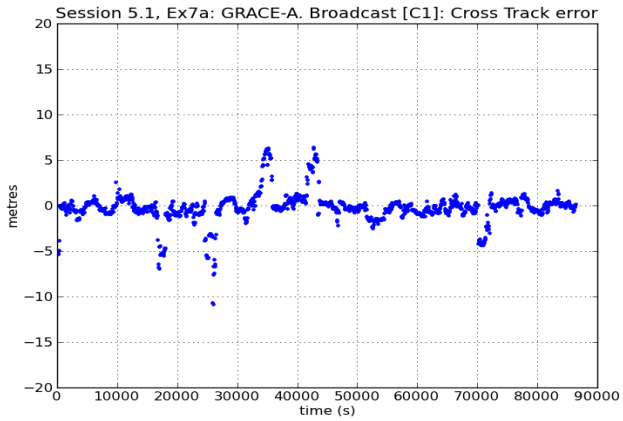

Session 5.1, Ex6b: Zenith tropospheric delay Estimation
PPP Kinematic
PPP Static
IGS
metres
time (s)

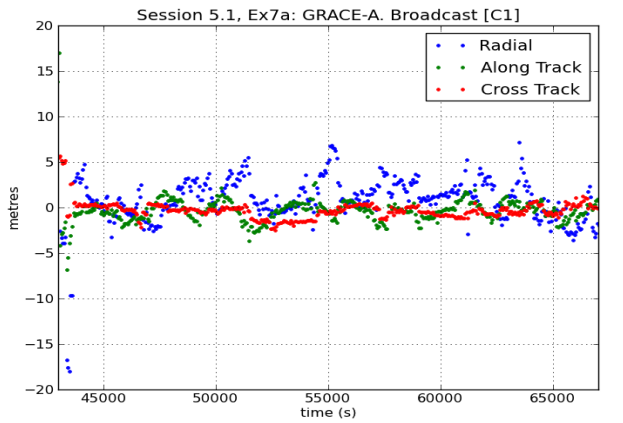

Session 5.1, Ex6c: Zenith tropospheric delay Estimation
PPP static (Forw+Back)
PPP static (Forward)
IGS
metres
time (s)

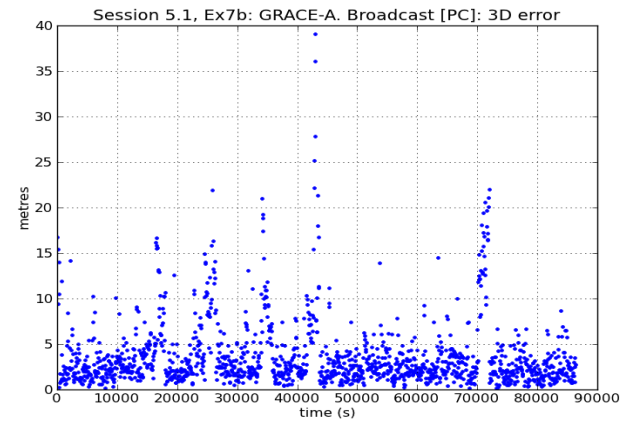

Session 5.1, Ex7a: GRACE-A. Broadcast [C1]: 3D error
metres
time (s)

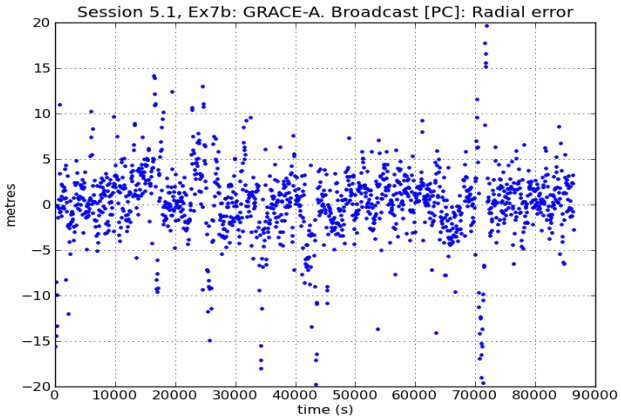

Session 5.1, Ex7a: GRACE-A. Broadcast [C1]: Radial error
metres
time (s)

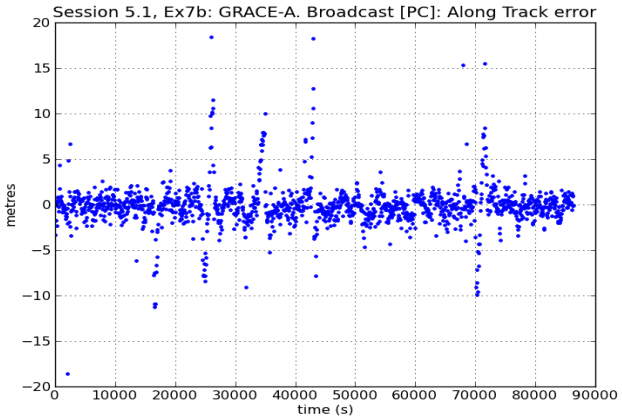

Session 5.1, Ex7a: GRACE-A. Broadcast [C1]: Along Track error
metres
time (s)

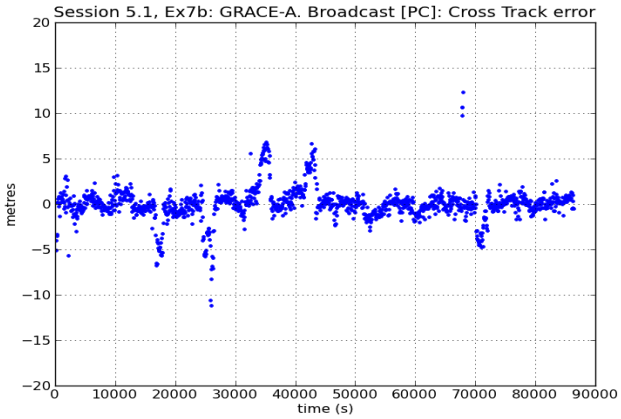

Session 5.1, Ex7a: GRACE-A. Broadcast [C1]: Cross Track error
metres
time (s)

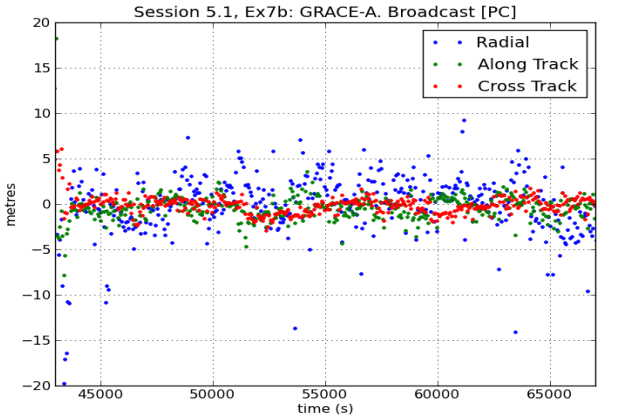

Session 5.1, Ex7a: GRACE-A. Broadcast [C1]
Radial
Along Track
Cross Track
metres
time (s)

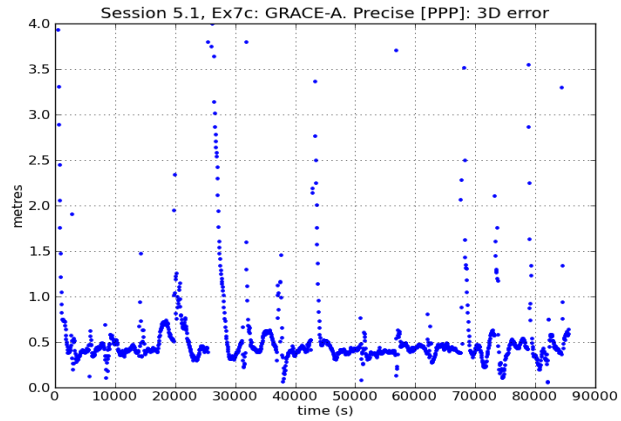

Session 5.1, Ex7b: GRACE-A. Broadcast [PC]: 3D error
metres
time (s)

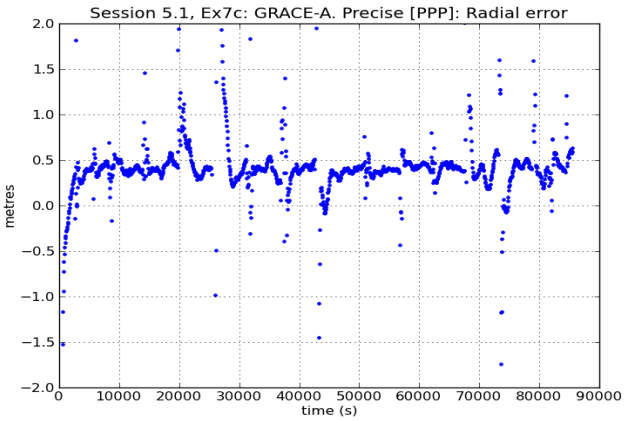

Session 5.1, Ex7b: GRACE-A. Broadcast [PC]: Radial error
metres
time (s)

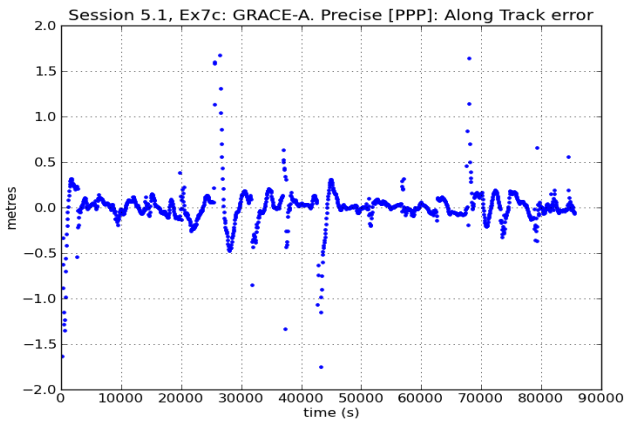

Session 5.1, Ex7b: GRACE-A. Broadcast [PC]: Along Track error
metres
time (s)

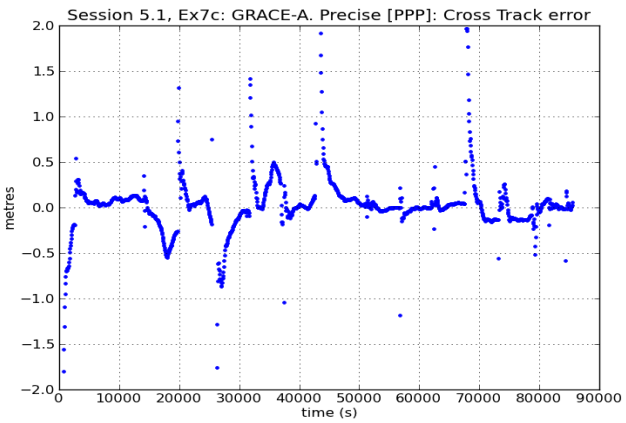

Session 5.1, Ex7b: GRACE-A. Broadcast [PC]: Cross Track error
metres
time (s)

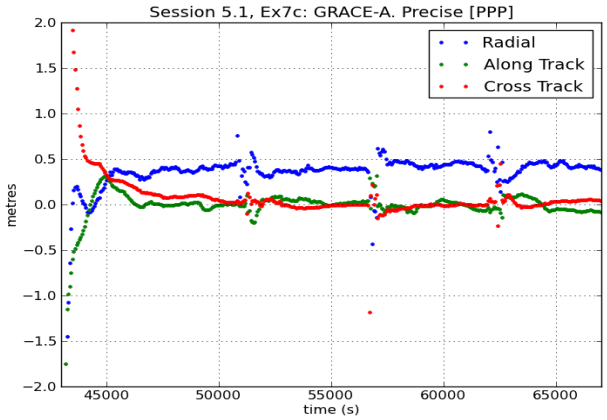

Session 5.1, Ex7b: GRACE-A. Broadcast [PC]
Radial
Along Track
Cross Track
metres
time (s)

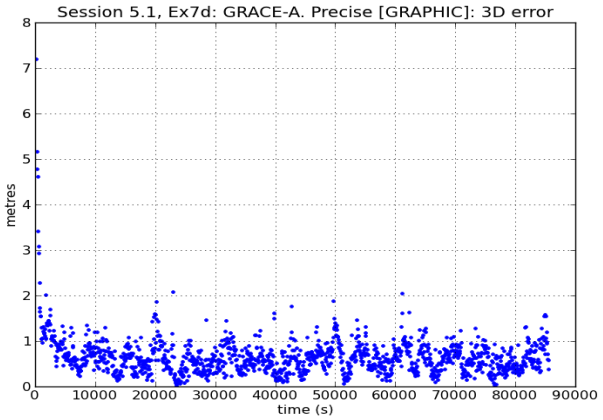

Session 5.1, Ex7c: GRACE-A. Precise [PPP]: 3D error
metres
time (s)

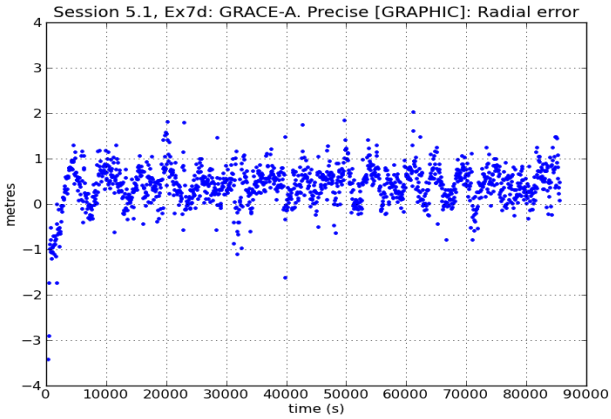

Session 5.1, Ex7c: GRACE-A. Precise [PPP]: Radial error
metres
time (s)

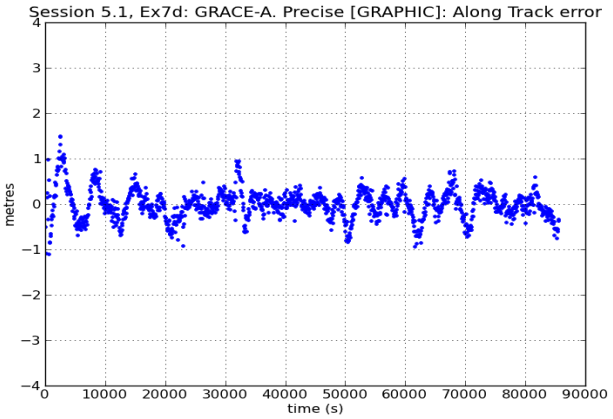

Session 5.1, Ex7c: GRACE-A. Precise [PPP]: Along Track error
metres
time (s)

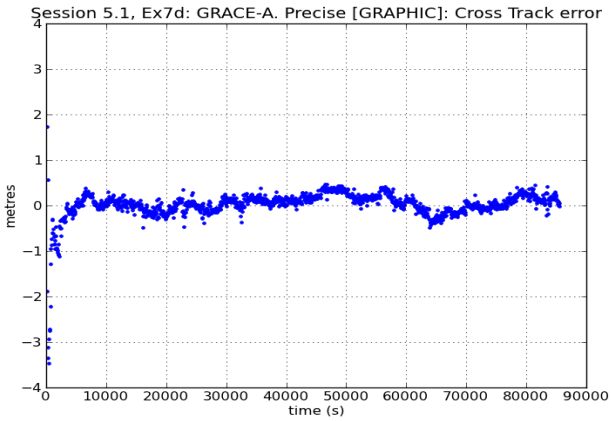

Session 5.1, Ex7c: GRACE-A. Precise [PPP]: Cross Track error
metres
time (s)

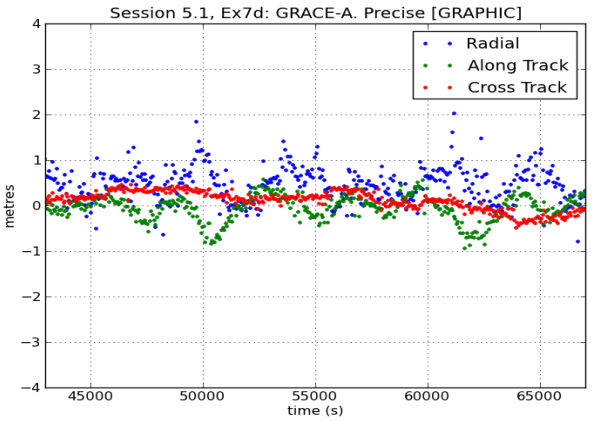

Session 5.1, Ex7c: GRACE-A. Precise [PPP]
Radial
Along Track
Cross Track
metres
time (s)

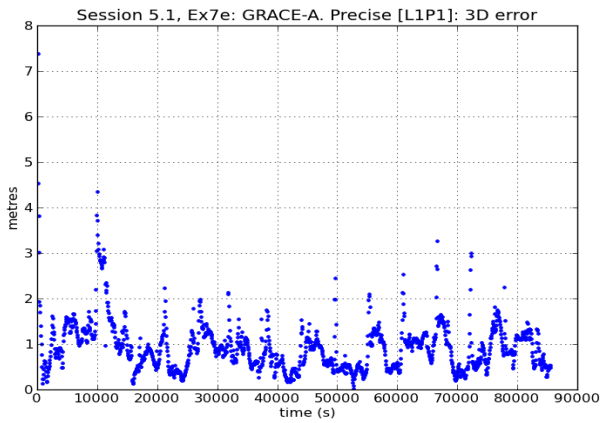

Session 5.1, Ex7d: GRACE-A. Precise [GRAPHIC]: 3D error
metres
time (s)

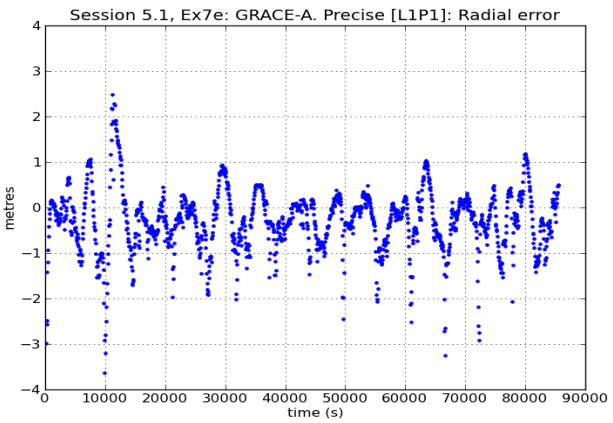

Session 5.1, Ex7d: GRACE-A. Precise [GRAPHIC]: Radial error
metres
time (s)

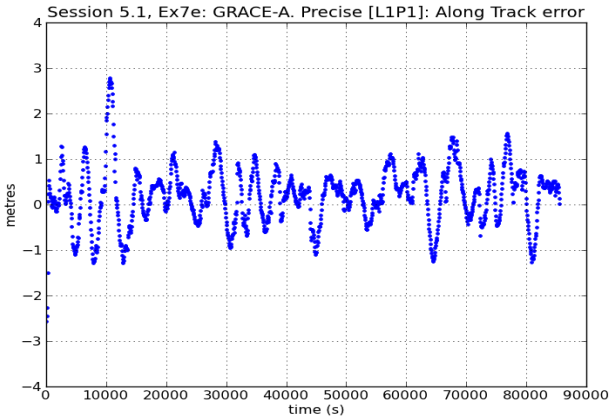

Session 5.1, Ex7d: GRACE-A. Precise [GRAPHIC]: Along Track error
metres
time (s)

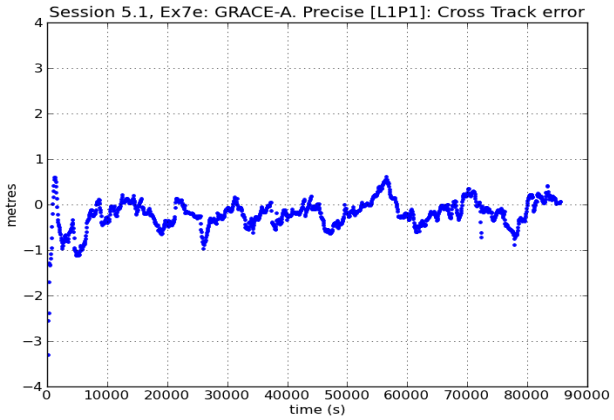

Session 5.1, Ex7d: GRACE-A. Precise [GRAPHIC]: Cross Track error
metres
time (s)

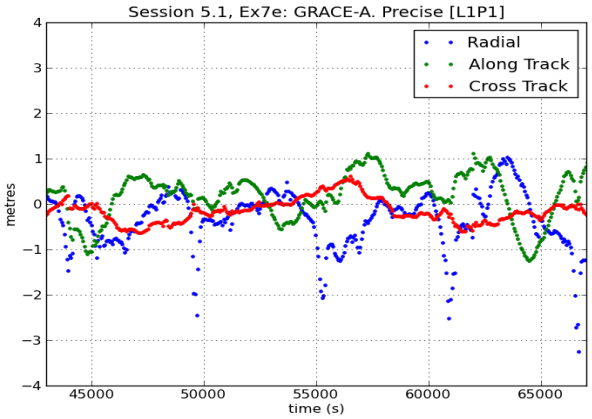

Session 5.1, Ex7d: GRACE-A. Precise [GRAPHIC]
Radial
Along Track
Cross Track
metres
time (s)

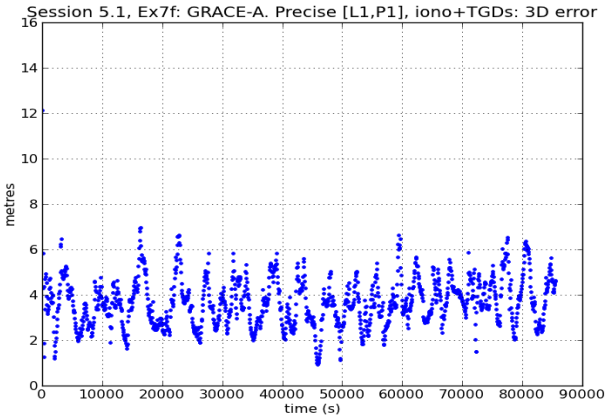

Session 5.1, Ex7e: GRACE-A. Precise [L1P1]: 3D error
metres
time (s)

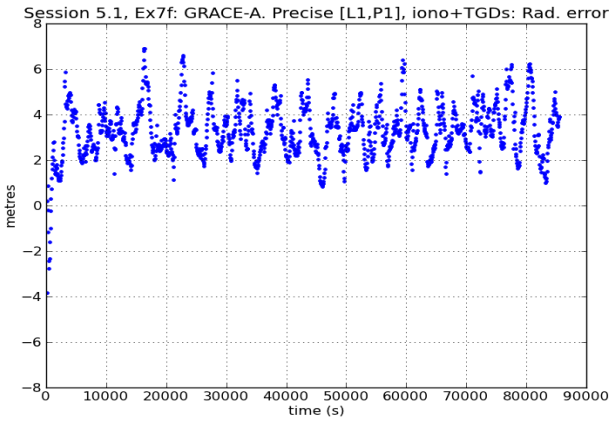

Session 5.1, Ex7e: GRACE-A. Precise [L1P1]: Radial error
metres
time (s)

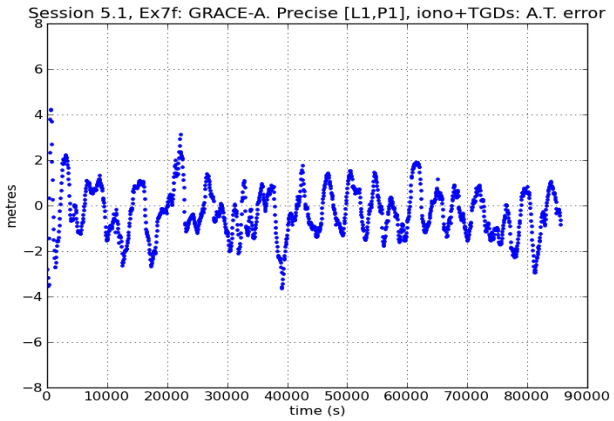

Session 5.1, Ex7e: GRACE-A. Precise [L1P1]: Along Track error
metres
time (s)

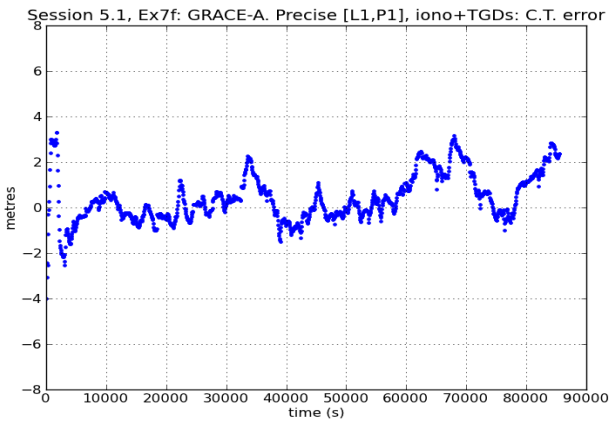

Session 5.1, Ex7e: GRACE-A. Precise [L1P1]: Cross Track error
metres
time (s)

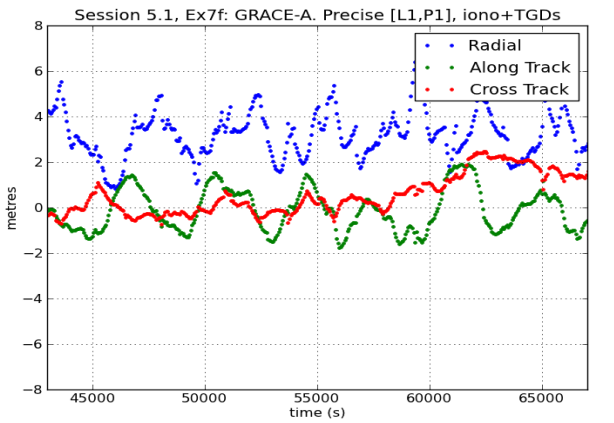

Session 5.1, Ex7e: GRACE-A. Precise [L1P1]
Radial
Along Track
Cross Track
metres
time (s)


Session 5.1, Ex7f: GRACE-A. Precise [L1,P1], iono+TGDs: 3D error
metres
time (s)


Session 5.1, Ex7f: GRACE-A. Precise [L1,P1], iono+TGDs: Rad. error
metres
time (s)


Session 5.1, Ex7f: GRACE-A. Precise [L1,P1], iono+TGDs: A.T. error
metres
time (s)


Session 5.1, Ex7f: GRACE-A. Precise [L1,P1], iono+TGDs: C.T. error
metres
time (s)


Session 5.1, Ex7f: GRACE-A. Precise [L1,P1], iono+TGDs
Radial
Along Track
Cross Track
metres
time (s)

Session 5.2. Model Component Analysis for SPP and PPP and Implementation of Algorithms

Objectives

To analyse in detail the measurement modelling for the SPP and PPP. In particular, to evaluate, in the range domain, the different components of code and carrier measurement modelling and its impact on the position domain.

To assess the effect of the P1–C1 DCBs on computation of the precise coordinates, receiver clock and tropospheric delay estimates.

To illustrate the implementation of algorithms by a 'step-by-step' example of measurement modelling and navigation equations solved with least squares and Kalman filtering.

Files to use
`UPC11490.05O`, `UPC11490.05N`, `roap1810.09o`, `igs15382.sp3`, `madr1960.10o`, `irtk1960.10o`, `darw1960.10o`, `brdc1960.10n`, `igs15924.sp3`, `igs15924.clk_30`, `igs15382.clk_30s`, `igs10P1592.snx`, `igs09P1538.snx`, `igs10P1585.snx`, `P1C11007.DCB`, `igs05_1525.atx`, `igs05_1585.atx` `GPS_Receiver_Types`

Programs to use
`gLAB_GUI.py`, `graph.py`, `GPSxyz.f`, `nwsw2cal.f`, `sub_trpUNB3.f`, `orbit.f`, `sub_klob.f`, `iono.f`, `tropo.f`, `car2geo.f`

Development

Session files:
Copy and uncompress these session files in the working directory:

```
cp ~/GNSS/PROG/SES52/* .
cp ~/GNSS/FILES/SES52/* .
gzip -d *.gz *.Z
```

1. Model components for SPP

This exercise focuses on describing in detail the modelling for the SPP. The effect of neglecting the different model components on the navigation solution will be assessed with `gLAB` using a data set collected by a receiver in a fixed location (static receiver).[45]

[45] The data set was collected with a NovAtel Millennium OEM's receiver, with Pinwheel Antenna (NovAtel 600), in Barcelona, Spain, on 29 May 2005 (DoY 149).

(a) *Data processing with* gLAB*: Full model for SPP*

i. Execute

`gLAB_GUI.py &`

to run the gLAB GUI.

The [Positioning] tab is opened by default. Click the [Input] button to open the input section panel.

ii. Select the **default options for SPP Template**. This is done by clicking the button

`SPP Template`

at the bottom of the GUI.

iii. In the [Input] section, click the button

`Examine`

for the RINEX observation file and in the WORK directory select the file UPC11490.05O.[46] Do the same for the RINEX navigation file[47] UPC11490.05N. The ANTEX file is not needed for the SPP template mode.[48]

iv. Click

`Run gLAB`

to compute the solution.

v. In the [Analysis] tab, click the buttom

`NEU positioning error`

and then click

`Plot`

to generate the figure. Note that this plot shows the discrepancies between the computed solution and the *A-priori receiver position* defined in the [Input] section of [Positioning] tab. By default gLAB uses the 'APPROX POSITION XYZ' of the RINEX file.[49]

vi. In the [Positioning] tab, click the button (in the bottom right corner)

`Show Output`

to edit the gLAB output file (named gLAB.out, by default). The previous plot is taken over the rows labelled 'OUTPUT' in this file.[50] It represents the columns 18 (North), 19 (East) and

[46] By default, the browser selects the files '*.??o', '*.??O' or '*.??n', '*.??N'. If the required file does not appear in the browser, then select the option "All files" to see what is available in the directory.

[47] In SPP mode, the button [⊙ Broadcast Orbits and Clocks] is selected by default.

[48] In the SPP, the satellite coordinates are computed from the broadcast navigation message and, thus, they are relative to the satellite APC (see sections 3.3.1 and 5.6.3, Volume I). On the other hand, the receiver coordinate estimates will be relative to the receiver APC for the L1 signal.

[49] *Remark*: The receiver coordinates of the RINEX file are not necessarily accurate. They must be taken only as an *approximated receiver position*. Nevertheless, to make the comparison easier, in this case, the L1 receiver APC coordinates have been written in the RINEX file as the receiver approximate position XYZ.

[50] The plotting configuration (file, axis, arguments, etc.) can be seen in the Global Graphic Parameters box in the [Analysis] tab panel.

20 (Up) versus column 4 (time).[51] The receiver coordinates, (X, Y, Z), computed in the ECEF system are given in columns 6, 7 and 8, respectively.

vii. In the `[Analysis]` tab, click the button

`Horizontal positioning error`

and then click

`Plot`

to generate a plot showing the discrepancies between the computed position and the approximate coordinates given in the RINEX file, in the horizontal plane. This plot represents column 18 (North) versus column 19 (East) of rows labelled 'OUTPUT' in file `gLAB.out`.

(b) *Relativistic clock correction analysis*

The relativistic clock correction due to the orbit eccentricity (see section 5.2.1, Volume I) can be assessed as follows:

i. Range domain effect:

In the `[Analysis]` tab, click the button

`Model Components`

and then click

`Plot`

to generate a plot showing the relativistic correction effect on the range measurements. This plot represents column 22 (relativistic clock correction) versus column 4 (seconds of day) of rows labelled 'MODEL' in file `gLAB.out`.

Note that this plot is configured by default in the *Templates* box of the `[Analysis]` tab, when clicking the button

`Model Components`

Other plots can be selected by clicking the symbol `▼` in the different plot options:

`Condition:` `MODEL ▼`,

`X Column:` `SEC ▼`,

`Y Column:` `RELCLOCK ▼`,

etc.

ii. Position domain effect:

Reprocess the previous data files, but without applying the relativistic clock correction. Proceed as follows:

A. In the `[Output]` section of the `[Positioning]` tab, rename[52] the output file as `gLAB1.out`.

B. In the `[Modelling]` section of the `[Positioning]` tab, uncheck the model component `[☐ Relativistic clock correction]`.

C. Click `Run gLAB` to compute the solution.

[51] The description of the different fields of this file can be obtained by placing the mouse over the different messages print options in the `[Output]` section of the `[Positioning]` tab.

[52] `gLAB.out` will be kept as a reference file, because it has the full modelling. The new results computed by unchecking the modelling options will be saved in the `gLAB1.out` file.

D. Plot the horizontal component effect. In the [Analysis] tab, click the plot option

Horizontal positioning error

and click the [⊙ Plot 2] button. Then, select gLAB.out as a source file[53] and configure the remaining options as in the case [⊙ Plot 1]:

Condition: OUTPUT ▼,

X Column: DSTAE ▼,

Y Column: DSTAN ▼,

and select

Circles ▼.

Finally, click

Plot.

E. Plot the vertical component effect. Click the plot option

NEU positioning error.

Then:

A. Click the [⊙ Plot 1] button[54] and then change DSTAN to DSTAU in the Y column, that is

Y Column: DSTAU ▼,

and select

Dotted Line ▼.

Change the Label to No rel. corr.

B. Click the [⊙ Plot 2] button and select gLAB.out as a source file. Then select DSTAU in the Y Column. Change the Label to Full model.

C. Click the [⊙ Plot 3] button and remove the gLAB1.out file in the Source file option.

D. Finally, click the button Plot to generate the plot.

(c) *Differential Code Bias* P1-P2*, or TGD*

The effect of the TGD in the SPP solution (see section 5.3, Volume I) is analysed as follows:

i. Range domain effect:

In the [Analysis] tab, click the button

Model Components[55]

and then select TGD for the Y Column. That is, set the option

Y Column: TGD ▼.

Finally, to generate the plot[56]click the button Plot.

[53] By default the last processed output file (i.e. gLAB1.out) is selected in this template for plotting in Plot1. Plot2 will have all its input boxes empty.

[54] The file gLAB1.out will be selected as a source file.

[55] Y Column: RELCLOCK ▼ is selected by default.

[56] *Comment:* The plot will be drawn using the file gLAB1.out generated in the previous run, but this is not a problem because the TGD corrections were computed in this run (only the relativistic clock correction was unchecked in that run).

ii. Position domain effect:
Set up again the default configuration for SPP template mode by clicking the button SPP Template in the [Input] section of the [Positioning] tab. The file gLAB1.out remains selected as the output file in the [Output] section.
Following the same procedure as in the previous analysis of relativistic clock correction, reprocess without applying the [P1-P2 correction] (i.e. the TGD correction) in the [Modelling] section. Generate the plots for the horizontal and vertical positioning error components as in the previous case.

(d) *Satellite clock offset effect*
Repeat the previous procedure to analyse the effect of the satellite clock offset correction (see section 5.2, Volume I) on the SPP solution. Note that this effect corresponds to [☒ Satellite clock offset correction] in the [Modelling] section. The option

Y Column: SATCLK ▼

must be selected for plotting.

(e) *Effect of ionospheric corrections*
Repeat the previous procedure to analyse the effect of the ionospheric corrections, computed from the Klobuchar model (see section 5.4.1.2.1, Volume I), on the SPP solution. Note that this effect corresponds to [☒ Ionospheric correction] in the [Modelling] section. The option

Y Column: IONO ▼

must be selected for plotting.
Disable [☐ Automatic Limits] and set

Y-min	0

to plot only the code corrections.

(f) *Effect of tropospheric corrections*
Repeat the previous procedure to analyse the effect of the tropospheric corrections (see section 5.4.2.1, Volume I) on the SPP solution. This effect corresponds to [☒ Tropospheric correction] in the [Modelling] section. The option

Y Column: TROP ▼

must be selected for plotting.

(g) *Effect of Earth's rotation during the signal flight time*
Repeat the previous procedure to analyse the effect of not taking into account Earth's rotation during the signal flight time (see section 5.1.1.2, Volume I) on the SPP solution. Note that this effect corresponds to [☒ Consider Earth rotation during the signal flight time] in the [Modelling] section.
The following procedure must be applied to make the plot of the range domain effect, as it cannot be generated directly by gLAB:
A. Computation:

i. Select the computed geometric range values for each epoch and satellite from the gLAB.out and gLAB1.out files.

Execute (in a single line):

```
cat gLAB.out gLAB1.out|gawk '{if ($1=="MODEL")
                        print $4,$6,$17}' > tmp.dat
```

ii. For each epoch and satellite compute the difference between the geometric ranges of the two files:

```
cat tmp.dat|gawk '{i=$1" "$2;if(length(r[i])!=0)
                  {dr[i]=r[i]-$3} else {r[i]=$3}}
      END{for (i in dr) print i,dr[i]}' > dr.dat
```

B. Plotting: In the [Analysis] tab click the button Clear to the right of the *Global Graphic Parameters* box. Then, click the Examine button and select dr.dat as a Source file.[57] Select the fields 1 and 3 of file dr.dat for plotting, by setting:

X Column: ▼ 1 ▼

Y Column: ▼ 3 ▼

Select the Dots ▼ option.

Comment: To select a given satellite, for instance PRN16, the following condition can be applied:

Condition: ▼ ▼ ($2==16)

(h) *Effect of taking the satellite coordinates in reception time instead of emission time*

Repeat the same procedure as in the previous question to analyse the effect of not taking into account the satellite's and Earth's motion during the signal flight time (see section 5.1.1.1, Volume I) on the SPP solution.

Note that the following model components must be unchecked in the [Modelling] section to analyse this effect:

[☐ Consider satellite movement during signal flight time]

[☐ Consider Earth rotation during signal flight time]

The same procedure as in exercise (1g) can be applied.

2. Model components for kinematic PPP

This exercise focuses on describing in detail the modelling for PPP. The impact of the small range corrections (additional to those analysed in the previous exercise for SPP) on the position domain will be assessed with gLAB, using a data set collected with a receiver in a fixed location (static receiver).[58] Although the receiver's position was kept fixed during the data collection (as in the previous exercise), its coordinates will be processed in kinematic mode to better assess the effect of modelling errors on the user solution.

[57] By default, the browser shows only the files '*.out', with lower or upper case letters. Thus the option "All files" at the bottom corner of the browser must be set to see the file dr.dat.

[58] This data set was collected with a Septentrio PolaRx2 receiver, with a SEN67157596+CR antenna, in Cádiz, Spain, on 30 June 2009 (DoY 181).

(a) *Data processing with* gLAB*: Full model for PPP*

i. To run the gLAB GUI execute:

`gLAB_GUI.py &`

The [Positioning] tab is opened by default. Click the [Input] button to open the input section panel.

ii. Select the **default options for PPP Template**. This is done by clicking the button `PPP Template` in the [Input] section.

iii. In the [Input] section, click the button `Examine` for the RINEX observation file and select the file `roap1810.09o`.
Do the same for the SP3 file[59] `igs15382.sp3` and the ANTEX file `igs05_1525.atx`.[60]
In the [Input] section, select the [⊙ Use SINEX File] option to get the 'A priori receiver position' from the SINEX file[61] `igs09P1538.snx`.
In the [Filter] section, select the [⊙ Kinematic] option to compute the navigation solution in a 'pure kinematic mode' (i.e. as a white-noise process), see section 6.1.2.1, Volume I.

iv. In the [Output] section set `gLAB.out` as the output file. As in the previous exercise, this file will contain the results with the full modelling.

v. Click `Run gLAB` to compute the solution.

vi. Apply the same procedure as in the previous exercise to produce the plots for the different cases of model components analysis indicated as follows.
Note that the output file should be named `gLAB1.out` when computing the solution with the different model components unchecked.

(b) *Wind-up correction*
Analysis of the effect of the carrier phase wind-up correction (see section 5.5, Volume I) on the Kinematic PPP solution. This correction corresponds to selecting the model component [☒ Wind up correction] in the [Modelling] section. It is activated by default for the PPP template mode. The option

Y Column: `WINDUP` ▼

must be selected for plotting.

(c) *Solid tides correction*
Analysis of the effect of solid tides corrections (see section 5.7.1, Volume I) on the kinematic PPP solution. This correction corresponds to the model component [☒ Solid Tides correction] in the [Modelling] section. It is activated by default for the PPP template mode. Select the option for plotting

Y Column: `SOLIDTIDES` ▼

[59] The button [⊙ Precise (1 file)] is selected by default.
[60] The `igs05_1525.atx` file is the ANTEX file associated with the `igs15382.sp3` orbit and clock file. Nevertheless, a newer file can be used, as well.
[61] SINEX files provide surveyed coordinates accurate to centimetre level or better.

(d) *Satellite APC correction*
Analysis of the effect of the satellite MC–APC correction (see section 5.6.3, Volume I) on the kinematic PPP solution. Note that the SP3 satellite precise orbits used are given relative to the MC and, hence, this correction must be applied.
The correction corresponds to the [☒ Satellite mass centre to antenna phase centre correction] in the [Modelling] section. It is activated by default for the PPP template mode. The satellite's APC parameters are taken from the ANTEX file. The option
Y Column: SATPCOPROJ ▼
must be selected for plotting.

(e) *Receiver APC correction*
Analysis of the effect of the receiver ARP–APC) correction (see section 5.6.2, Volume I) on the kinematic PPP solution. The receiver's APC values are taken from the ANTEX file.
This correction corresponds to the [☒ Receiver antenna phase centre correction] in the [Modelling] section. It is activated by default for the PPP template mode. The option
Y Column: RECPCOPROJ ▼
must be selected for plotting.

(f) *Relativistic path range correction*
Analysis of the effect of the relativistic path range correction (see section 5.1.2, Volume I) on the kinematic PPP solution.
This correction corresponds to model component [☐ Relativistic path range correction] in the [Modelling] section. It is activated by default for PPP. The option
Y Column: SATRECGEOMDIST ▼
must be selected for plotting.

(g) *Tropospheric delay estimation effect*
Analysis of the impact on the navigation solution of not estimating the wet tropospheric delay.[62]
Set the default configuration for PPP mode by clicking the button PPP Template. Disable the option [☐ Estimate Troposphere] in the [Filter] options. As in the previous case, represent on the same plot the results using the full model (i.e. file gLAB.out) and without estimating the tropospheric correction (i.e. file gLAB1.out).

3. GPS P1–C1 DCBs: Effect on receiver position and clock estimate

The impact of using the P1–C1 DCBs on receiver coordinates, troposphere and clock estimates will be assessed in this exercise. Before starting the analysis, some issues regarding the different GPS receiver types must be considered; for more details see Volume I, section 5.3.1.

As explained in Volume I, section 2.2.1, under A/S conditions, P codes (P1, P2) are encrypted to (Y1, Y2) for unauthorised users. Nevertheless,

[62] The residual relative to the nominal value given by the tropospheric model used in PPP, equation (5.66), Volume I.

RINEX files can be found from commercial receivers containing P1, P2 measurements among the C1. The generation of such P1, P2 codes under A/S condition depends on the code tracking technology, which must be taken into account for correct DCB handling.

Three different kinds of receivers are considered by IGS (for more details see [Schaer, S. and Steingenberger, P., 2006]):

Type 1: Cross-correlated receiver
The C1 and P2 measurements must be corrected by the $DCB_{P1\text{-}C1}$:

$$\boxed{\begin{array}{l} C1_{raw} + DCB_{P1-C1} \longrightarrow P1, \\ P2_{raw} + DCB_{P1-C1} \longrightarrow P2 \end{array}}$$

Type 2: Receivers reporting C1 in place of P1
The C1 must be corrected by the $DCB_{P1\text{-}C1}$:

$$\boxed{C1_{raw} + DCB_{P1-C1} \longrightarrow P1}$$

Type 3: Receivers reporting L1, L2, P1, P2 as a consistent set
No bias removal is needed.

`gLAB` uses `GPS_Receiver_Type` file[63] to identify the receiver type and the `P1C1YYMM.DCB`[64] file for applying the P1–C1 DCB corrections. It works by default in a flexible P1–C1 handling mode to perform PPP under not high demanding requirements (allowing data processing when such files are not available). Moreover, strict DCB handling is also implemented to ensure that the IGS conventions for DCB handling are applied.

The main features of the P1–C1 DCB correction modes are given next:

- P1–C1 DCB correction SET in `Flexible` mode: `gLAB` uses whatever C1 or P1 measurement is available in the receiver, identifying P1=C1, but without correcting DCBs. If both code measures are available, P1 is used as the default to compute the ionosphere-free combination codes PC.
- P1–C1 DCB correction SET in `Strict` mode: `gLAB` identifies the receiver type from file `GPS_Receiver_Type` file and applies the P1–C1 DCB correction form file `P1C1YYMM.DCB`, if needed, to get consistent C1, P1 and P2 codes.
  `gLAB` stops if either of these two files is not available or the receiver cannot be matched with the `GPS_Receiver_Type` file.
- P1–C1 DCB correction UNSET: gLAB does not identify P1=C1. Thus, it only computes the ionosphere-free combination code PC when the P1 (not C1) code is available (among the P2 code).[65]

Examples of the computation of the PPP solution using the default configuration of `gLAB` for PPP mode and using the *Strict* P1–C1 DCB correction handing option are provided as follows. Solution accuracies will be also compared for coordinates and clocks.

[63] This file is from the Jet Propulsion Laboratory (JPL) server *ftp://sideshow.jpl.nasa.gov/pub/gipsy_products/gipsy_params*.

[64] These files are available in server *ftp://ftp.unibe.ch/aiub/CODE/*.

[65] Note that the P1 code is required to detect cycle slips in the GPS L1 signal. Therefore, if this code is not provided by the receiver, no cycle-slip detection can be undertaken and no PC combination is computed.

(a) *Example of a receiver reporting C1 instead of P1: Static PPP*
The RINEX file `darw1960.10o` contains measurements from a Leica GRX1200GGPRO receiver. According to file `GPS_Receiver_Types`, this is a receiver reporting C1 instead of P1 (i.e. C1–P1 Receiver Type 2) and therefore needs the C1 measurements to be corrected by $DCB_{P1\text{-}C1}$.

Complete the following steps:

i. Compute the `PPP` solution using the default configuration[66] of `gLAB`, but setting the `Precise (2 files)` option in the `[Input]` section. Use the precise GPS orbit and clock files `igs15924.sp3` and `igs15924.clk_30s`, and the ANTEX file `igs05_1585.atx`.

A. In the `[Input]` section, set option `SINEX file` in the receiver `A priori position` and select file `igs10P1585.snx`.
B. In the `[Modelling]` section, check if the *Flexible* mode for the `[☒ P1-C1 correction]` option[67] is set.
C. In the `[Output]` section, set the output file as `gLAB_def.out`.
D. Click the button `Run gLAB` to compute the solution.

Plot the (E,N,U) positioning error and the receiver clock estimate.

ii. Set the *Strict* `P1-C1 correction` option in the `[Modelling]` section and recompute the `PPP` solution. In the `[Input]` section select the precise `P1C11007.DCB` file and `GPS_Receiver_Types` file.

A. In the `[Output]` section, set the output file as `gLAB_str.out`.
B. Click the button `Run gLAB` to compute the solution.

Plot the (E,N,U) positioning error and the receiver clock estimate.

iii. Plot the discrepancies between the receiver coordinates, troposphere and clock from files `gLAB_def.out` and `gLAB_str.out`. Discuss why DCBs mostly affect the receiver clock estimate.

Execute for instance:

A. Receiver coordinates and troposphere discrepancy:

Compute the discrepancy between the different estimates:

```
grep OUTPUT gLAB_def.out|grep -v INFO|
         gawk '{print $4,$18,$19,$20,$30}' > def.tmp

grep OUTPUT gLAB_str.out|grep -v INFO|
         gawk '{print $4,$18,$19,$20,$30}' > str.tmp

cat def.tmp str.tmp | gawk '{if (length(e[$1])==0)
           {e[$1]=$2;n[$1]=$3;u[$1]=$4;t[$1]=$5}else
  {print $1,$2-e[$1],$3-n[$1],$4-u[$1],$5-t[$1]}}'
                                      > str_def.enut
```

[66] This default configuration involves the static PPP template and the flexible handling of DCBs. Note that, as the `GPS_Receiver_Types` or `P1C1YYMM.DCB` files are not used in the `Flexible` mode, then the P1-C1 correction will not be applied.

[67] When option `[☐ P1-C1 correction]` is unchecked, `gLAB` only computes the PC combination if both P1 and P1 codes are available in the RINEX header. For instance, PC will not be computed in this case, because P1 is not available in the RINEX file.

Plot the results for the coordinates:

```
graph.py -f str_def.enut -x1 -y2  -l "North error"
         -f str_def.enut -x1 -y3 -l "East error"
         -f str_def.enut -x1 -y4 -l "UP error"
         --yn -0.02 --yx 0.02 --xl "time (s)" --yl "(m)"
```

Plot the results for the troposphere:

```
graph.py -f str_def.enut -x1 -y5  -l "troposphere"
         --yn -0.02 --yx 0.02 --xl "time (s)" --yl "(m)"
```

B. Receiver clock discrepancy:

Compute the discrepancy between receiver clock estimates:

```
grep FILTER gLAB_def.out | grep -v INFO|
                         gawk '{print $4,$8}' > def.tmp

grep FILTER gLAB_str.out | grep -v INFO|
                         gawk '{print $4,$8}' > str.tmp

cat def.tmp str.tmp | gawk '{if (length(c[$1])==0)
       {c[$1]=$2} else {print $1,$2-c[$1]}}' > str_def.clk
```

Plot the results for the clock:

```
graph.py -f str_def.clk -s.-  --yn -0.3 --yx 0.9
                              --xl "time (s)" --yl "(m)"
```

iv. Compare the receiver clock estimates of files `gLAB.out_def` and `gLAB.out_str` with the IGS estimate of file `igs15924.clk_30s`.

Comparing clock estimates from `gLAB` and IGS:

Extract the IGS clock estimates for the DARW receiver:

```
grep DARW igs15924.clk_30s| gawk 'BEGIN{c=299792458}
            {print $6*3600+$7*60+$8,$10*c}' > darw_igs.clk
```

Plot the results:

```
graph.py -f gLAB_def.out -x4 -y8 -c '($1=="FILTER")'
     -l"Def" -f gLAB_str.out -x4 -y8 -c '($1=="FILTER")'
     -l"Str" -f darw_igs.clk -x1 -y2  -l "IGS Clock_rec"
                  --xn 30000 --xx 55000 --yn -4 --yx 4
```

Plotting discrepancies against IGS determinations:

Extract the IGS clock estimates for DARW receiver:

```
cat darw_igs.clk def.tmp|gawk '{if (length(c[$1*1])==0)
   {c[$1*1]=$2} else {print $1,$2-c[$1*1]}}' > def_igs.clk

cat darw_igs.clk str.tmp|gawk '{if (length(c[$1*1])==0)
   {c[$1*1]=$2} else {print $1,$2-c[$1*1]}}' > str_igs.clk
```

Plot the results:

```
graph.py -f def_igs.clk -l "No DCBs" -s.-
         -f str_igs.clk -l "Strict DCBs" -s.
         --yn -1  --yx 1 --xl "time (s)" --yl "(m)"
```

(b) *Example of a receiver reporting C1 instead of P1: Kinematic PPP*
Repeat the previous processing, but in kinematic mode. Similar conclusions will be obtained. Note that, in kinematic mode, the coordinates are treated as white noise, like the receiver clock. Nevertheless, as in the previous case, where the coordinates were set as constants, the DCBs' mismodelling mostly affects the receiver clock and has a very low impact on the coordinates.

(c) *Example of a cross-correlated receiver: Static PPP*
The RINEX file `irkt1960.10o` contains measurements of a ROGUE SNR-8000 receiver. According to file `GPS_Receiver_Types`, this is a cross-correlated receiver (i.e. C1–P1 Receiver Type 1) and therefore needs the C1 and P2 measurements to be corrected by $DCB_{P1\text{-}C1}$.

Comment: There is a mistake in the `# / TYPES OF OBSERV` list given in the RINEX header. The following list is written

```
     7    C1    L1    L2    P2    P1    S1    S2
```

when it should be

```
     6    C1    L1    L2    P2    S1    S2
```

That is, no P1 code is provided, as erroneously indicated.

Notice that only six measurements are written in the body text. See for instance:

```
10  7 15  0  0  0.0000000  0  8G24G27G09G12G15G25G22G18
 21699141.9294   -8448919.36548  -6583574.85643  21699143.3344
       48.0004          22.1004
 22502118.9554  -11123955.71447  -8668016.79443  22502119.5034
       45.7004          18.6004
```

This issue must be taken into account when processing with `gLAB`, because it uses this header information to identify the RINEX file content.

Complete the following steps:

i. Edit file `irkt1960.10o` and fix the mistake in the measurements list (i.e. `# / TYPES OF OBSERV`) as indicated above. Care must be taken to respect the format file; that is, in respecting the separation between the fields. The file `irkt1960_edt.10o` corresponds to the previous file, but with the header mistake fixed.

ii. Repeat the same analysis as in the previous exercise 3a, using the file `irkt1960_edt.10o`. Compare the results.
As in the previous exercise, plot results for the (E,N,U) positioning error, the ZTD and the receiver clock estimate.

4. P1–C1 DCB assessment

The $DCB_{P1\text{-}C1}$ will be estimated in this exercise from Ashtech Z-X12 receiver measurements (i.e. a type 3 receiver with consistent C1, P1 and P2 codes). The result will be compared with different determinations of public domain data files.

(a) Using the data file `madr1960.10o` collected by an Ashtech Z-XII3 receiver, compute the mean value of P1–C1 bias.

Complete the following steps:

i. Using the configuration file `meas.cfg`, generate the MEAS file `madr1960.10.meas` with the following content:

```
[MEAS YY DoY sec GAL PRN el Az N list C1C L1C C1P L1P C2P L2P]
[  1   2  3   4   5   6   7  8 9  10   11  xx  13  14  15  16]
```

Execute for instance:

```
gLAB_linux -input:cfg meas.cfg -input:obs madr1960.10o
                                    > madr1960.10.meas
```

ii. Compute the mean value of the difference of code measurements P1 and C1. Execute for instance:

```
cat madr1960.10.meas|
 gawk '{n[$6]++;p1c1[$6]=p1c1[$6]+$13-$11}
        END{for(i in p1c1)print i,p1c1[i]/n[i]}'
                                    | sort -n > P1C1.dat
```

(b) Compare the previous determinations with those of file `P1C11007.DCB` from IGS. Complete the following steps:

i. Select the DCBs of GPS satellites from file `P1C11007.DCB` and change the units from nanoseconds to metres of delay.
Execute for instance:

```
grep G P1C11007.DCB| gawk 'BEGIN{c=299792458}
  {if(NF==3)print substr($1,2,3),$2*c*1e-9}'> p1c1.igs
```

ii. Plot the results. Execute for instance:

P1–C1 DCBs:

```
graph.py -f P1C1.dat -x1 -y2 -so- -l "P1-C1 from rcv"
         -f p1c1.igs -x1 -y2 -so- -l "P1-C1 from IGS"
--yn -3 --yx 3 --xl "PRN" --yl "metres of L1-L2 delay"
```

5. Calculation of modelled pseudorange and prefit residuals

Using files `UPC11490.05O` and `UPC11490.05N` compute the SPP solution as in exercise 1. Afterwards, calculate *by hand* the modelled C1 pseudorange and the prefit residual for satellite PRN25 at time $t = 300$ seconds of day 29 May 2005 (DoY 149) and compare the results with `gLAB`.

Complete the following steps for the calculations by hand:

(a) *Broadcast navigation message selection*
For PRN25, select from file `UPC11490.05N` the last transmitted navigation message, before $t = 300$ seconds of DoY 149 of year 2005.

Solution:

The last data record transmitted by PRN25 before $t = 300\,\mathrm{s}$ is:

```
25 5  5 29  2  0  0.0 9.401096031070E-05 9.094947017729E-13 0.000000000000E+00
   8.400000000000E+01-1.061875000000E+02 4.825915304457E-09-2.255215633503E+00
  -5.284324288368E-06 1.204112719279E-02 5.686655640602E-06 5.153704689026E+03
   7.200000000000E+03 2.011656761169E-07-2.689273653419E+00 1.396983861923E-07
   9.492799505545E-01 2.625625000000E+02-1.460408709553E+00-8.100337411567E-09
  -3.643008888800E-11 1.000000000000E+00 1.325000000000E+03 0.000000000000E+00
   2.800000000000E+00 0.000000000000E+00-7.450580596924E-09 8.520000000000E+02
   1.800000000000E+01 0.000000000000E+00 1.000000000000E+00 0.000000000000E+00
```

These data were transmitted by PRN25 at second 18 of GPS week 1325 (i.e. `1.325000000000E+03`, `1.800000000000E+01` in the message).

The associated `YY:MM:DD:hh:mm:ss` with this transmission time can be computed using program `nwsw2cal` as follows:

```
echo 1325 18| nwsw2cal
```

(b) *Satellite clock offset*

Compute the satellite clock offset from navigation message data at $t = 300$s (see equation (5.17), Volume I).

Solution:

- The satellite clock offset $\widetilde{\delta t}^{sat}$ is computed with the polynomial $\widetilde{\delta t}^{sat} = a_0 + a_1\,(t - t_0) + a_2\,(t - t_0)^2$.

  From the previous data block: $t_0 = 2\,\text{h}\ 0\,\text{min}\ 0\,\text{s} = 7200\,\text{s}$, $a_0 =$ `9.401096031070E-05` $a_1 =$ `9.094947017729E-13`, $a_2 =$ `0.000000000000E+00` (use also $c = 299\,792\,458\,\text{m/s}$).

  Hence: $\boxed{\widetilde{\delta t}^{sat} = 9.400\,468\,48 \cdot 10^{-5}\,\text{s}} \Longrightarrow c\,\widetilde{\delta t}^{sat} = 28\,181.895\,51\text{m}.$

- The values computed by `gLAB` can be found by processing the RINEX files `UPC11490.05O` and `UPC11490.05N` with the default configuration of `SPP` mode, as in exercise 1.
  Consider `gLAB.out`, the output file. Then execute:

```
grep MODEL gLAB.out | grep -v INFO |
    gawk '{if ($6==25) print $4,$6,$18}' |head -1
```

(c) *Satellite TGD*

Select the TGD value from the navigation message.

Solution:

- `TGD = -7.450580596924E-09 (in seconds)`
  Thus: $\boxed{\text{TGD} = -7.450580596924\text{E} - 09 \times c = -2.233\,63\,\text{m}}$.

- The values computed by `gLAB` can be found by:

```
grep MODEL gLAB.out | grep -v INFO |
    gawk '{if ($6==25) print $4,$6,$27}' |head -1
```

(d) *Satellite coordinates at transmission time*

Calculate the coordinates of satellite PRN25 at signal transmission time, applying the pseudorange-based algorithm (section 5.1.1.1, Volume I). Complete the following steps:

i. Compute the transmission time (T) from the reception time $t = 300\,\text{s}$ (RINEX time tags), using equation (5.6), Volume I.

Solution:

$$T = 300 - (1/c)\,\texttt{C1} - \widetilde{\delta t}^{sat} = 299.923\,662\,24\,s$$

where `C1` is the code pseudorange measurement for PRN25 at the RINEX time tag $t = 300\,\text{s}$.

Note that, from file `UPC11490.05O`, `C1` = 22857303.996 m at $t = 300$ s.

```
5  5 29  0  5  0.0000000  0  9G25G 9G 6G 1G21G 2G 5G30G14
22857303.996    22857301.3054  120115969.49948  93596862.76546     2723.29048
    2122.09146
24466601.337    24466601.6684  128572940.94147 100186651.00844    -3729.38047
   -2905.98944
20405995.011    20405993.9894  107234297.78349  83559175.41846     1058.26649
     824.62446
22758443.914    22758442.9824  119596458.09448  93192027.40946      221.51848
     172.61946
```

ii. Compute the satellite coordinates at transmission time, for instance with the help of program `GPSxyz.f`, as follows:

A. Generate the file `eph.dat` with the already selected broadcast message data of PRN25 (see file `GPSxyz.f` header).[68]

B. Execute:

```
echo "2005 149 299.92366224" > time.dat
cat time.dat eph.dat | GPSxyz
```

Solution:

$$\widetilde{\mathbf{r}}^{sat} = [6\,364\,868.618, -14\,298\,233.062, 21\,851\,197.941]\,m\,.$$

Note that these coordinates are in the ECEF system at transmission time T and must be transformed to the ECEF system at reception time, as follows (and section 5.1.1.2, Volume I).

C. Account for Earth's rotation during the flight signal travel time (see equation (5.11), Volume I):

$$\mathbf{r}^{sat} = \mathbf{R}_3\,(\omega_E \Delta t) \cdot \widetilde{\mathbf{r}}^{sat}$$

where the flight signal travel time Δt can be computed as[69]

$$\Delta t = \frac{1}{c}||\widetilde{\mathbf{r}}^{sat} - \mathbf{r}_{0_{rcv}}||$$

with $\omega_E = 7.292\,115\,146\,7 \cdot 10^{-5}$ rad/s and

$$\mathbf{r}_{0_{rcv}} = [4\,789\,032.6277, 176\,595.0498, 4\,195\,013.2503]\,\text{m}$$

the approximate receiver (a priori) coordinates in the header of the RINEX file `UPC11490.05O`.

Solution:

- $\Delta t = 0.076\,337\,713$ s.
  $\mathbf{r}^{sat} = [6\,364\,789.025, -14\,298\,268.493, 21\,851\,197.941]$ m.

The values computed by `gLAB` can be found by:

```
grep MODEL gLAB.out | grep -v INFO |
  gawk '{if ($6==25)print $4,$6,$11,$12,$13}'|head -1
```

[68] This file is provided together with the exercise files.

[69] The pseudorange could also be used to compute Δt, but it includes the receiver clock error, which can reach up to 1 ms (or even more, depending on the receiver configuration).

(e) *Geometric range computation*

From previous results, compute the geometric range between the satellite[emission time] and receiver[reception time] coordinates.

Solution:

- $\rho_0 = ||\mathbf{r}^{sat} - \mathbf{r}_{0_{rcv}}|| = 22\,885\,487.5548\,\text{m}$.
- The values computed by `gLAB` can be found by:

```
grep MODEL gLAB.out | grep -v INFO |
    gawk '{if ($6==25) print $4,$6,$17}'|head -1
```

(f) *Relativistic clock correction*

Calculate the relativistic clock correction using (see section 5.2.1, Volume I)

$$\Delta_{rel} = -2\,\frac{\mathbf{r}\cdot\mathbf{v}}{c^2} \tag{5.1}$$

Solution:

- The velocity can be computed from the $\mathbf{r}^{sat}$ variation in two close epochs (for instance, $dt = 0.001\,\text{s}$) around transmission time.
  Execute for instance (in the ECEF system):

```
echo "2005 149 299.92366224" > time.dat
cat time.dat eph.dat | GPSxyz
```
$\Longrightarrow \widetilde{\mathbf{r}}^{sat}(T)$

```
echo "2005 149 299.92466224" > time.dat
cat time.dat eph.dat | GPSxyz
```
$\Longrightarrow \widetilde{\mathbf{r}}^{sat}(T+dt)$

  From previous results

  $\widetilde{\mathbf{v}} = [\widetilde{\mathbf{r}}^{sat}(T+dt) - \widetilde{\mathbf{r}}^{sat}(T)/dt] = [2443.422, 1116.567, 25.641]\,\text{m/s}$

- Equation (5.1) can be evaluated using[70] $\mathbf{r} = \widetilde{\mathbf{r}}^{sat}(T)$, with $\widetilde{\mathbf{r}}^{sat}(T) = [6\,364\,868.618, -14\,298\,233.062, 21\,851\,197.940]\,\text{m}$.

  Thus, $\Delta_{rel} = -3.280\,34 \cdot 10^{-9}\text{s} \Longrightarrow \boxed{c\,\Delta_{rel} = -0.983\,42\,\text{m}}$.

- The relativistic clock correction from `gLAB` can be found by:

```
grep MODEL gLAB.out | grep -v INFO |
    gawk '{if ($6==25) print $4,$6,$22}' | head -1
```

Comment: The relativistic correction can also be computed using the relation $\Delta_{rel} = -2\,\mathbf{r}\cdot\mathbf{v}/c^2 = -2(\sqrt{\mu\,a}/c^2)\,e\sin E$ *where* a, e *and* E *correspond to the osculating orbit computed from broadcast ephemerides. The semi-major axis and eccentricity are given in broadcast ephemerides as* $\sqrt{a} = 5.153\,704\,689\,026 \cdot 10^3\ m^{1/2}$, $e = 1.204\,112\,719\,279 \cdot 10^{-2}$. *Take* $\mu = 3\,986\,004.418 \cdot 10^8\ m^3/s^2$.

The eccentric anomaly is provided by program `GPSxyz`, by executing:

```
echo "2005 149 299.92366224" > time.dat
cat time.dat eph.dat | GPSxyz
```

The computation outputs $E = 3.022\,976$ rad.

[70] Note that $\mathbf{r}$ and $\mathbf{v}$ in equation (5.1) are the position and velocity in an inertial system, but the scalar product $\mathbf{r}\cdot\mathbf{v}$ can be evaluated either in a Conventional Celestial Reference System (CRS) or Conventional Terrestrial Reference System (TRS), i.e. ECEF, system. Indeed, Earth's rotation $\boldsymbol{\omega}\times\mathbf{r}$ should be added to the ECEF computed velocity to obtain $\mathbf{v}$ in an inertial system, but it cancels in the scalar product with $\mathbf{r}$.

Using these values, it is found that $c\,\Delta_{rel} = -0.978\,12$ m.
The small discrepancy of a few millimetres with the previous result is due to the use of the osculating orbit to compute Δ_{rel} (i.e. the value of $\mathbf{v}$ for the osculating orbit is slightly different from the previously computed one $\mathbf{v} = \Delta\mathbf{r}/dt$).

(g) *Tropospheric delay*

Compute the tropospheric correction, applying the algorithm defined in section 5.4.2.1, Volume I.

Hint: It can be computed with the help of program `tropo.f` as follows:

i. Generate the file `tropo.dat`[71] as indicated in the header of the program `tropo.f`.
ii. Execute:

```
cat tropo.dat | tropo
```

Solution:

$T = 4.465\,83$ m.

The values computed by `gLAB` can be found by:

```
grep MODEL gLAB.out | grep -v INFO |
    gawk '{if ($6==25) print $4,$6,$24}' |head -1
```

(h) *Ionospheric delay*

Compute the ionospheric correction, applying the Klobuchar model defined in section 5.4.1.2.1, Volume I:

Hint: It can be computed with the help of program `iono.f` as follows:

i. Generate file `iono.dat` as indicated in the header of program `iono.f`.
ii. Execute:

```
cat iono.dat | iono
```

Solution:

$I_1 = 2.472\,64$ m of `L1` delay.

The values computed by `gLAB` can be found by:

```
grep MODEL gLAB.out | grep -v INFO |
    gawk '{if ($6==25) print $4,$6,$25}' |head -1
```

(i) *Calculate the value of modelled pseudorange (see Volume I, equation (6.1))*

$\texttt{C1}_{mod} = \rho - c\,\delta t^{sat} + T + I_1 + \mathrm{TGD}$

Solution:

From previous computations
$\rho = 22\,885\,487.554\,79$ m, TGD $= -2.233\,63$ m,
$c\,\delta t^{sat} = c\,\widetilde{\delta t}^{\,sat} + c\,\Delta_{rel} = 28\,181.895\,51 - 0.98\,342 = 28\,180.912\,09$ m,
$T = 4.465\,83$ m, $I_1 = 2.472\,64\,\mathrm{m}_{L_1}$.
Thus, $\texttt{C1}_{mod} = 22\,857\,311.347\,54$ m.

The values computed by `gLAB` can be found by:

```
grep MODEL gLAB.out | grep -v INFO |
    gawk '{if ($6==25) print $4,$6,$10}' |head -1
```

[71] This file is provided together with the exercise files.

(j) *Calculate the prefit residual (see Volume I, equation (6.8))*

Solution:

From the RINEX measurement file `UPC11490.05O`, the measured code pseudorange for PRN25 at time $t = 300$ s is `C1` $= 22\,857\,303.996$ m. Therefore the prefit residual is

Prefit $=$ `C1` $-$ `C1`$_{mod} = -7.351\,54$ m.

The values computed by `gLAB` can be found by:

```
grep PREFIT gLAB.out | grep -v INFO |
    gawk '{if ($6==25) print $4,$6,$8}' |head -1
```

6. Navigation equations system and least squares solution

Using the data sets of the previous exercise, build the navigation equations system to compute the East North Up (ENU) solution (in SPP mode) for the satellites in view at the instant $t = 300$ s (see section 6.1 and equation (B.14) in Volume I).

Then compute the Least Squares (LS) solution. Also, compute HDOP, VDOP and TDOP.

Complete the following steps:

(a) The basic linearised measurement equation is $\mathbf{y} = \mathbf{G} \cdot \mathbf{x}$, where $\mathbf{y}$ is a vector containing the prefit residuals and $\mathbf{G}$ is the geometry matrix, whose rows are defined for each satellite in view as

$$\mathbf{G}_i = [-\cos El^i \sin Az^i, -\cos El^i \cos Az^i, -\sin El^i, 1]$$

The satellite elevation El^i and azimuth Az^i angles can be found from the `gLAB.out` output file.

Hint: Matrix $\mathbf{G}$ and prefit residual vector $\mathbf{y}$ can be generated directly from the `gLAB.out` output file of the previous exercise as follows:[72]

```
grep PREFIT gLAB.out | grep -v INFO | grep -v "PREFIT\*" | gawk
     'BEGIN{g2r=atan2(1,1)/45}{if ($4==300) {e=$15*g2r;a=$16*g2r;
      print $8,-cos(e)*sin(a),-cos(e)*cos(a),-sin(e),1}}' > M.dat
```

Vector $\mathbf{y}$ corresponds to the first column of file `M.dat` and matrix $\mathbf{G}$ to the last four columns.

(b) Compute the LS solution of the navigation system. Using Octave or MATLAB, upload the contents of file `M.dat` and execute the following instructions as well:

```
y=M(:,1)
G=M(:,2:5)
x=inv(G'*G)*G'*y
```

[72] Satellite PRN21 is excluded due to the 5° elevation mask used by default in `gLAB` (see `Elevation mask` in the `[Preprocess]` section of the `gLAB` GUI). This exclusion is indicated by the symbol `*` in the `PREFIT` label (i.e. `PREFIT*`). In the following sentence the rows labelled as `PREFIT*` are excluded with the '`grep -v` ' instruction (i.e. `grep -v "PREFIT\*"`).

The values computed by gLAB can be found by:

```
(E,N,U) coordinates:
grep OUTPUT gLAB.out | grep -v INFO |
              gawk '{if ($4==300) print $19,$18,$20}'
Receiver clock:
grep FILTER gLAB.out | grep -v INFO |
                        gawk '{if ($4==300) print $8}'
```

(c) HDOP, VDOP and TDOP can be computed as follows:

```
Compute
Q=inv(G'*G)
Then:
HDOP=sqrt(Q(1,1)+Q(2,2))
VDOP=sqrt(Q(3,3))
TDOP=sqrt(Q(4,4))
```

The values computed by gLAB can be found by:

```
grep OUTPUT gLAB.out | grep -v INFO |
              gawk '{if ($4==300) print $27,$28,$26}'
```

7. Navigation equations system and LS solution (XYZ)

Repeat the previous exercise, but writing the system and computing the solution in (XYZ) coordinates. Also, compute GDOP, Precision Dilution Of Precision (PDOP) and TDOP.

Complete the following steps:

(a) The matrix $\mathbf{G}$ is now

$$\mathbf{G}_i = \left[\frac{x_0 - x^i}{\rho_0^i}, \frac{y_0 - y^i}{\rho_0^i}, \frac{z_0 - z^i}{\rho_0^i}, 1 \right]$$

where $\mathbf{r}_0 = (x_0, y_0, y_0)$ is the 'a priori' receiver coordinates at reception time, $\mathbf{r}^i = (x^i, y^i, y^i)$ are the satellite coordinates at transmission time, and $\rho_0^i = ||\mathbf{r}^i - \mathbf{r}_0||$.

Hint: Matrix $\mathbf{G}$ and prefit residual vector $\mathbf{y}$ can be generated directly from the gLAB.out output file as follows:[73]

```
grep MODEL gLAB.out | grep C1 | gawk 'BEGIN{x=4789032.6277;
     y=176595.0498;z=4195013.2503} {if ($4==300 && $6!=21)
     {r1=x-$11;r2=y-$12;r3=z-$13;r=sqrt(r1*r1+r2*r2+r3*r3);
     print $9-$10,r1/r,r2/r,r3/r,1}}' > M.dat
```

Vector $\mathbf{y}$ corresponds to the first column of file M.dat and matrix $\mathbf{G}$ to the last four columns.

[73] Satellite PRN21 is excluded as in the previous case, because its elevation at time 300 s is below the elevation mask of 5° used by default in the gLAB computations (see Elevation mask in the [Preprocess] section of the gLAB GUI).

The matrix **G** and vector **y** values computed by gLAB can be found by:

```
grep PREFIT gLAB.out | grep -v INFO |
 gawk '{if ($4==300 && $6!=21) print $8,$11,$12,$13,$14}'
```

(b) Compute the LS solution of the navigation system. Using Octave or MATLAB, upload the contents of file M.dat and execute the following instructions, as well:

```
y=M(:,1)
G=M(:,2:5)
x=inv(G'*G)*G'*y
```

The values computed by gLAB can be found by:

```
(X,Y,Z) coordinates:
grep OUTPUT gLAB.out | grep -v INFO |
              gawk '{if ($4==300) print $9,$10,$11}'
Receiver clock:
grep FILTER gLAB.out | grep -v INFO |
                       gawk '{if ($4==300) print $8}'
```

(c) GDOP, PDOP and TDOP can be computed as follows:

```
Compute
Q=inv(G'*G)
Then:
GDOP=sqrt(Q(1,1)+Q(2,2)+Q(3,3)+Q(4,4))
PDOP=sqrt(Q(1,1)+Q(2,2)+Q(3,3))
TDOP=sqrt(Q(4,4))
```

The values computed by gLAB can be found by:

```
grep OUTPUT gLAB.out | grep -v INFO |
             gawk '{if ($4==300) print $24,$25,$26}'
```

8. Solving with the Kalman filter

The measurement file UPC11490.05O has been collected by a receiver with fixed coordinates. Using navigation file UPC11490.05N, compute the SPP solution in static mode[74] and check *by hand* the computation of the solution for the first three epochs (i.e. $t = 300$, $t = 600$ and $t = 900$ seconds).

Complete the following steps:

(a) Set the default configuration of gLAB for the SPP mode. Then, in section [Filter], select [⊙ Static] in the Receiver Kinematics option. To process the data click Run gLAB.

[74] This solution involves the application of the Kalman filter assuming coordinates as constant and the clock as white noise, according to the model defined in section 6.1.2.1.1, Volume I.

(b) Write the Kalman filter equations according to Fig. 6.2, in section 6.1.2 of Volume I.

(c) Using the previous equations and the configuration parametres applied by gLAB compute by hand the solution for the first three epochs[75] (i.e. $t = 300$, $t = 600$ and $t = 900$ s).

Note: Use the prefit residual vector $\mathbf{y}(k)$ *and design matrix* $\mathbf{G}(k)$ *computed by* gLAB.

Hint:

i. Filter configuration (according to gLAB):

- Initialisation:
  $\widehat{\mathtt{x0}} \equiv \widehat{x}(0) = (0, 0, 0, 0)$,
  $\mathtt{P0} \equiv \mathbf{P}(0) = \sigma_0^2\,\mathbf{I}$, with $\sigma_0 = 3 \cdot 10^5$m.
- Process noise $\mathbf{Q}$ and transition matrices $\mathbf{\Phi}$:

$$\mathtt{Q} \equiv \boldsymbol{Q}(k) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{dt}^2 \end{bmatrix}, \quad \Phi \equiv \mathbf{\Phi}(k) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

  with $\sigma_{dt} = 3 \cdot 10^5$m.
- Measurement covariance matrix:
  $\mathtt{Rk} \equiv \mathbf{R}(k) = \sigma_y^2\,\mathbf{I}$, with $\sigma_y = 1\,\mathrm{m}$.

ii. Kalman filter iterations:

k=1:

Predict:

$$\mathtt{x1}^- = \mathtt{\Phi} \cdot \widehat{\mathtt{x}}\mathtt{0}$$
$$\mathtt{P1}^- = \mathtt{\Phi} \cdot \mathtt{P0} \cdot \mathtt{\Phi}^{\mathtt{T}} + \mathtt{Q}$$

Update:

$$\mathtt{P1} = [\mathtt{G1}^{\mathtt{T}} \cdot \mathtt{R1}^{-1} \cdot \mathtt{G1} + (\mathtt{P1}^-)^{-1}]^{-1}$$
$$\widehat{\mathtt{x}}\mathtt{1} = \mathtt{P1} \cdot [\mathtt{G1}^{\mathtt{T}} \cdot \mathtt{R1}^{-1} \cdot \mathtt{y1} + (\mathtt{P1}^-)^{-1} \cdot \mathtt{x1}^-]$$

k=2:

Predict:

$$\mathtt{x2}^- = \mathtt{\Phi} \cdot \widehat{\mathtt{x}}\mathtt{1}$$
$$\mathtt{P2}^- = \mathtt{\Phi} \cdot \mathtt{P1} \cdot \mathtt{\Phi}^{\mathtt{T}} + \mathtt{Q}$$

Update:

$$\mathtt{P2} = [\mathtt{G2}^{\mathtt{T}} \cdot \mathtt{R2}^{-1} \cdot \mathtt{G2} + (\mathtt{P2}^-)^{-1}]^{-1}$$
$$\widehat{\mathtt{x}}\mathtt{2} = \mathtt{P2} \cdot [\mathtt{G2}^{\mathtt{T}} \cdot \mathtt{R2}^{-1} \cdot \mathtt{y2} + (\mathtt{P2}^-)^{-1} \cdot \mathtt{x2}^-]$$

k=3:

...

iii. Data vectors and matrices: Vectors $\mathtt{yk} \equiv \mathbf{y}(k)$ and design matrices $\mathtt{Gk} \equiv \mathbf{G}(k)$ are generated from the gLAB.out file.

[75] The default configuration of gLAB uses 300 s of data decimation.

Execute for instance:[76]

```
grep "PREFIT" gLAB.out | grep -v INFO |
          gawk '{if ($6!=21 )print $0}'|
  gawk '{if ($4==300) print $8,$11,$12,$13,$14}'
                                        > M300.dat

grep "PREFIT" gLAB.out | grep -v INFO |
  gawk '{if ($4==600) print $8,$11,$12,$13,$14}'
                                        > M600.dat

grep "PREFIT" gLAB.out | grep -v INFO |
  gawk '{if ($4==900) print $8,$11,$12,$13,$14}'
                                        > M900.dat
```

Then using Octave or MATLAB:

```
y1=M300(:,1)
G1=M300(:,2:5)

y2=M600(:,1)
G2=M600(:,2:5)

y3=M900(:,1)
G3=M900(:,2:5)
```

iv. Results computed by gLAB:

A. (X,Y,Z) coordinates:

```
grep OUTPUT gLAB.out | grep -v INFO |
        gawk '{if ($4==300) print $9,$10,$11}'

grep OUTPUT gLAB.out | grep -v INFO |
        gawk '{if ($4==600) print $9,$10,$11}'

grep OUTPUT gLAB.out | grep -v INFO |
        gawk '{if ($4==900) print $9,$10,$11}'
```

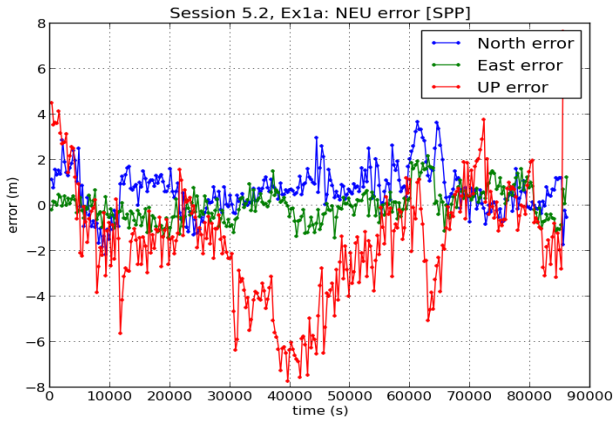
B. Receiver clock

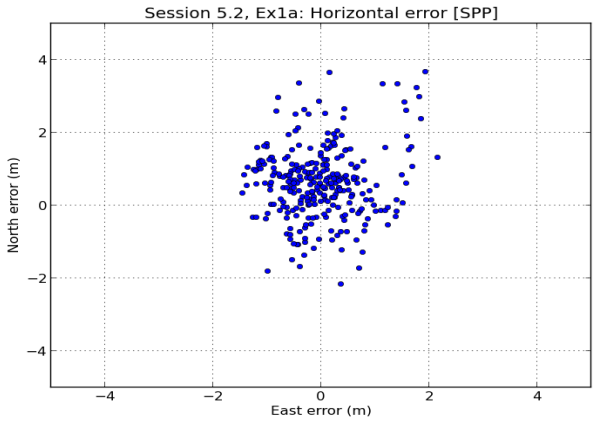
```
grep FILTER gLAB.out | grep -v INFO |
                gawk '{if ($4==300) print $8}'

grep FILTER gLAB.out | grep -v INFO |
                gawk '{if ($4==600) print $8}'

grep FILTER gLAB.out | grep -v INFO |
                gawk '{if ($4==900) print $8}'
```

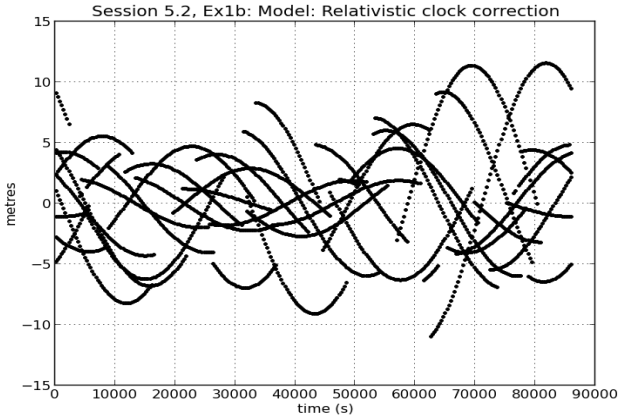
Comment:
A wide user-friendly collection of MATLAB scripts (M-files) for processing GPS observations are provided by professor K. Borre at the following site: http://kom.aau.dk/∼borre/. The textbooks [Strang and Borre, 1997] and [Borre et al., 2006] are also recommended.

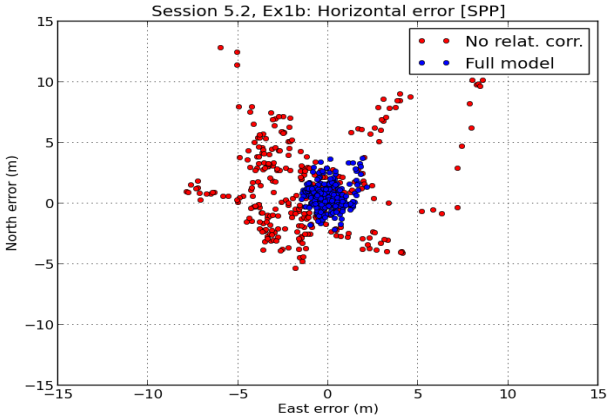
[76] Satellite PRN21 is excluded at $t = 300\,\mathrm{s}$, as in previous exercises, due to the elevation mask.

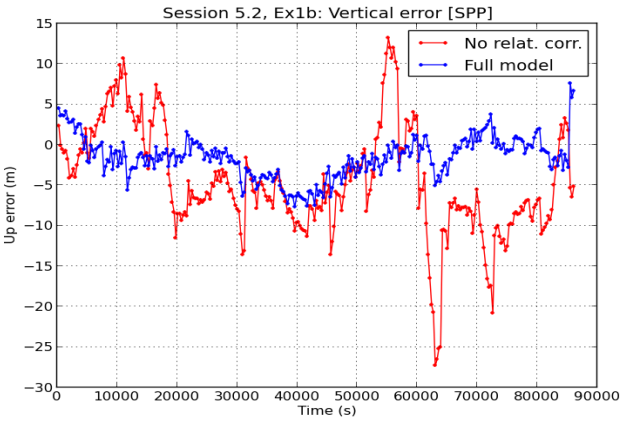
Graphs Session 5.2

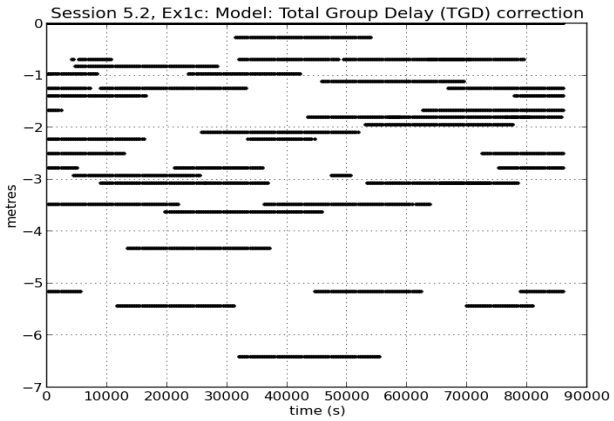


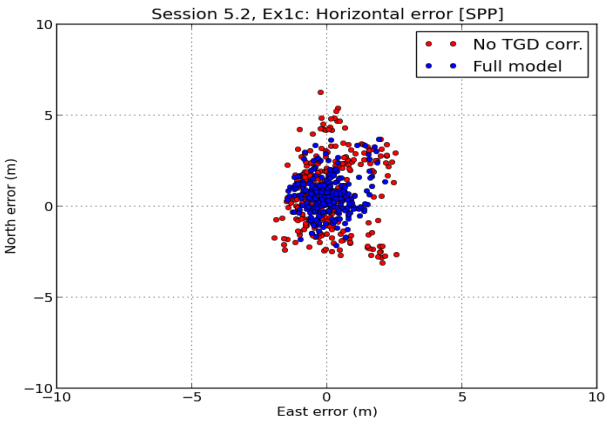


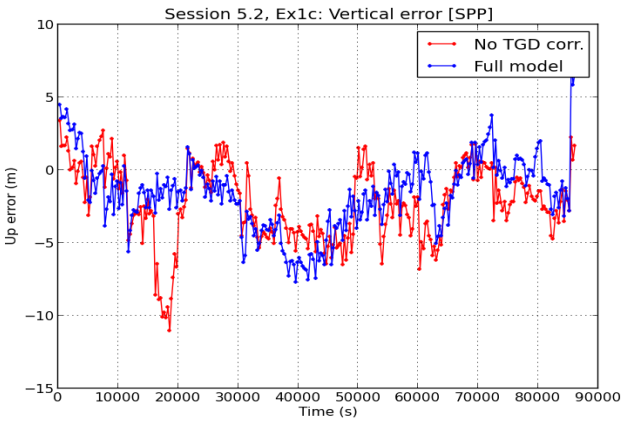


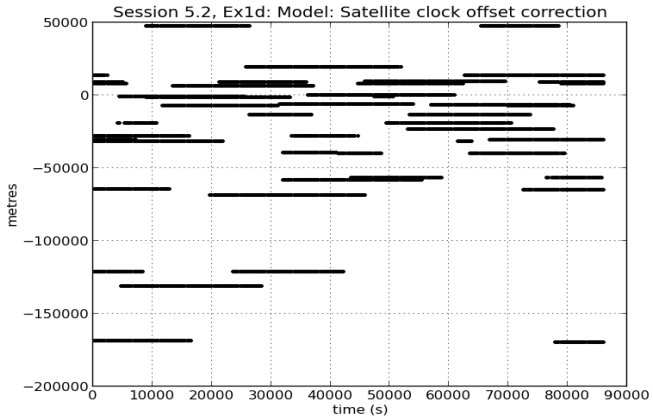


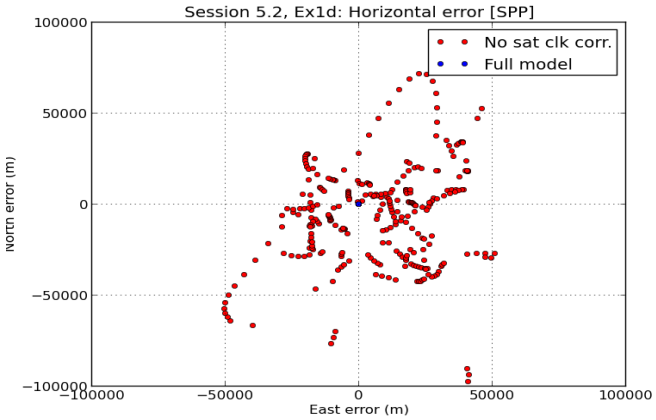


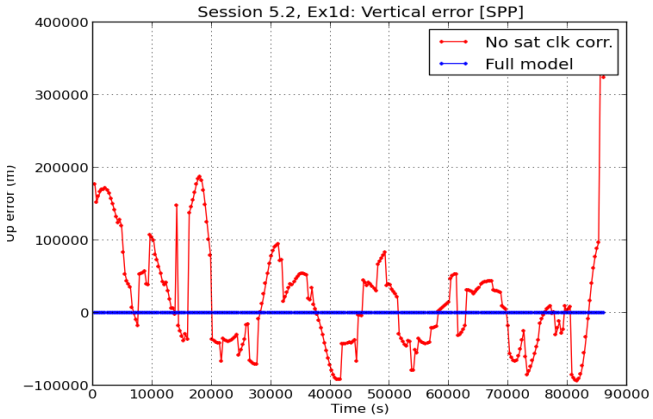


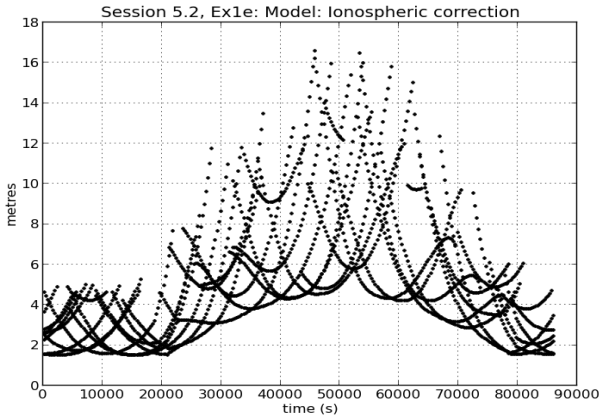


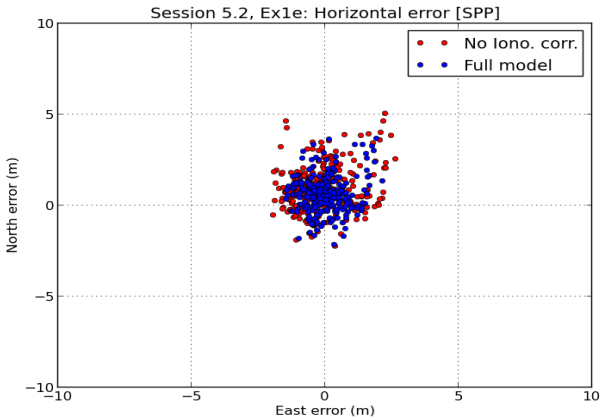


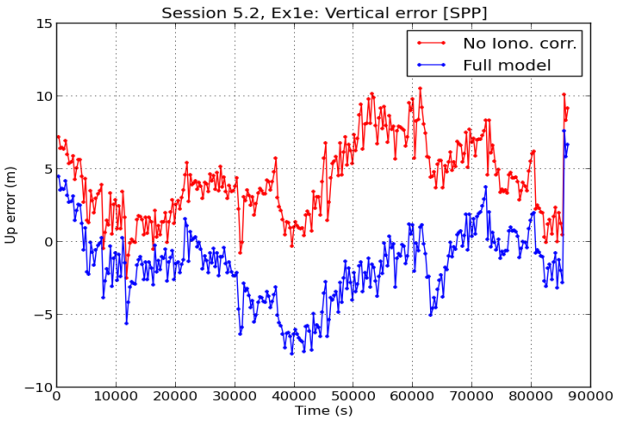

Session 5.2, Ex1d: Model: Satellite clock offset correction
metres
time (s)

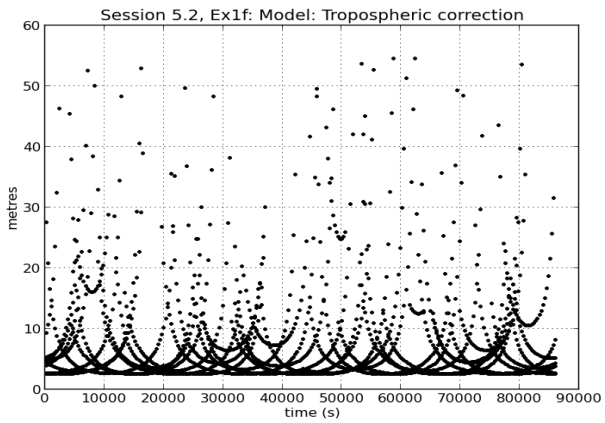

Session 5.2, Ex1d: Horizontal error [SPP]
No sat clk corr.
Full model
North error (m)
East error (m)

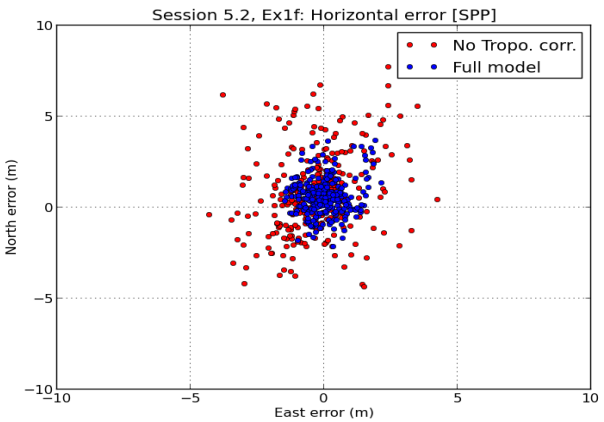

Session 5.2, Ex1d: Vertical error [SPP]
No sat clk corr.
Full model
Up error (m)
Time (s)

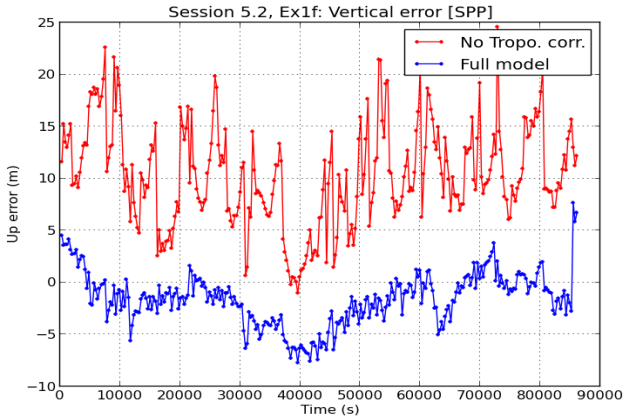

Session 5.2, Ex1e: Model: Ionospheric correction
metres
time (s)

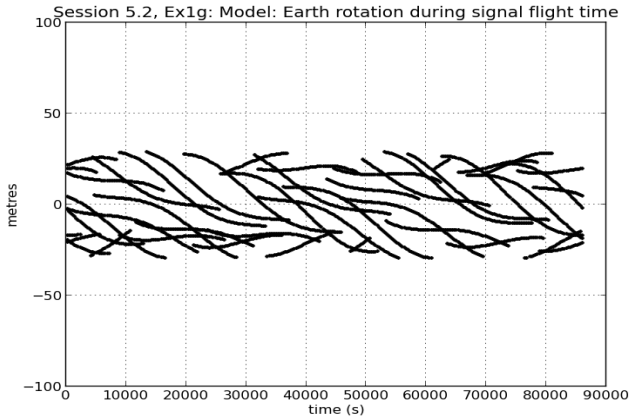

Session 5.2, Ex1e: Horizontal error [SPP]
No Iono. corr.
Full model
North error (m)
East error (m)

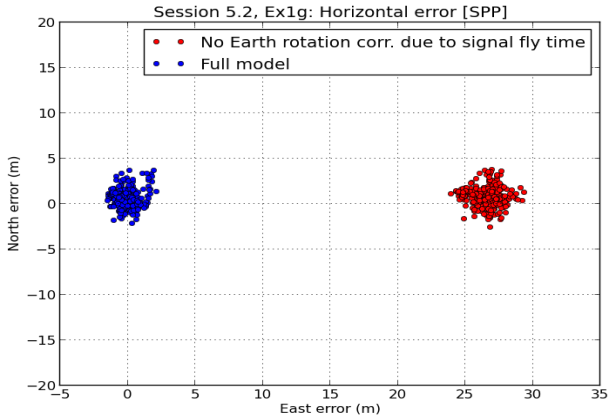

Session 5.2, Ex1e: Vertical error [SPP]
No Iono. corr.
Full model
Up error (m)
Time (s)

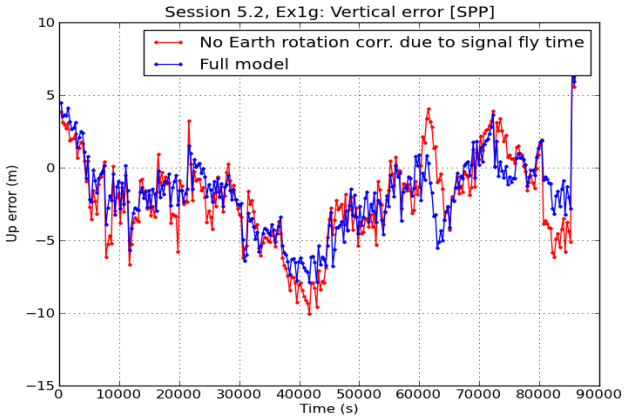

Session 5.2, Ex1f: Model: Tropospheric correction
metres
time (s)

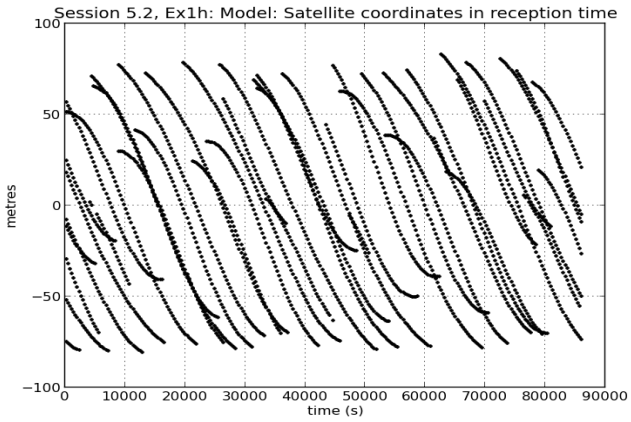

Session 5.2, Ex1f: Horizontal error [SPP]
No Tropo. corr.
Full model
North error (m)
East error (m)

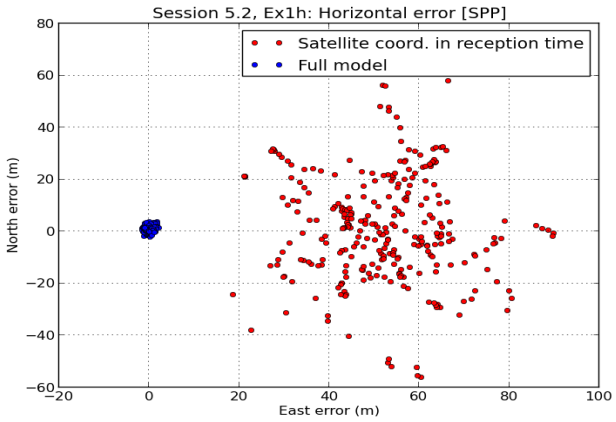

Session 5.2, Ex1f: Vertical error [SPP]
No Tropo. corr.
Full model
Up error (m)
Time (s)

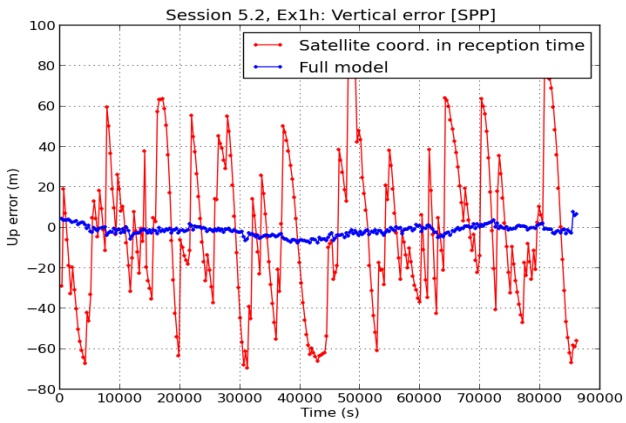

Session 5.2, Ex1g: Model: Earth rotation during signal flight time
metres
time (s)

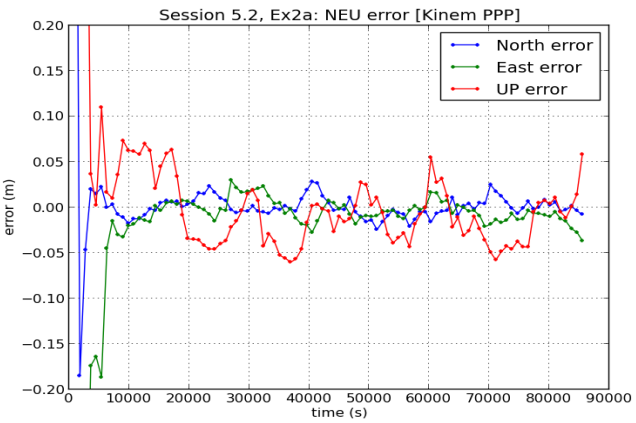

Session 5.2, Ex1g: Horizontal error [SPP]
No Earth rotation corr. due to signal fly time
Full model
North error (m)
East error (m)

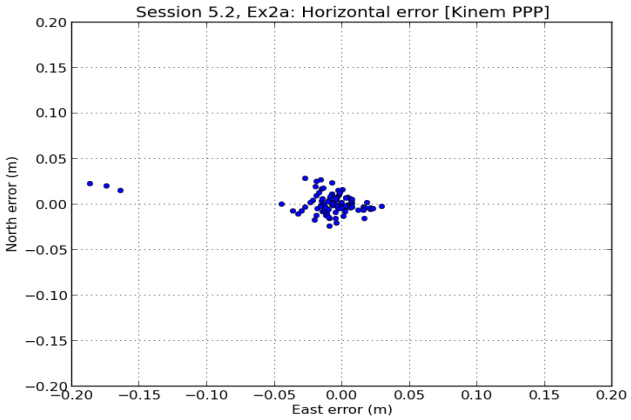

Session 5.2, Ex1g: Vertical error [SPP]
No Earth rotation corr. due to signal fly time
Full model
Up error (m)
Time (s)

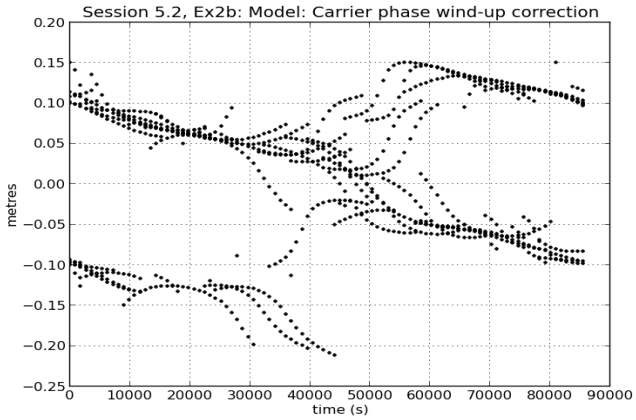

Session 5.2, Ex1h: Model: Satellite coordinates in reception time
metres
time (s)

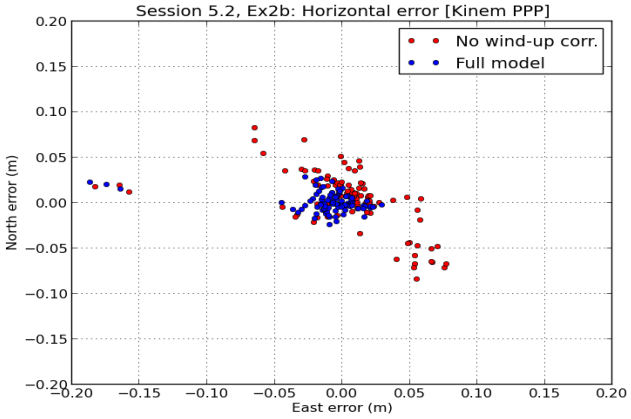

Session 5.2, Ex1h: Horizontal error [SPP]
Satellite coord. in reception time
Full model
North error (m)
East error (m)

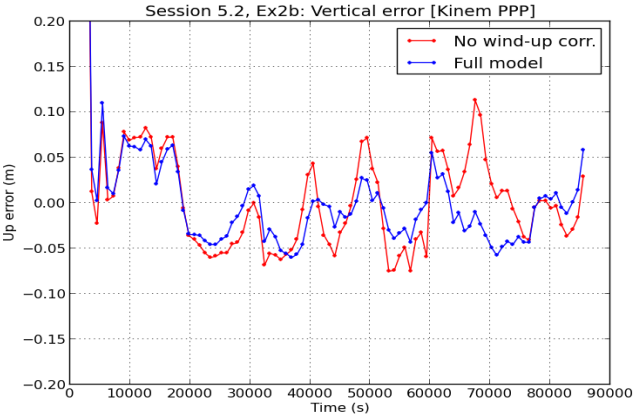

Session 5.2, Ex1h: Vertical error [SPP]
Satellite coord. in reception time
Full model
Up error (m)
Time (s)

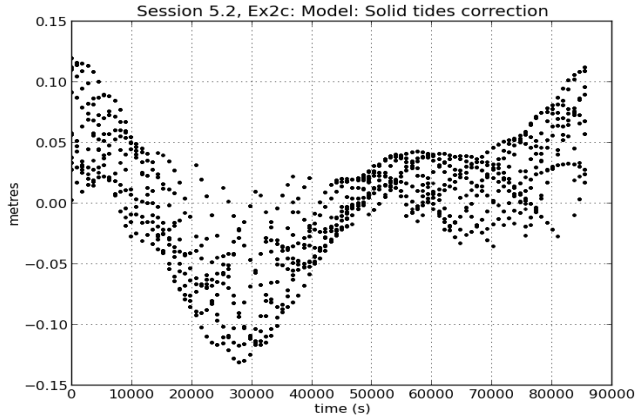

Session 5.2, Ex2a: NEU error [Kinem PPP]
North error
East error
UP error
error (m)
time (s)

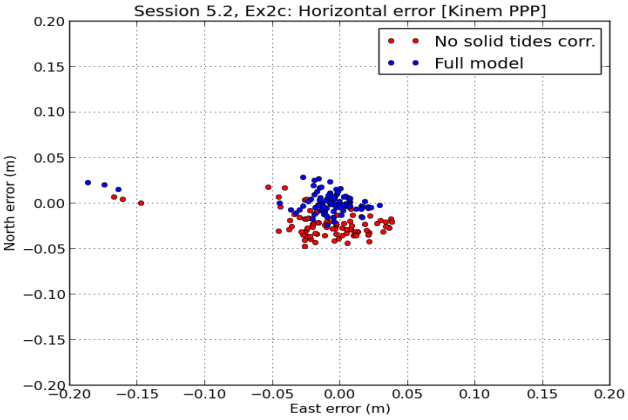

Session 5.2, Ex2a: Horizontal error [Kinem PPP]
North error (m)
East error (m)

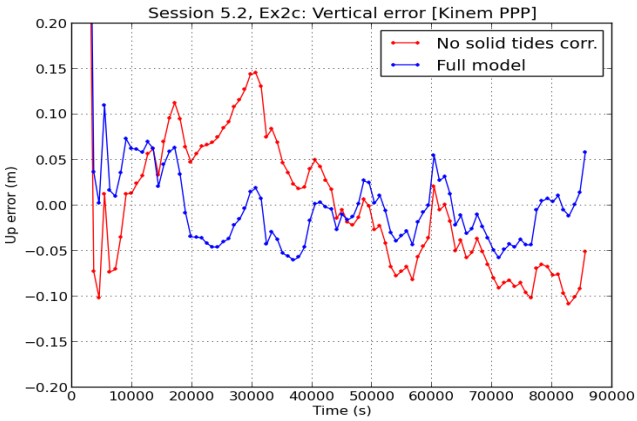

Session 5.2, Ex2b: Model: Carrier phase wind-up correction
metres
time (s)

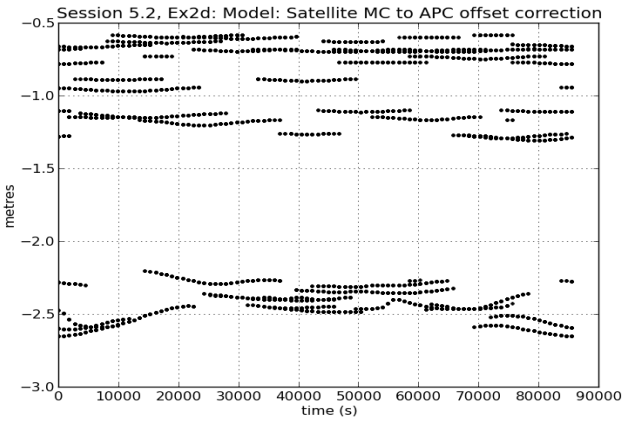

Session 5.2, Ex2b: Horizontal error [Kinem PPP]
No wind-up corr.
Full model
North error (m)
East error (m)

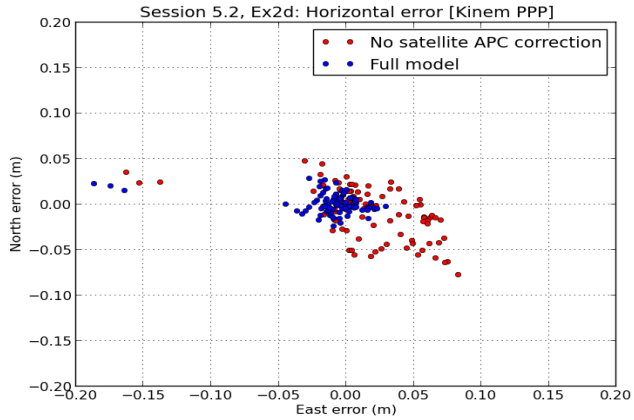

Session 5.2, Ex2b: Vertical error [Kinem PPP]
No wind-up corr.
Full model
Up error (m)
Time (s)

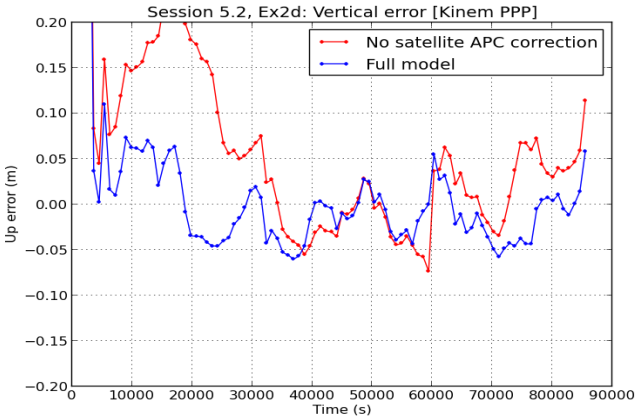

Session 5.2, Ex2c: Model: Solid tides correction
metres
time (s)

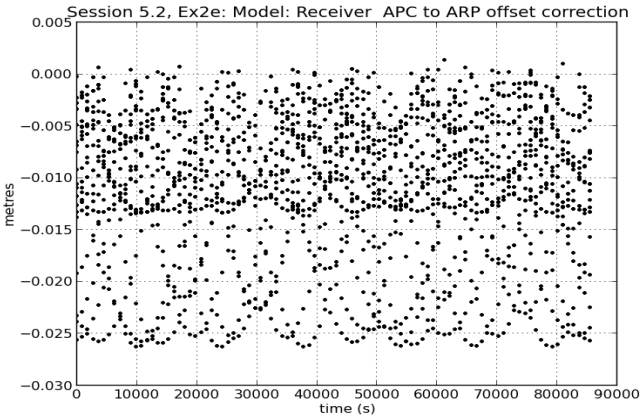

Session 5.2, Ex2c: Horizontal error [Kinem PPP]
No solid tides corr.
Full model
North error (m)
East error (m)

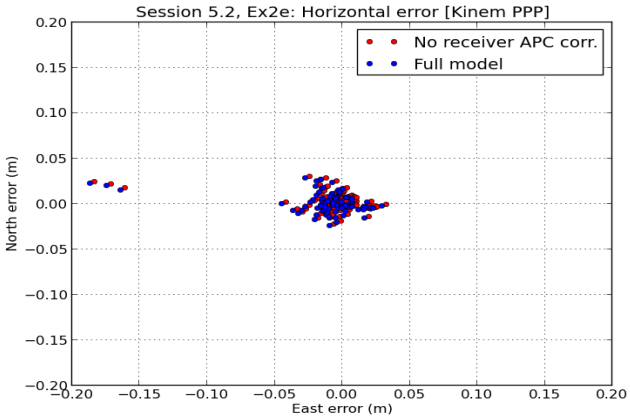

Session 5.2, Ex2c: Vertical error [Kinem PPP]
No solid tides corr.
Full model
Up error (m)
Time (s)

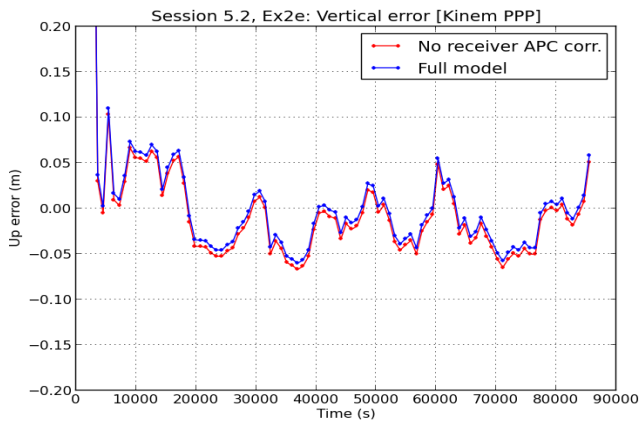

Session 5.2, Ex2d: Model: Satellite MC to APC offset correction
metres
time (s)

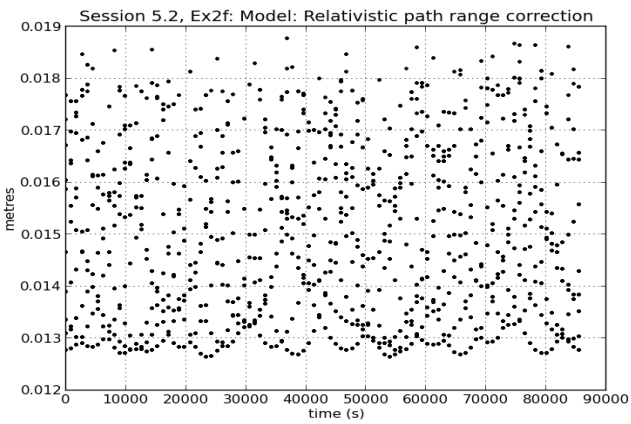

Session 5.2, Ex2d: Horizontal error [Kinem PPP]
No satellite APC correction
Full model
North error (m)
East error (m)

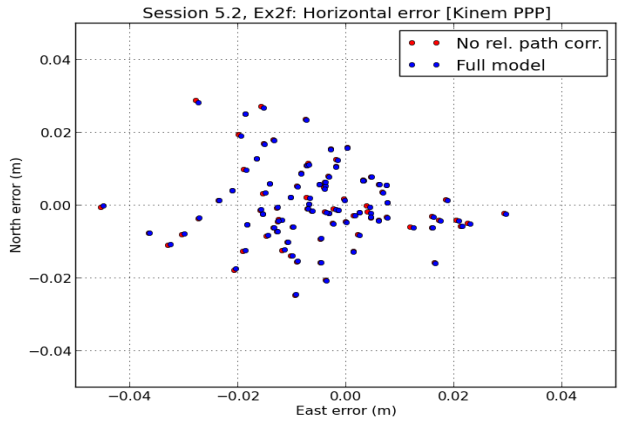

Session 5.2, Ex2d: Vertical error [Kinem PPP]
No satellite APC correction
Full model
Up error (m)
Time (s)

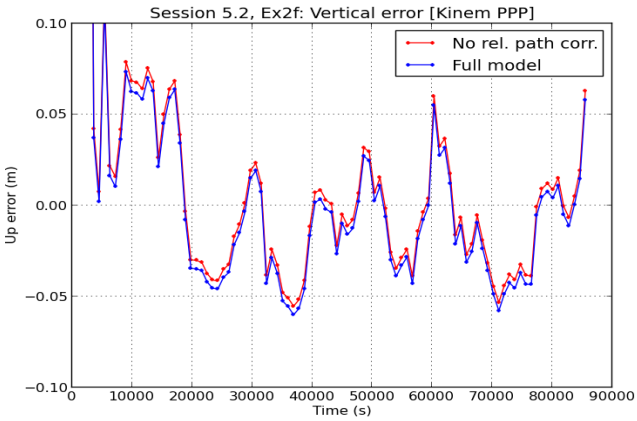

Session 5.2, Ex2e: Model: Receiver APC to ARP offset correction
metres
time (s)

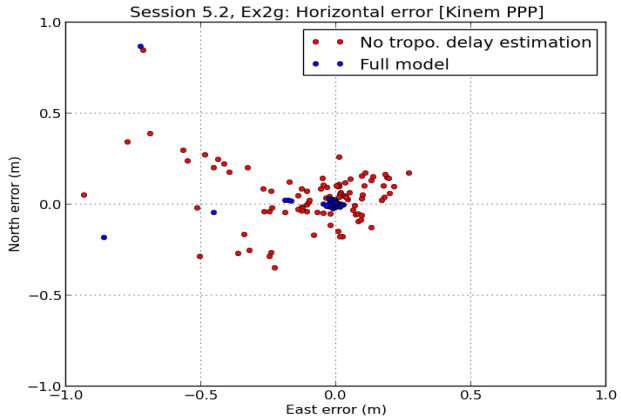

Session 5.2, Ex2e: Horizontal error [Kinem PPP]
No receiver APC corr.
Full model
North error (m)
East error (m)

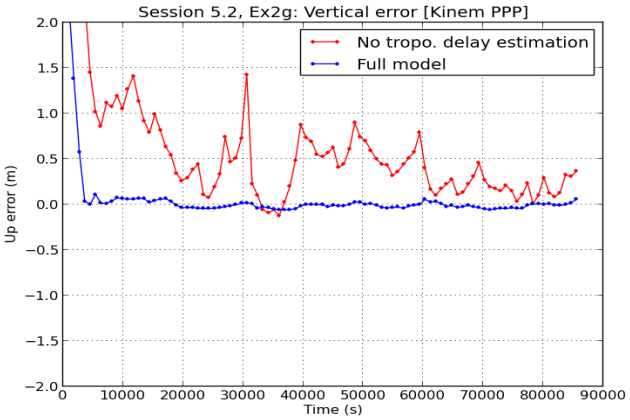

Session 5.2, Ex2e: Vertical error [Kinem PPP]
No receiver APC corr.
Full model
Up error (m)
Time (s)

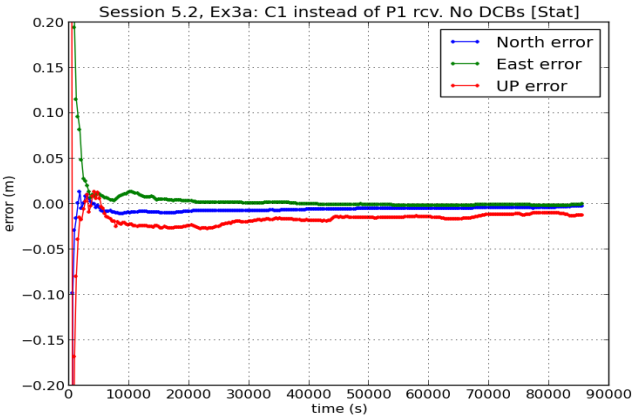

Session 5.2, Ex2f: Model: Relativistic path range correction
metres
time (s)

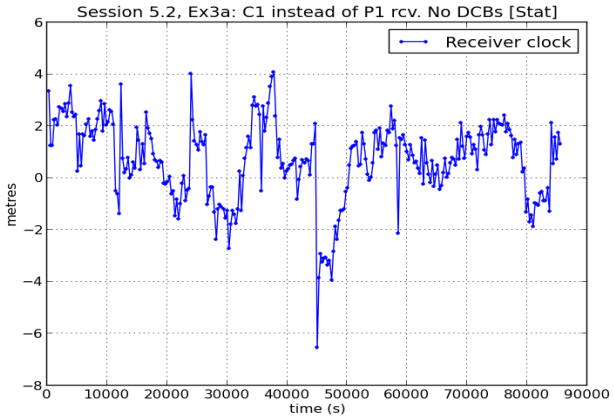

Session 5.2, Ex2f: Horizontal error [Kinem PPP]
No rel. path corr.
Full model
North error (m)
East error (m)

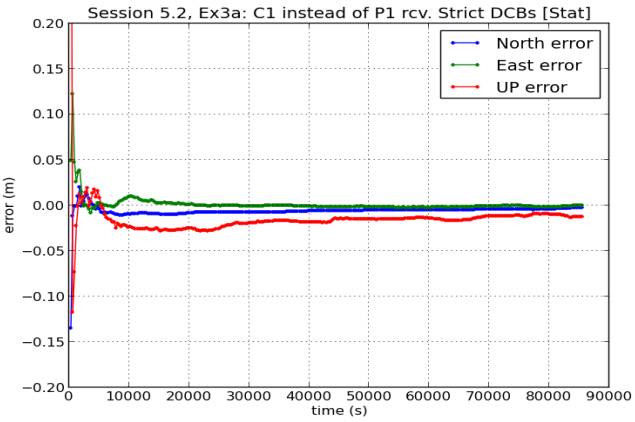

Session 5.2, Ex2f: Vertical error [Kinem PPP]
No rel. path corr.
Full model
Up error (m)
Time (s)

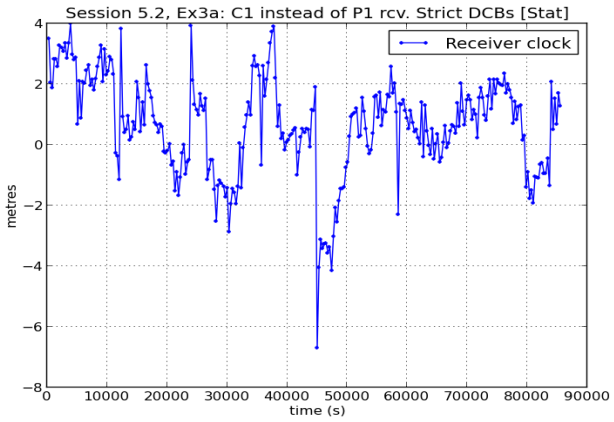

Session 5.2, Ex2g: Horizontal error [Kinem PPP]
No tropo. delay estimation
Full model
North error (m)
East error (m)

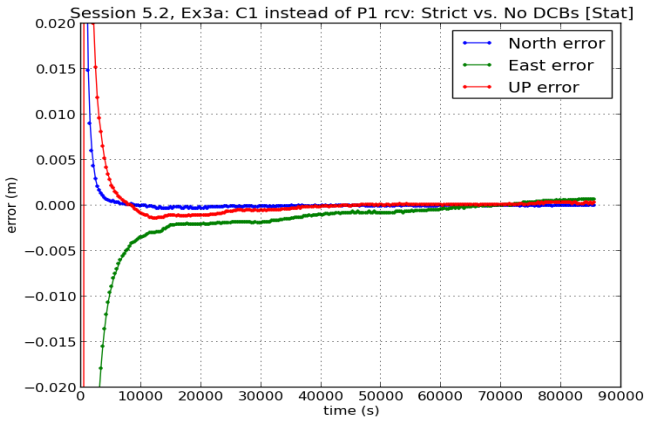

Session 5.2, Ex2g: Vertical error [Kinem PPP]
No tropo. delay estimation
Full model
Up error (m)
Time (s)

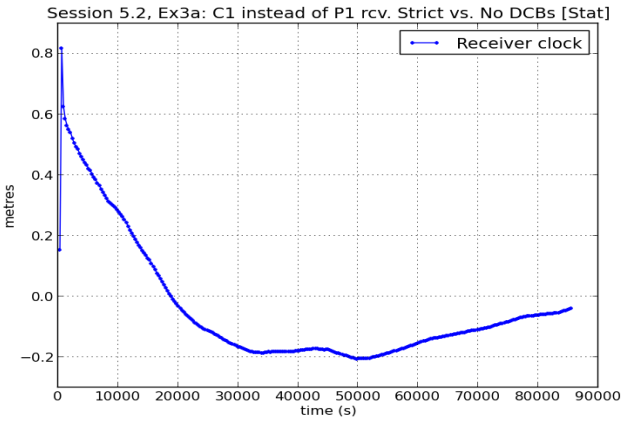

Session 5.2, Ex3a: C1 instead of P1 rcv. No DCBs [Stat]
North error
East error
UP error
error (m)
time (s)

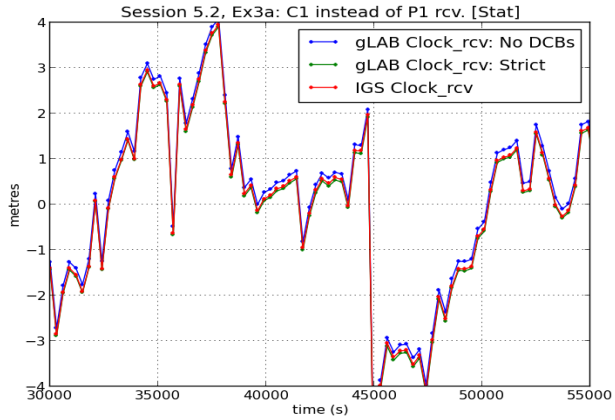

Session 5.2, Ex3a: C1 instead of P1 rcv. No DCBs [Stat]
Receiver clock
metres
time (s)

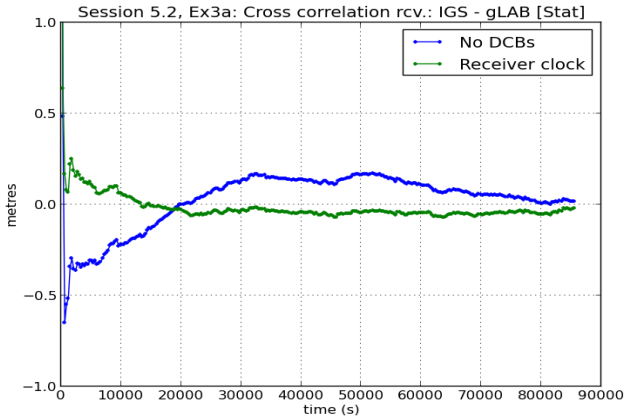

Session 5.2, Ex3a: C1 instead of P1 rcv. Strict DCBs [Stat]
North error
East error
UP error
error (m)
time (s)

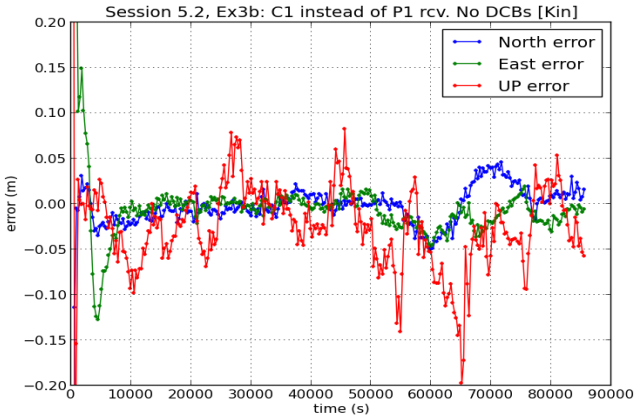

Session 5.2, Ex3a: C1 instead of P1 rcv. Strict DCBs [Stat]
Receiver clock
metres
time (s)

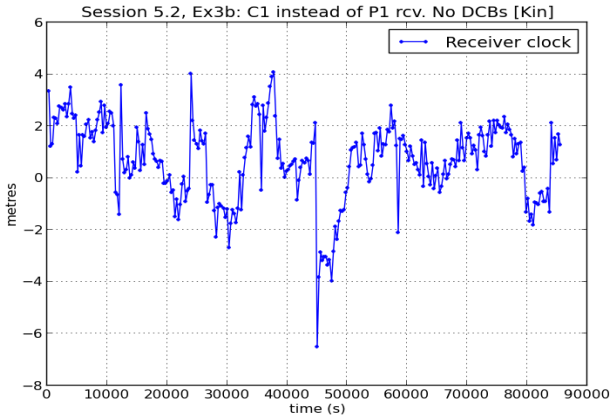

Session 5.2, Ex3a: C1 instead of P1 rcv: Strict vs. No DCBs [Stat]
North error
East error
UP error
error (m)
time (s)

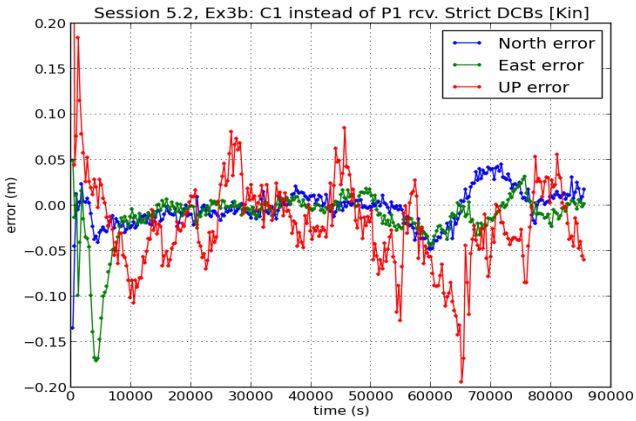

Session 5.2, Ex3a: C1 instead of P1 rcv. Strict vs. No DCBs [Stat]
Receiver clock
metres
time (s)

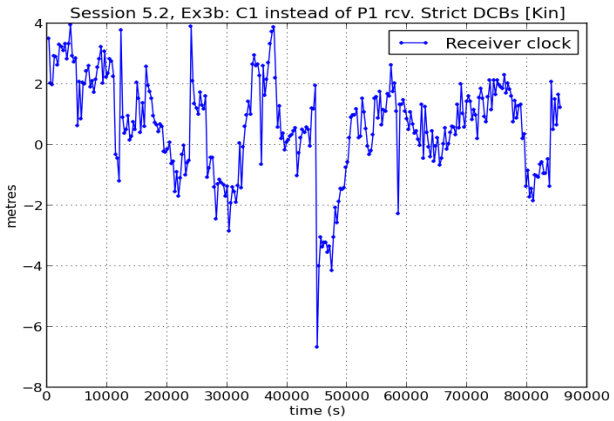

Session 5.2, Ex3a: C1 instead of P1 rcv. [Stat]
gLAB Clock_rcv: No DCBs
gLAB Clock_rcv: Strict
IGS Clock_rcv
metres
time (s)

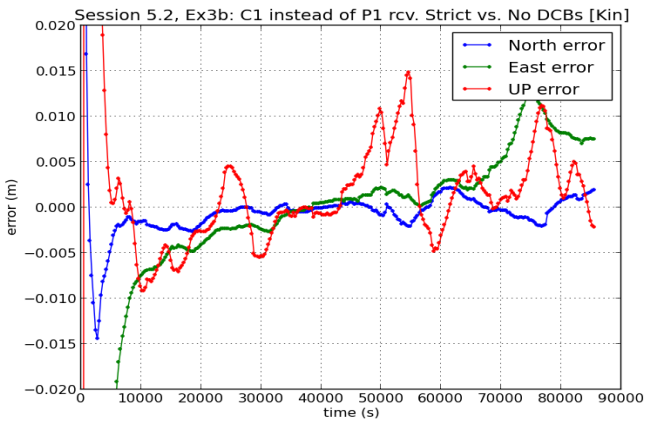

Session 5.2, Ex3a: Cross correlation rcv.: IGS - gLAB [Stat]
No DCBs
Receiver clock
metres
time (s)

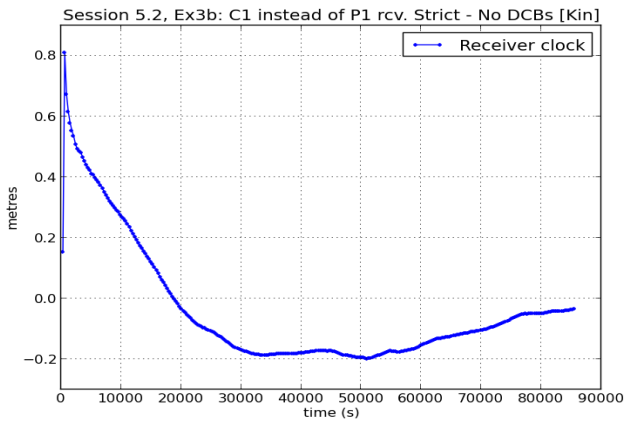

Session 5.2, Ex3b: C1 instead of P1 rcv. No DCBs [Kin]
North error
East error
UP error
error (m)
time (s)

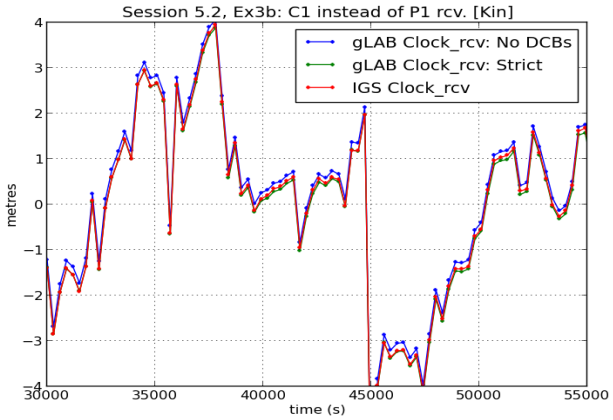

Session 5.2, Ex3b: C1 instead of P1 rcv. No DCBs [Kin]
Receiver clock
metres
time (s)

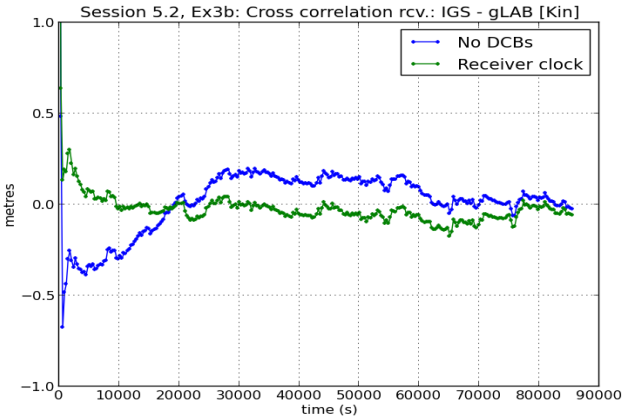

Session 5.2, Ex3b: C1 instead of P1 rcv. Strict DCBs [Kin]
North error
East error
UP error
error (m)
time (s)

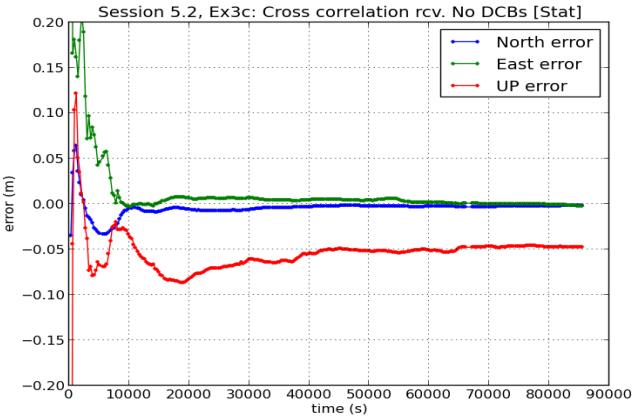

Session 5.2, Ex3b: C1 instead of P1 rcv. Strict DCBs [Kin]
Receiver clock
metres
time (s)

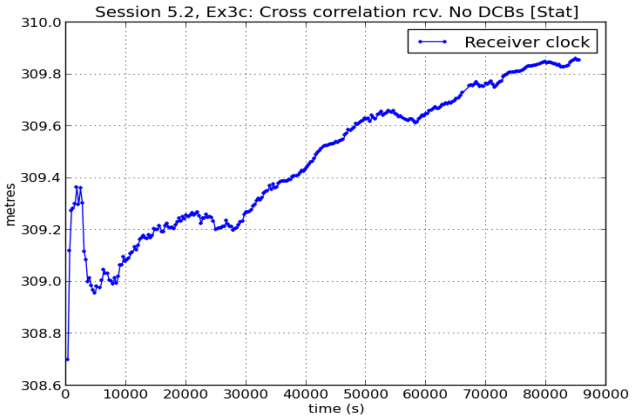

Session 5.2, Ex3b: C1 instead of P1 rcv. Strict vs. No DCBs [Kin]
North error
East error
UP error
error (m)
time (s)

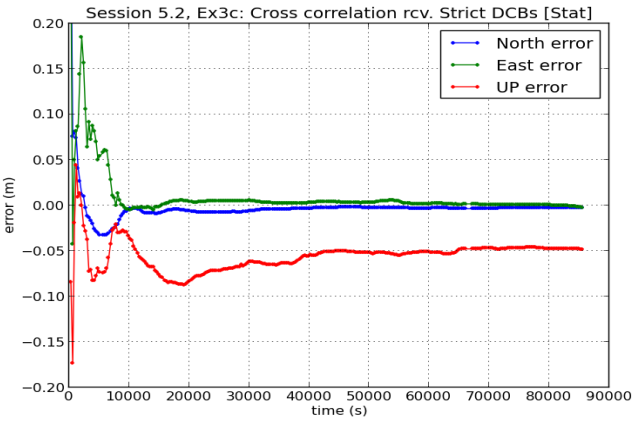

Session 5.2, Ex3b: C1 instead of P1 rcv. Strict - No DCBs [Kin]
Receiver clock
metres
time (s)

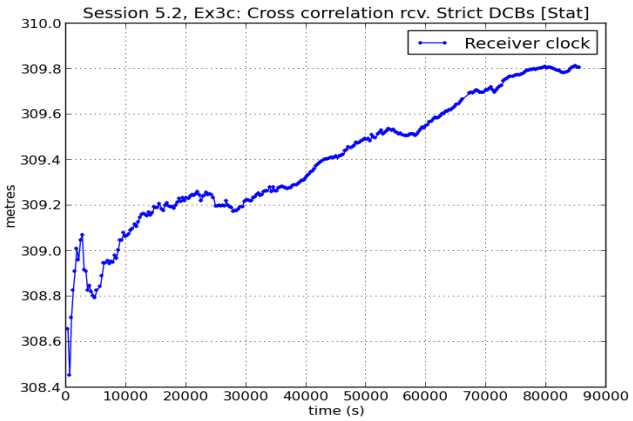

Session 5.2, Ex3b: C1 instead of P1 rcv. [Kin]
gLAB Clock_rcv: No DCBs
gLAB Clock_rcv: Strict
IGS Clock_rcv
metres
time (s)

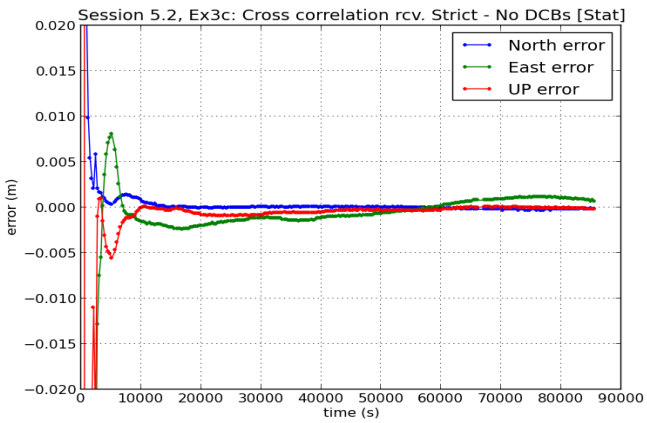

Session 5.2, Ex3b: Cross correlation rcv.: IGS - gLAB [Kin]
No DCBs
Receiver clock
metres
time (s)

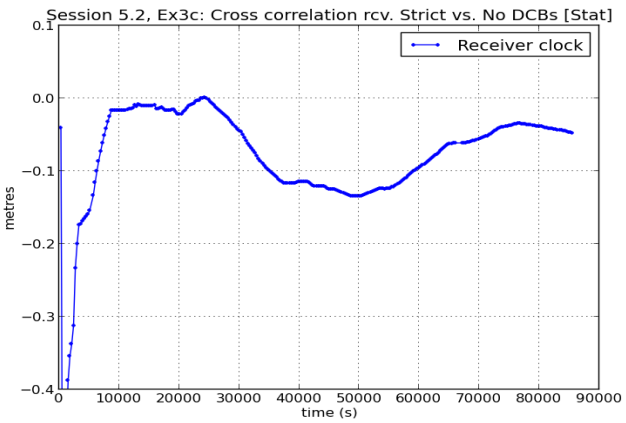

Session 5.2, Ex3c: Cross correlation rcv. No DCBs [Stat]
North error
East error
UP error
error (m)
time (s)

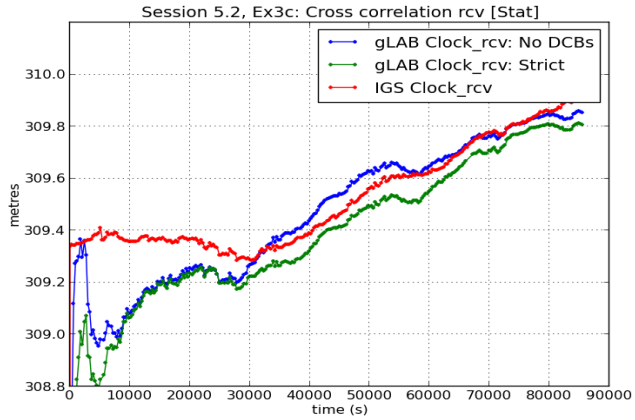

Session 5.2, Ex3c: Cross correlation rcv. No DCBs [Stat]
Receiver clock
metres
time (s)

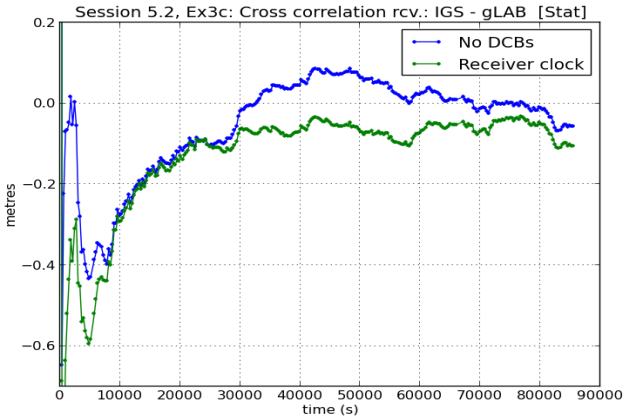

Session 5.2, Ex3c: Cross correlation rcv. Strict DCBs [Stat]
North error
East error
UP error
error (m)
time (s)

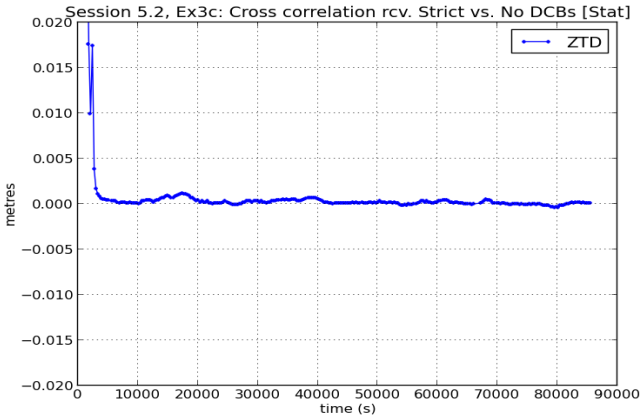

Session 5.2, Ex3c: Cross correlation rcv. Strict DCBs [Stat]
Receiver clock
metres
time (s)

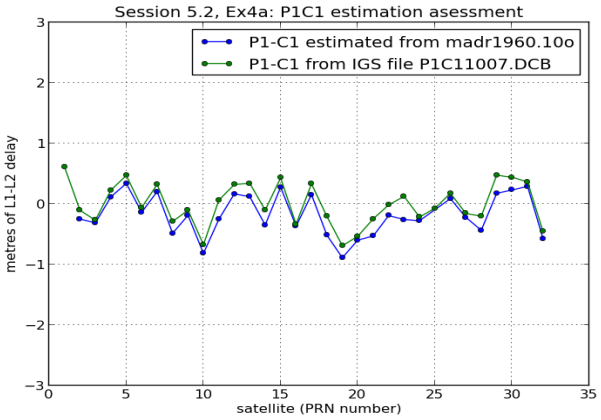

Session 5.2, Ex3c: Cross correlation rcv. Strict - No DCBs [Stat]
North error
East error
UP error
error (m)
time (s)


Session 5.2, Ex3c: Cross correlation rcv. Strict vs. No DCBs [Stat]
Receiver clock
metres
time (s)


Session 5.2, Ex3c: Cross correlation rcv [Stat]
gLAB Clock_rcv: No DCBs
gLAB Clock_rcv: Strict
IGS Clock_rcv
metres
time (s)


Session 5.2, Ex3c: Cross correlation rcv.: IGS - gLAB [Stat]
No DCBs
Receiver clock
metres
time (s)


Session 5.2, Ex3c: Cross correlation rcv. Strict vs. No DCBs [Stat]
ZTD
metres
time (s)


Session 5.2, Ex4a: P1C1 estimation asessment
P1-C1 estimated from madr1960.10o
P1-C1 from IGS file P1C11007.DCB
metres of L1-L2 delay
satellite (PRN number)

Session 5.3. Model Component Accuracy Assessment for GPS SPP

Objectives

To make a rough assessment of the error budget in the measurement modelling for SPP. This is done using actual measurements and comparing the different model terms (troposphere, ionosphere, orbits and clocks) with precise IGS references. This study includes prefit and postfit residual checking to show how range domain errors are transferred to the receiver coordinates and clock.

Files to use
`brus1810.09o`, `brdc1810.09n`, `brus1810.09zpd`, `igs09P1538.snx`, `gps_brd.atx`, `igs05_1525.atx`, `codg1810.09i`, `igs15382.clk_30s`, `igs15382.sp3`, `upcg1810.09i.Z`, `P1P20906.DCB.Z`

Programs to use
`gLAB_GUI.py`, `graph.py`

Development

Session files:
Copy and uncompress these session files in the working directory:

```
cp ~/GNSS/PROG/SES53/* .
cp ~/GNSS/FILES/SES53/* .
gzip -d *.gz *.Z
```

1. Positioning error and prefit residuals

The RINEX measurement file `brus1810.09o` will be processed in SPP mode as the starting point of this session. This data file has been collected by the permanent (i.e. fixed coordinates) IGS receiver BRUS, located in Brussels, Belgium.

Once the navigation solution is calculated, the positioning error (relative to a true reference) and the prefit residuals will be analysed.

(a) *Assessment of the SPP navigation solution*

Execute:

`gLAB_GUI.py &`

i. Click the `[Input]` section button and then click

`SPP template`

to compute the SPP solution with the RINEX measurement and broadcast files `brus1810.09o` and `brdc1810.09n`, respectively, using the default configuration.

ii. In the [Input] section, upload the RINEX measurement and broadcast files brus1810.09o and brdc1810.09n and select [Use SINEX File] to get the 'A priori receiver position' from the IGS file igs09P1538.snx.[77]

iii. In section [Preprocess] set Data decimation [30] seconds.

iv. In section [Modelling] set the option [☒ Receiver antenna reference point correction] and select [⊙ Read from RINEX].

v. In section [Output] set Print MEAS Messages and name the output file as 'gLAB_SPP.out'. Finally, to compute the solution click [Run gLAB].

Complete the following steps:

i. Plot the (ENU) Navigation System Error (NSE). Execute for instance:

Select the OUTPUT message (to make plot faster with a smaller file)

```
grep OUTPUT gLAB_SPP.out > output.tmp
```

Plot the results as a function of time:

```
graph.py -f output.tmp -x4 -y18 -f output.tmp -x4 -y19
         -f output.tmp -x4 -y20 --yn -10 --yx 10
```

Discuss the results taking into account that precise receiver coordinates from IGS SINEX files have been taken as a reference. What is the position accuracy? Is the obtained accuracy as expected for the SPS? Is there any global bias in the coordinate estimates?

ii. Plot the receiver clock estimate and compare the results with the precise determination of the BRUS receiver clock from IGS file igs15382.clk_30s.

Execute for instance:

From IGS file igs09P1538.snx generate a file with the following content [time(sec), dt_clock(m)]:

```
grep BRUS igs15382.clk_30s|gawk 'BEGIN{c=299792458}
          {print $6*3600+$7*60+$8,$10*c}' > brus_igs.clk
```

Plot the receiver clock

```
graph.py -f gLAB_SPP.out -x4 -y8 -c '($1=="FILTER")'
         -l "Estimated Clock_rcv" -f brus_igs.clk -x1 -y2
                     -l "IGS Clock_rcv" --yn 157 --yx 177
                        --xl "time (s)" --yl "error (m)"
```

Is there any correlation between clock and coordinate estimates? Why does a bias appear between the clock estimates and the IGS clock values?

Note: Some useful information can be found in exercise 5.

(b) *Prefit residuals*[78] *plotting and analysis*

From output file 'gLAB_SPP.out' generate a file with the following content: [PRN,time(sec),prefit(m),elev(deg)]. Plot the prefit residuals as a function of time and elevation.

[77] SINEX files provide surveyed coordinates accurate to centimetre level or better.

[78] Prefit residuals are defined in section 6.1 of Volume I.

i. Plot the prefit residuals.

Complete the following steps:

Generate the file with the prefit residual (exclude the rows labelled *, because they are not included in the filter):

```
grep PREFIT gLAB_SPP.out | grep C1C
 | grep -v \* | gawk '{print $6,$4,$8,$15}' > prefit.tmp
```

Plot the results as a function of time:

```
graph.py -f prefit.tmp -x2 -y3 -l "ALL"
        -f prefit.tmp -c '($1==04)' -x 2 -y 3 -l "PRN04"
        -f prefit.tmp -c '($1==15)' -x 2 -y 3 -l "PRN15"
        -f prefit.tmp -c '($1==24)' -x 2 -y 3 -l "PRN24"
       --yn 157 --yx 177 --xl "time (sec)" --yl "metres"
```

Plot the results as a function of elevation:

```
graph.py -f prefit.tmp -x4 -y3 -l "ALL"
        -f prefit.tmp -c '($1==04)' -x 4 -y 3 -l "PRN04"
        -f prefit.tmp -c '($1==15)' -x 4 -y 3 -l "PRN15"
        -f prefit.tmp -c '($1==24)' -x 4 -y 3 -l "PRN24"
       --yn 157 --yx 177 --xl "time (sec)" --yl "metres"
```

What might be the source of the bias of about 170 m appearing in the prefit residuals? Why is the noise higher at low elevations?

2. Tropospheric corrections

The accuracy of the tropospheric corrections computed by `gLAB` will be assessed in this exercise using the IGS tropospheric determinations. The tropospheric model applied by default in SPP mode is described in section 5.4.2.1, Volume I).

(a) *Zenith Tropospheric Delay (ZTD) assessment*

The IGS file `brus1810.09zpd` contains an accurate estimate (at the level of a few millimetres)[79] of the tropospheric delay at the zenith of the BRUS receiver. This estimate will be used as a reference value (i.e. the truth) to assess the tropospheric corrections applied by `gLAB` in SPP mode.

The ZTD used by `gLAB` in SPP mode is not directly provided in the output file `gLAB_SPP.out`, but it can be computed from the slant delay given in field #24 of message `MODEL` of this file. This can be done by un-projecting such a slant delay with the mapping function[80]

$$M(E) = \frac{1.001}{\sqrt{0.002\,001 + \sin^2(E)}} \tag{5.2}$$

where E is the satellite elevation (field #29 of message `OUTPUT`).

Complete the following steps:

i. From fields #24 and #29 of message `MODEL` of `gLAB_SPP.out` and using mapping function (5.2), obtain the nominal ZTD value

[79] See *http://igscb.jpl.nasa.gov/components/prods.html*.

[80] In SPP mode, `gLAB` applies the tropospheric model of section 5.4.2.1 in Volume I, which uses the mapping (5.2).

applied by gLAB. Execute for instance:

```
grep MODEL gLAB_SPP.out | grep -v INFO |
     gawk '{g2r=atan2(1,1)/45;
       M=1.001/sqrt(0.002001+sin($29*g2r)**2);
                          print $4,$24/M}' > nom.ztd
```

ii. Compare the obtained nominal value with the IGS estimate ZTD of station BRUS from file brus1810.09zpd.
Note: The ZTD of the IGS file is given in millimetres of delay.
Execute for instance:

```
grep BRUS brus1810.09zpd |gawk -F\: '{print $3}'|
            gawk '{print $1,$2/1000}' > brus.ztd
```

Plot the results:

```
graph.py -f brus.ztd -x1 -y2 -s.- -l "IGS"
            -f nom.ztd -x1 -y2 -s.- -l "Modelled"
--xl "time (s)" --yl "metres" --yn 2.40 --yx 2.50
```

What is the level of discrepancy? Is the error a zero-mean noise or a systematic error (i.e. mostly a bias)? How can it affect the position solution?

(b) *Slant tropospheric delay assessment*

Compare the Slant TROPospheric delay (STROP) modelled by gLAB with the IGS reference values projected in the slant direction.

Complete the following steps:

i. From message MODEL of gLAB_SPP.out, generate a file containing [PRN, time(sec), STRPg(m), elev(deg)], where STRPg is the slant tropospheric delay used by gLAB in the measurement modelling.
Execute for instance:

```
grep MODEL gLAB_SPP.out | grep -v INFO
         | gawk '{print $6,$4,$24,$29}' > trp.tmp
```

ii. Using mapping function (5.2), compute the slant delay[81] from the IGS ZTD estimates. Generate a file with the contents: [PRN, time(sec), STRPg(m), STRPo(m), STRPg-STRPo(m), elev(deg)], where STRPo is the slant reference value computed from the IGS vertical delays.

Execute for instance:

```
cat brus.ztd trp.tmp|gawk '{if (NF==2) {T[$1*1]=$2} else
  {if (length(T[$2*1])!=0)
   {ST=1.001/sqrt(0.002001+sin($4*3.14/180)**2)*T[$2*1];
              print $1,$2,$3,ST,$3-ST,$4}}}' > brus.trp
```

[81] The error due to the mapping function will be neglected in this assessment. Notice that the error introduced by this mapping is less than 20 cm for elevations over 5°, see question (2c).

iii. Plot the discrepancy between the slant tropospheric correction values used by gLAB and the reference values from IGS.
Execute for instance:

Discrepancy as a function of elevation:

```
graph.py -f brus.trp -x 6 -y 5 -l "Model-IGS"
--xl "elev (deg)" --yl "metres" --yn -1 --yx 0.2
```

Discrepancy as a function of time:

```
graph.py -f brus.trp -x 2 -y 5 -l "Model-IGS"
--xl "time (sec)" --yl "metres" --yn -1 --yx 0.2
```

(c) *Comparison of mapping functions*
The mapping function (5.2) used in the previous section is a simple approach for computing the obliquity factor. A more accurate mapping is the one proposed by Niell (see section 5.4.2.2.1, Volume I), which is used for PPP. This exercise is devoted to comparing the discrepancy between the slant delays computed from both mappings, as a function of the elevation.
Complete the following steps:

i. Set the same configuration as in exercise 1, but select the mapping of "Niell" for the tropospheric correction in the section [Modelling]. In the section [Output] set 'gLAB_SPPn.out' as the output file. Click Run gLAB to compute the solution.

ii. Calculate the discrepancy between the tropospheric corrections of file gLAB_SPPn.out, generated using the mapping of Niell, and file gLAB_SPP.out, generated in the previous exercise using the simple mapping given by equation (5.2).
Note that the same nominal values are applied in both cases.
Execute for instance:

Computing the discrepancy between both determinations:

```
grep MODEL gLAB_SPPn.out |grep -v INFO
   | gawk '{print $6,$4,$24,$29,0}' > trp.tmp0

grep MODEL gLAB_SPP.out |grep -v INFO
   | gawk '{print $6,$4,$24,$29}' > trp.tmp

cat trp.tmp0 trp.tmp|gawk '{if(NF==5){T[$1" "$2]=$3}
    else {if (length(T[$1" "$2])!=0)
                {print $0,$3-T[$1" "$2]}}}' > map.dat
```

Plot the results (over 0°):

```
graph.py -f map.dat -x 4 -y 5 -l "SIMPLE-NIELL"
         --xn 0 --xl "elevation (deg)" --yl "metres"
```

Plot the results (over 5°):

```
graph.py -f map.dat -x 4 -y 5 -l "SIMPLE-NIELL"
   --xn 4 --yn -0.01 --yx 0.2 --xl "elevation (deg)"
```

What is the discrepancy between both slant determinations for elevations over 5°? And below 5°?
Note: gLAB *applies an elevation mask of* 5° *by default.*

3. Ionospheric corrections

The accuracy of Klobuchar ionospheric corrections computed by gLAB to account for the ionospheric delay is assessed in this exercise.

Dual-frequency code and carrier measurements will be used to obtain the ionospheric refraction – with the code instrumental delays added, see equation (5.3). The satellite and receiver instrumental delays will be taken from the IGS IONEX files.[82]

As explained in equations (4.19) of Volume I, the ionosphere-free combination of two-frequency codes, or carriers, provides the ionospheric delay plus a code bias, or carrier ambiguity:[83]

$$\begin{aligned} P_2 - P_1 &= I + K_{21} \\ L_1 - L_2 &= I + \text{ambiguity} \end{aligned} \tag{5.3}$$

where I is the Slant Total Electron Content (STEC) in L1–L2 units of delay, see equation (4.16) of Volume I.

K_{21} is the instrumental delay or DCB between P2 and P1 codes (in L1–L2 delay units; see equation (4.11) in Volume I), which is decomposed into a receiver and a satellite term:

$$K_{21} = K_{21_{rcv}} - K_{21}^{sat} \tag{5.4}$$

In the SPP solution, the ionospheric delay is computed with the Klobuchar model (i.e. $\mathtt{I1}_{\mathtt{Klob}}$), whose parameters are broadcast in the navigation message, together with the satellite instrumental delay (i.e. $\mathtt{TGD}_{\mathtt{brd}}$), see details below. The receiver instrumental delay, on the other hand, is assimilated into the receiver clock estimate in the navigation solution, because it is common to all satellites.

The target of this exercise is to assess (roughly) the accuracy of the combined terms $\mathtt{I1}_{\mathtt{Klob}} + \mathtt{TGD}_{\mathtt{brd}}$ used in the SPP model to account for the ionospheric refraction and the satellite instrumental delay.

In this simple exercise, IONEX files will be used to get an estimate of the receiver instrumental delay. This instrumental delay, after being aligned with the broadcast satellite TGD, will be combined with this satellite TGD and the Klobuchar ionospheric correction to check the consistence of the first equation of equations (5.3). Due to the large code noise, only a global ionospheric fit, after removing the instrumental delays, will be possible to assess.

The shape of the slant ionospheric delay, involving the mapping function, will be assessed from carrier measurements, according to the second equation of equations (5.3), shifted locally to match the computed ionospheric delays ($\mathtt{I1}_{\mathtt{Klob}}$).

Before starting the exercise, some remarks on the contents and units of GPS broadcast navigation messages and IONEX files are needed.

[82] The agreement between the different IGS centres in the GPS P1–P2 DCB estimates is at the level of a tenth of a nanosecond for the satellites and a few nanoseconds for the receivers [Hernández-Pajares, 2004]. See also exercise 6e.

[83] Where the measurement noise, the APC correction and the carrier wind-up have been neglected for clarity.

GPS navigation message:

- The satellite instrumental delay K_{21}^{sat} is broadcast in the GPS navigation message as the $\mathtt{TGD}_{\mathtt{brd}}$, in L1 delay units:[84]

$$K_{21}^{sat} = -(\gamma_{12} - 1)\,\mathtt{TGD}_{\mathtt{brd}} \tag{5.5}$$

  with $\gamma_{12} = (f_1/f_2)^2 = (77/60)^2$, see equation (5.20) in Volume I.

- The slant ionospheric delay I is computed from the Klobuchar model, whose coefficients are broadcast in the GPS navigation message. Indeed, the Klobuchar model gives $\mathtt{I}_{1_{\mathtt{Klob}}}$ in L1 delay units, thus

$$I = (\gamma_{12} - 1)\,\mathtt{I}_{1_{\mathtt{Klob}}} \tag{5.6}$$

Note that $\mathtt{TGD}_{\mathtt{brd}}$ is given in seconds (of L1 delay) in RINEX navigation files. $\mathtt{I}_{1_{\mathtt{Klob}}}$ is in metres (of L1 delay) in the `gLAB_SPP.out` file.

IONEX files:

- Both satellite and receiver instrumental delays are provided in the IONEX files (i.e. $K_{rcv_{IONEX}}$ and K_{IONEX}^{sat}). Nevertheless, to match previous definitions the following issues must be taken into account:
  (a) The DCBs are given in L1–L2 units of delay (not in L1 units as in the GPS RINEX navigation file).
  (b) The DCBs are given for P1–P2, not P2–P1, as considered previously, and according to the model

$$P_1 - P_2 = -I + \mathtt{K}_{\mathtt{rcv}_{\mathtt{IONEX}}} + \mathtt{K}_{\mathtt{IONEX}}^{\mathtt{sat}} \tag{5.7}$$

  Thus,

$$\begin{aligned} K_{21_{rcv}} &\rightarrow -\mathtt{K}_{\mathtt{rcv}_{\mathtt{IONEX}}} \\ K_{21}^{sat} &\rightarrow +\mathtt{K}_{\mathtt{IONEX}}^{\mathtt{sat}} \\ \mathtt{TGD}_{\mathtt{brd}} &\rightarrow -\tfrac{1}{\gamma_{12}-1}\,\mathtt{K}_{\mathtt{21}_{\mathtt{IONEX}}}^{\mathtt{sat}} \end{aligned} \tag{5.8}$$

  (c) Usually the sum of IONEX DCBs is constrained to zero for the satellite values. So a bias is usually found when comparing the TGDs from RINEX navigation files with $-\frac{1}{\gamma_{12}-1}\mathtt{K}_{\mathtt{IONEX}}^{\mathtt{sat}}$. This is a common bias for all satellites, in that

$$\begin{aligned} P_2 - P_1 = &\; I - \mathtt{K}_{\mathtt{rcv}_{\mathtt{IONEX}}} - \mathtt{K}_{\mathtt{IONEX}}^{\mathtt{sat}} \\ = &\; (\gamma_{12} - 1)(\mathtt{I}_{1_{\mathtt{Klob}}} + \mathtt{K}_{1_{\mathtt{rcv}}} + \mathtt{TGD}_{\mathtt{brd}}) \end{aligned} \tag{5.9}$$

  where $\mathtt{K}_{1_{\mathtt{rcv}}}$ can be computed as

$$\mathtt{K}_{1_{\mathtt{rcv}}} = -\mathtt{TGD}_{\mathtt{brd}} - \frac{1}{\gamma_{12} - 1}(\mathtt{K}_{\mathtt{rcv}_{\mathtt{IONEX}}} + \mathtt{K}_{\mathtt{IONEX}}^{\mathtt{sat}}) \tag{5.10}$$

[84] Notice that $1/(\gamma_{12} - 1) = 1.546$ is the conversion factor between L1–L2 and L1 units of delay.

- Note that IONEX files provide global *vertical ionospheric delays*, although such ionospheric corrections are not used in this exercise.[85] An obliquity factor, or mapping function, based on a single-layer model at a fixed height, is used to compute the slant delays. Details can be found at the following URL: *http://igscb.jpl.nasa.gov/components/formats.html*.

Remark: In IONEX files, `DCBs` are given in nanoseconds (of L1–L2 delay) and the vertical ionospheric delay is typically expressed in units of 0.1 TECUs.[86]

Exercise development.

(a) *Receiver DCB*

Using expression (5.10), compute the receiver instrumental delay $\mathtt{K}_{1_{\mathtt{rcv}}}$ for the receiver BRUS, aligned with the TGDs broadcasted in the GPS navigation message.

The IONEX file `codg1810.09i` can be used to get the $\mathtt{K}_{\mathtt{rcv}_{\mathtt{IONEX}}}$ and $\mathtt{K}^{\mathtt{sat}}_{\mathtt{IONEX}}$ instrumental delays. The broadcast TGDs can be taken from the RINEX navigation file `brdc1810.09n`, or field #27 of message `MODEL` in the `gLAB_SPP.out` file, as well.

Solution: $\mathtt{K}_{1_{\mathtt{rcv}}} = 4.4914$ m of L1 delay.

Hint:
$c = 299\,792\,458 m/s$ *and* $\gamma_{12} = (77/60)^2$.
Take, for instance, satellite PRN15. Then, from GPS navigation message `brdc1810.09n`,
$\mathtt{TGD}_{\mathtt{brd}} = -0.977\,888\,703\,346$*E-08 s (of L1 delay)*
From IONEX file `codg1810.09i`
$\mathtt{K}^{\mathtt{sat}}_{\mathtt{IONEX}} = 0.466$ *ns,* $\mathtt{K}_{\mathtt{rcv}_{\mathtt{IONEX}}} = -3.832$ *ns (of L1–L2 delay).*

(b) *Ionospheric delay plus DCB assessment*

Make a rough assessment of the combined value $\mathtt{I}_{1_{\mathtt{Klob}}} + \mathtt{K}_{1_{\mathtt{rcv}}} + \mathtt{TGD}_{\mathtt{brd}}$ using the geometry-free combination of codes P2–P1.
Notice that, due to the large measurement noise of P codes,[87] this assessment will be reduced to a simple check of the global bias between both determinations according to the first equation of equations (5.3). That is,

$$\mathtt{I}_{1_{\mathtt{Klob}}} + \mathtt{K}_{1_{\mathtt{rcv}}} + \mathtt{TGD}_{\mathtt{brd}} \simeq \frac{1}{\gamma_{12} - 1}(P_2 - P_1) \tag{5.11}$$

[85] The accuracy of the Vertical TEC of IONEX is about 2–8 TECUs (see *http://igscb.jpl.nasa.gov/components/prods.html*). This accuracy can be assumed to be 3 TECUs (about 45 cm of L1 delay) over well-covered areas (e.g. Europe) in undisturbed conditions.

[86] 1 TECU corresponds to $\alpha_1 = 40.3 \cdot 10^{16}/(154 \times 10.23 \cdot 10^6)^2 = 0.1624$ m of delay in the L1 signal (see eq. 4.10 of Volume I), or $\alpha_2 - \alpha_1 = 0.105\,05$ m of delay in the L1–L2 combination (see Table 4.2, Volume I).

[87] A better plot could be done using carrier smoothed code, with a 100-sample window (for instance), but it would require higher sampling rate data (e.g. 1 s) to avoid an excessive divergence of the ionosphere (see Fig. 4.6, Volume I). No smoothing is applied in this exercise, because the sampling rate of the RINEX file `brus1810.09o` is 30 s.

Complete the following steps:

i. Using the `gLAB_SPP.out` output file, compute the geometry-free combination of codes P2 and P1 from message `MEAS` of the file `gLAB_SPP.out` and generate the file `PI.dat` containing:
`[PRN, time(sec), P2-P1(`m_{L1}`), elev(deg)]`
where P2–P1 is in metres of L1 delay. Execute for instance:

```
grep MEAS gLAB_SPP.out | grep -v INFO |
    gawk '{print $6,$4,1.5457*($15-$13),$7}' > PI.dat
```

ii. Using the `gLAB_SPP.out` output file, and $K_{1_{rcv}} = 4.4914\,m_{L1}$ obtained above, compute $I_{1_{Klob}} + K_{1_{rcv}} + TGD_{brd}$.
Generate a file containing `[PRN,time(sec),`$I_{1_{Klob}}$`+`$K_{1_{rcv}}$`+`TGD_{brd}`]`.
Execute for instance:

```
grep MODEL gLAB_SPP.out | grep -v INFO | grep C1 |
gawk '{print $6,$4,$25+4.4914+$27}' >iono_Krcv_TGD.tmp
```

iii. From files `PI.dat` and `iono_Krcv_TGD.tmp`, generate a new file containing the fields `[PRN, time, Model, Error]`, where `Model` = $I_{1_{Klob}}$+$K_{1_{rcv}}$+TGD_{brd} and `Error`=`Model`–1.5457×(`P2-P1`).
Execute for instance:

```
cat PI.dat iono_Krcv_TGD.tmp|
gawk '{if (NF==4){I[$1" "$2]=$3;E[$1" "$2]=$4}
         else{if (length(E[$1" "$2])!=0)
print $0,$3-I[$1" "$2],E[$1" "$2]}}'>iono_Krcv_TGD.dat
```

iv. Plot the results:
A. Check the global fit (i.e. bias) between $I_{1_{Klob}}$+$K_{1_{rcv}}$+TGD_{brd} and 1.5457×(`P2-P1`):

```
graph.py -f PI.dat -x 2 -y3 -l "1.5457(P2-P1)"
         -f iono_Krcv_TGD.dat -l "KLOB+Krcv+TGD"
                     --cl r -x 2 -y 3 --yn -10 --yx 20
```

B. Plot as a function of the elevation the discrepancies between the determinations $I_{1_{Klob}}$+$K_{1_{rcv}}$+TGD_{brd} and 1.5457×(`P2-P1`). Highlight the satellite PRN15 as well:

```
graph.py -f iono_Krcv_TGD.dat -x 5 -y 4
     -f iono_Krcv_TGD.dat -c '($1==15)' -x 5 -y 4 -so
     --xl "elev" --yl "m of L1 delay" --yn -10 --yx 10
```

C. Produce the same plot as before, but as a function of time.

```
graph.py -f iono_Krcv_TGD.dat -x 2 -y 4
      -f iono_Krcv_TGD.dat -c '($1==15)' -x 2 -y 4 -so
      --xl "sec" --yl "m of L1 delay" --yn -10 --yx 10
```

(c) *Ionospheric corrections shape assessment with carrier measurements*

Assess the shape of the Klobuchar slant ionospheric corrections by comparing with L1–L2 carrier determinations, shifted to align with the Klobuchar values.

Complete the following steps:

i. Using the `gLAB_SPP.out` file, compute the geometry-free combi-

nation of carriers L1–L2 from message `MEAS` in `gLAB_SPP.out`. Generate a file with the following contents:

`[PRN, time(sec), L1-L2(m`$_{L1}$`), elev(deg)]`

where L1–L2 is in metres of L1 delay.
Execute for instance:

```
grep MEAS gLAB_SPP.out | grep -v INFO |
     gawk '{print $6,$4,1.5457*($14-$16),$7}' > LI.dat
```

ii. Plot the results. Generate the following plots for satellite PRN15: Overlap in the same plot the L1–L2 geometry-free combination of carriers and the $\mathtt{I}_{1_{\mathtt{Klob}}}+\mathtt{K}_{1_{\mathtt{rcv}}}+\mathtt{TGD}_{\mathtt{brd}}$ modelled values. Shift the L1–L2 carriers to align with the modelled values at the lowest elevation point. Make one plot a function of the elevation and another a function of time. Add the code P2–P1 values to the plots.

Execute for instance:

Plot the results as a function of elevation:

```
graph.py -f PI.dat -c'($1==15)' -x 4 -y3 -l "P2-P1"
-f LI.dat -c'($1==15)' -x 4 -y'($3+16.5)' -l "L1-L2"
-f iono_Krcv_TGD.dat -c '($1==15)' -l "KLOB+Krcv+TGD"
-x 5 -y 3 --xn 0 --xx 90 --yn 0 --yx 10
```

Note a whole arc is seen (from $elev_{min_o} \rightarrow elev_{max} \rightarrow elev_{min_f}$).

Plot the results as a function of time:

```
graph.py -f PI.dat -c'($1==15)' -x 2 -y3 -l "P2-P1"
-f LI.dat -c'($1==15)' -x 2 -y'($3+16.5)' -l "L1-L2"
-f iono_Krcv_TGD.dat -c '($1==15)' -l "KLOB+Krcv+TGD"
-x 2 -y 3 --xn 30000 --xx 60000 --yn 0 --yx 10
```

What is the level of discrepancy at low elevation? Why is a larger error found around 55 000 s than around 35 000 s?

4. Broadcast orbit and clock errors

The range error[88] due to the broadcast orbits and clocks will be assessed in this exercise using the IGS precise orbits and clocks as reference values. This will be done for each component individually (i.e. orbits and clocks), and for the combined result of 'satellite orbits+clocks'.

To make a proper assessment, it must be taken into account that the IGS products are linked to satellite APC values different from those associated with GPS broadcast orbits and clocks (for more details see section 5.6.3, Volume I). This issue will also be analysed here.

(a) *Orbits*: *Range error due to broadcast orbit accuracy*

A direct assessment of the range error due to the broadcast orbit accuracy can be done by comparing the geometric range between the receiver and satellite APCs computed using broadcast against

[88] The analysis of GPS broadcast orbit accuracy in radial, along-track and cross-track form, as well as the satellite clocks, was extensively done in session 3.2. The analysis here is devoted only to assessing the error in range.

those computed using the IGS precise products.[89]
As a first step for this part of the exercise, the SPP and PPP solutions and model components must be computed as follows:

i. *SPP solution computation.* Repeat the SPP processing as in exercise 1, or use file `gLAB_SPP.out` generated previously.

ii. *PPP solution computation with APC file* `gps_brd.atx`.[90] Set the default configuration to obtain the PPP solution (i.e. the `PPP Template`) with the RINEX measurement file `brus1810.09o`. Use the ANTEX file `gps_brd.atx` for the satellite APCs. Then, in the `[Input]` section, set the option `[⊙ Precise(2 files)]` and select the precise orbit file `igs15382.sp3` and the clock file `igs15382.clk_30s`. Then select `[Use SINEX File]` in the 'A priori receiver position' and select the `igs09P1538.snx` file.
In section `[Preprocess]`, set `Data decimation` to `30` s. In section `[Modelling]` unset the `[□ Receiver antenna phase centre correction]`[91] and unset the option `[□ Solid Tides correction]`.[92] In section `[Output]`, set `gLAB_PPP.out` as the output file.
Click `Run gLAB` to compute the solution.

Note: The broadcast orbits and clocks are also referred to the APC in the ionosphere-free combination PC.

iii. *Error in range due to the broadcast orbits.* Compute the discrepancy between the geometric ranges of files `gLAB_SPP.out` and `gLAB_PPP.out`.
Complete the following steps:[93]

```
grep MODEL gLAB_PPP.out | grep -v INFO |
    gawk '{printf "%s %s %12.3f %s \n",
                    $6,$4,$17+$19,"P"}' > oc.tmp0

grep MODEL gLAB_SPP.out | grep -v INFO |
                   gawk '{print $6,$4,$17}' > oc.tmp

cat oc.tmp0 oc.tmp | gawk '{if(NF==4){T[$1" "$2]=$3}
           else {if (length(T[$1" "$2])!=0)
                 {print $0,$3-T[$1" "$2]}}}' > oc.dat
```

[89] The IGS precise orbits and clocks are accurate at the level of a few centimetres, see *http://igscb.jpl.nasa.gov/components/prods.html*.

[90] The ANTEX file `gps_brd.atx` is used for the PPP modelling in order to refer the precise coordinates to the same satellite APC as the broadcast orbits, see exercise 2 in session 3.2.

[91] As the ANTEX file `gps_brd.atx` compiled for `gLAB` does not contain the receiver APCs for receivers, then the Antenna Reference Point (ARP) will be used instead of the receiver APC. Notice that this will not affect the comparison, because both range computations SPP and PPP will be done under the same conditions. On the other hand, in our case, the APC bias between L1 and L2 is less than 3 cm.

[92] Note that no solid tides were applied in the SPP processing.

[93] The `Satellite-receiver geometric distance` given in the `MODEL` output message of `gLAB` is taken from the `A priori receiver position` to satellite coordinates. Such satellite coordinates correspond to the satellite APC when they are computed from broadcast orbits, and to the MC when they are computed from the IGS `*.sp3` orbit files. This is the reason why the satellite APC correction (`"$19"`) is added to the satellite coordinates (`"$17"`) in the computation of geometric range `oc.tmp0` from `gLAB_PPP.out` in the sentence that follows.

Plot the results, showing PRN04, 15 and 24, as well:

```
graph.py -f oc.dat -x 2 -y 4 -l "ALL"
      -f oc.dat -c '($1==04)' -x 2 -y 4 -l "PRN04"
      -f oc.dat -c '($1==15)' -x 2 -y 4 -l "PRN15"
      -f oc.dat -c '($1==24)' -x 2 -y 4 -l "PRN24"
      --yn -9 --yx 5 --xl "time (sec)" --yl "metres"
```

What is the orbit accuracy in range?

iv. Repeat the previous analysis, but using `igs05_1525.atx` as the ANTEX file instead of `gps_brd.atx` with the IGS APC in the PPP processing.
Is a large error expected to be found?

(b) *Orbits and clocks*: *Range error due to combined broadcast orbits and clocks*

Complete the following steps:

i. *SPP solution computation.* Repeat the SPP computation as in the previous exercise, or use file `gLAB_SPP.out` computed previously, if available.

ii. *PPP solution computation (with IGS APC file).* Repeat the previous PPP processing, but using ANTEX file `igs05_1525.atx` with the IGS APC, instead of `gps_brd.atx`.

iii. *Error in range due to the combined orbits+clocks value.* Compute the discrepancy between the geometric 'range plus satellite clock' of files `gLAB_SPP.out` and `gLAB_PPP.out`:

```
grep MODEL gLAB_PPP.out| grep -v INFO| gawk '{printf
"%s %s %10.4f %s \n",$6,$4,$17+$19+$18,"P"}'>oc.tmp0

grep MODEL gLAB_SPP.out| grep -v INFO| gawk '{printf
"%s %s %10.4f \n",$6,$4,$17+$18}' > oc.tmp

cat oc.tmp0 oc.tmp|gawk '{if(NF==4){T[$1" "$2]=$3}
             else{if (length(T[$1" "$2])!=0)
                {print $0,$3-T[$1" "$2]}}}' > oc.dat
```

Note: As in the previous case, `printf`*, with its explicit format, has been used in* `gawk` *to avoid decimal truncation in the sum.*

Plot the results, showing PRN04, 15 and 24 explicitly, as well:

```
graph.py -f oc.dat -x 2 -y 4 -l "ALL"
      -f oc.dat -c '($1==04)' -x 2 -y 4 -l "PRN04"
      -f oc.dat -c '($1==15)' -x 2 -y 4 -l "PRN15"
      -f oc.dat -c '($1==24)' -x 2 -y 4 -l "PRN24"
      --yn -9 --yx 5 --xl "time (sec)" --yl "metres"
```

What is the orbit and clock accuracy in range? Is there any bias common to all satellites? How would a common bias affect the navigation solution (position and time)?

iv. Repeat the previous analysis, but using `gps_brd.atx` as the ANTEX file with the IGS APC in the PPP processing, instead of the file `igs05_1525.atx`.

Why does a larger error than in previous case appear?

5. Prefit and postfit residual analysis

(a) *Assessment of the SPP navigation solution*

Using `gLAB_SPP.out` generated in exercise 1, complete the following steps:

i. Plot the (ENU) Navigation System Error (NSE):
Execute for instance:

Select the `OUTPUT` message (to make plot faster with a smaller file):

```
grep OUTPUT gLAB_SPP.out > output.tmp
```

Plot the results as a function of time:

```
graph.py -f output.tmp -x4 -y18 -f output.tmp -x4 -y19
         -f output.tmp -x4 -y20 --yn -10 --yx 10
```

Discuss the results taking into account the error budget seen in previous exercises for the different model components. What is the position accuracy?

ii. Plot the receiver clock estimate and compare the results with the IGS precise determination from file `igs15382.clk_30s`.

Check the bias between both determinations. Justify this bias taking into account previous results.

Hint: The instrumental delay of the BRUS receiver is $\mathtt{K}_{1_{\mathtt{rcv}}} = 4.4914\,\mathrm{m}_{\mathtt{L1}}$, and an error of about 4–5 m, common to all satellites, has been found in the modelled geometric range when analysing the orbit and clock accuracy (exercise 4b).

Execute for instance:

Compute the BRUS receiver clock:

```
grep BRUS igs15382.clk_30s | gawk 'BEGIN{c=299792458}
          {print $6*3600+$7*60+$8,$10*c}' > brus_igs.clk
```

Plot the receiver clock:

```
graph.py -f gLAB_SPP.out -x4 -y8 -c '($1=="FILTER")'
         -l "Estimated Clock_rcv"
         -f brus_igs.clk -x1 -y'($2 + 4.4914 + 4.5)'
         -l "IGS Clock_rcv + K1_rcv + Geom.Range error"
         -f brus_igs.clk -x1 -y2 -l "IGS Clock_rcv"
         --yn 157 --yx 177
```

(b) *Prefit residual plotting and analysis*

From output file `gLAB_SPP.out` generate a file with the following contents: `[PRN, time(sec), prefit(m), elev(deg)]`. Plot the prefit residuals as a function of time and elevation.

i. *Prefit residual bias analysis.* As the precise coordinates[94] of the BRUS receiver have been used in the computation of the SPP solution, the prefit residual bias should be mainly associated

[94] Actually the receiver reference point coordinates have been used, not the APC. Nevertheless, they differ in few centimetres.

with the receiver clock and receiver instrumental delay, among other biases common to all satellites.
To assess the previous sentence, the receiver clock and instrumental delay are going to be removed from the prefit residuals as follows:

A. *Removing receiver clock and instrumental delays from prefit residuals.* Accurate estimation of the BRUS receiver clock is provided in the IGS file `igs15382.clk_30s`. The instrumental delay of the BRUS receiver was computed in a previous exercise as $\mathtt{K1}_{\mathtt{rcv}} = 4.4914\,\mathrm{m}$ of L1 delay.
Using the previous files and results, generate a file containing [`PRN`, `sec`, `prefit`, `elev`, `prefit-clock`$_{rcv}$-$\mathtt{K1}_{rcv}$].
Plot the results as a function of time and elevation.

Complete the following steps:

Compute the BRUS receiver clock:

```
grep BRUS igs15382.clk_30s|gawk 'BEGIN{c=299792458}
     {print $6*3600+$7*60+$8,$10*c}' > brus_igs.clk
```

Remove the receiver clock and instrumental delay from prefits:

```
cat brus_igs.clk prefit.tmp|gawk '{if (NF==2)
   {c[$1*1]=$2}else{if (length(c[$2*1])!=0)
      {print $0,$3-c[$2*1]-4.4914}}}' > prefit.dat
```

Plot the results as a function of time:

```
graph.py -f prefit.dat -x2 -y5 -l "ALL"
    -f prefit.dat -c '($1==04)' -x 2 -y 5 -l "PRN04"
    -f prefit.dat -c '($1==15)' -x 2 -y 5 -l "PRN15"
    -f prefit.dat -c '($1==24)' -x 2 -y 5 -l "PRN24"
    --yn -10 --yx 10 --xl "time (sec)" --yl "metres"
```

Plot the results as a function of elevation:

```
graph.py -f prefit.dat -x4 -y5 -l "ALL"
    -f prefit.dat -c '($1==04)' -x 4 -y 5 -l "PRN04"
    -f prefit.dat -c '($1==15)' -x 4 -y 5 -l "PRN15"
    -f prefit.dat -c '($1==24)' -x 4 -y 5 -l "PRN24"
    --yn -10 --yx 10 --xl "elev (deg)" --yl "metres"
```

There is still a bias of about 4–5 m. Taking into account previous results, discuss what the source of this bias is.

Hint: See the previous question and the geometric range error due to orbit and clock accuracy (exercise 4b).

(c) *Postfit residual analysis*

Using the previous output file `gLAB_SPP.out`, plot the postfit residuals. Show explicitly the satellites PRN04, 15 and 24.

Generate the postfit residual file:

```
grep POSTFIT gLAB_SPP.out | grep C1C |
            gawk '{print $6,$4,$8,$11}' > postfit.dat
```

Plot postfits as a function of time:

```
graph.py -f postfit.dat -x2 -y3 -l "ALL"
  -f postfit.dat -c '($1==04)' -x 2 -y 3 -l "PRN04"
  -f postfit.dat -c '($1==15)' -x 2 -y 3 -l "PRN15"
  -f postfit.dat -c '($1==24)' -x 2 -y 3 -l "PRN24"
  --yn -10 --yx 10 --xl "time (sec)" --yl "metres"
```

Plot postfits as a function of elevation:

```
graph.py -f postfit.dat -x4 -y3 -l "ALL"
  -f postfit.dat -c '($1==04)' -x 4 -y 3 -l "PRN04"
  -f postfit.dat -c '($1==15)' -x 4 -y 3 -l "PRN15"
  -f postfit.dat -c '($1==24)' -x 4 -y 3 -l "PRN24"
  --yn -10 --yx 10 --xl "elev (deg)" --yl "metres"
```

Roughly analyse the postfit residuals:

(a) Are outliers appearing? Do they look like white noise?

(b) Are the postfit residuals elevation dependent?

(c) Are similar dispersions found for all satellites?

(d) From the postfit residual plots, give an estimate of the measurement noise. Can the same σ_y values be used for all satellites?

(e) Which value has been used in the `gLAB` computations?[95] Has the same σ_y value been used for all satellites?

(f) What is affecting the solution: the absolute value of the different noise terms used in the filter or their relative values?

6. Comparison of P1–P2 determinations

The agreement between broadcast TGDs and the IGS P1–P2 DCBs will be assessed in this exercise. Note that broadcast TGDs have been used in exercise 3 but without discussing their accuracy.

Different determinations of P1–P2 DCBs from IGS file `P1P20906.DCB`, IONEX file `upcg1810.09i` and broadcast in the GPS message (from file `brdc1810.09n`) are compared next.

Complete the following steps:

(a) Using `gLAB`, generate a data file with the TGDs of the GPS broadcast message.
This can be done, for instance, by processing the `brus1810.09o` file with the default configuration of SPP mode, using the broadcast file `brdc1810.09n`. The broadcast TGDs are given in the `OUTPUT` message of output file `gLAB.out`.

[95] See the `StdDev` value used by `gLAB` in the `"Measurement configuration and noise"` box in the `[Modelling]` section of `gLAB` GUI.

Execute for instance:

```
The TGDs can be obtained in metres of L1-L2 delay
as follows (note the change in sign of the TGDs, see equation (5.8)):
grep MODEL gLAB.out | grep -v INFO | grep C1 |
       gawk 'BEGIN{g12=(154/120)^2}
                 {print $6,-(g12-1)*$27}'|sort -nu > TGD.dat
```

(b) Generate a file with the P1–P2 DCBs of the IONEX file `upcg1810.09i` and change the units from nanoseconds to metres of `L1-L2` delay.
Execute for instance:

```
grep "PRN / BIAS" upcg1810.09i | gawk 'BEGIN{c=299792458}
                              {print $1,$2*c*1e-9}' > p1p2.ionx
```

(c) Generate a file with the DCBs of GPS satellites from `P1P20906.DCB` and change the units from nanoseconds to metres of delay.
Execute for instance:

```
grep G P1P20906.DCB | gawk 'BEGIN{c=299792458}
       {if(NF==3)print substr($1,2,3),$2*c*1e-9}' > p1p2.igs
```

(d) Compare the different determinations in a plot:
Execute for instance:

```
graph.py -f TGD.dat -x1 -y'($2-1.7)' -l "brdc1810.09n"
              -f p1p2.ionx  -l "IGS IONEX file upcg1810.09i"
              -f p1p2.igs  -l "IGS DCB file P1P20906.DCB"
```

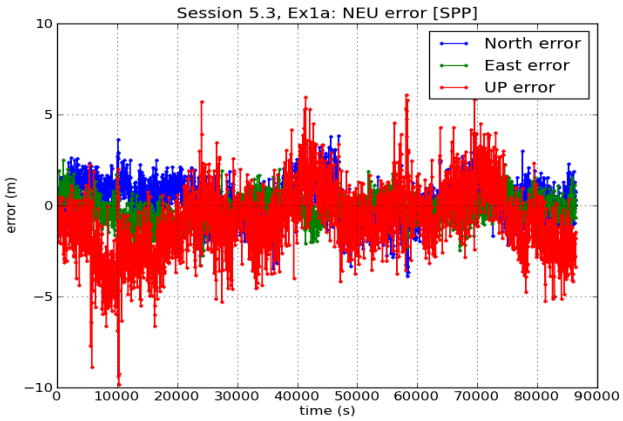
(e) Calculate and plot the discrepancies between the TGD broadcast in the GPS navigation message (file `brdc1810.09n`) and the IGS determinations from DCB file `P1P20906.DCB` and IONEX file `upcg1810.09i`. Give the results in metres of L1–L2 delay.
Execute for instance:

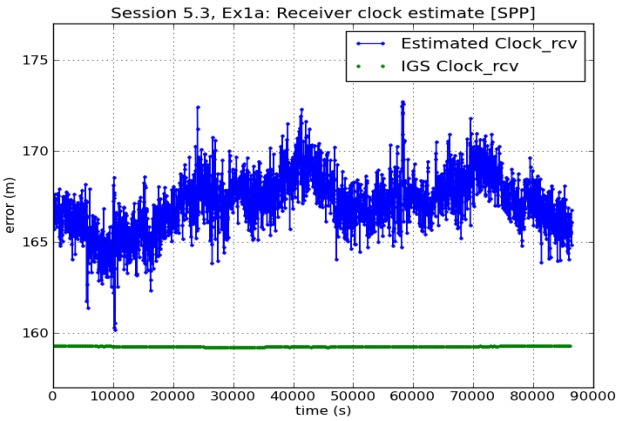
```
Compute the discrepancies between Broadcast and IGS file DCBs
(Note: The common bias is removed):
cat TGD.dat p1p2.igs|gawk '{prn=$1*1;if (length(v[prn])==0)
    {v[prn]=$2}else{d[prn]=v[prn]-$2;m=m+d[prn];n=n+1}}
        END{for (i in d) print i,d[i]-m/n}'|sort -n -k1 > d1
```

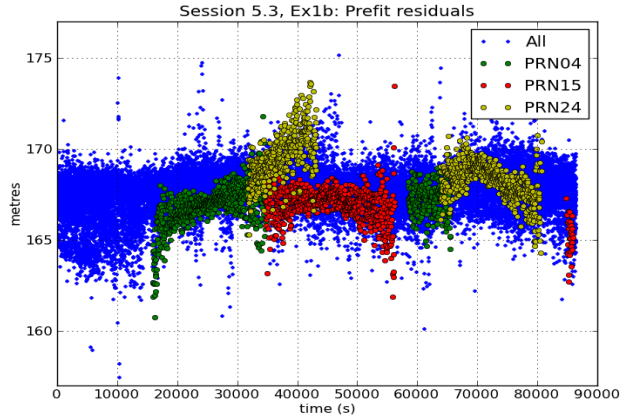
```
Compute the discrepancies between Broadcast and IONEX file DCBs:
cat TGD.dat p1p2.ionx|gawk '{prn=$1*1;if (length(v[prn])==0)
    {v[prn]=$2}else{d[prn]=v[prn]-$2;m=m+d[prn];n=n+1}}
        END{for (i in d) print i,d[i]-m/n}'|sort -n -k1 > d2
```

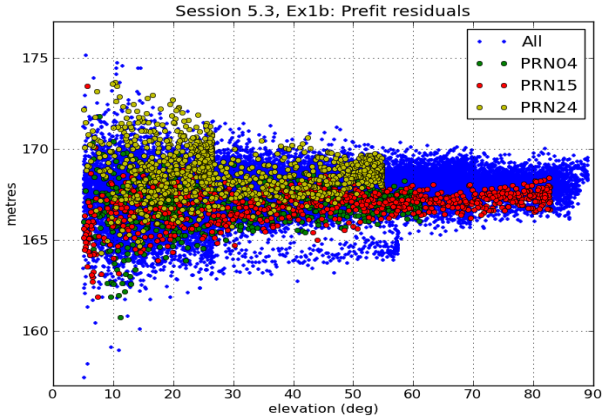
```
Plot results:
graph.py -f d1 -x1 -y2 -so- -l "BRD vs. DCB file"
           -f d2 -so- --cl r -l "BRD vs. IONEX file"
   --yn -.2 --yx 0.2 --xl "PRN" --yl "metres of L1-L2 delay"
```

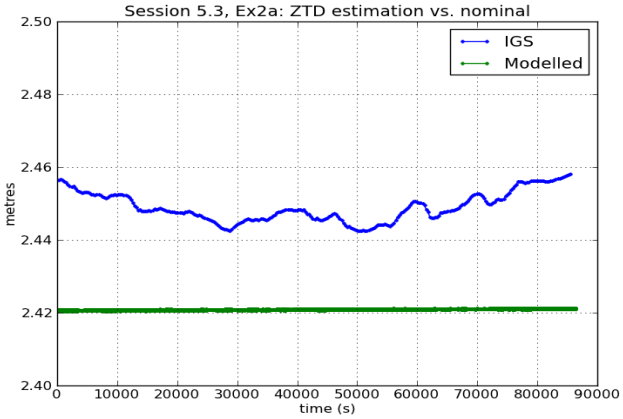
Graphs Session 5.3

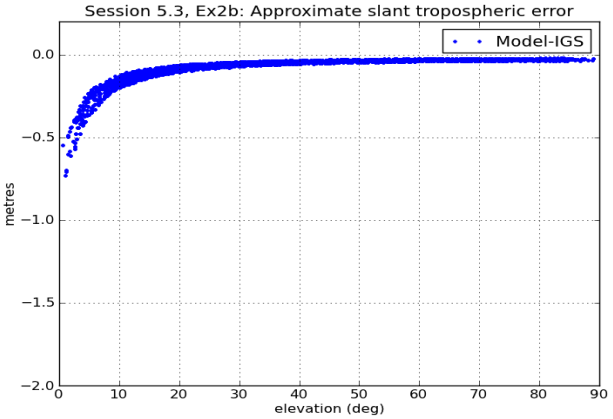


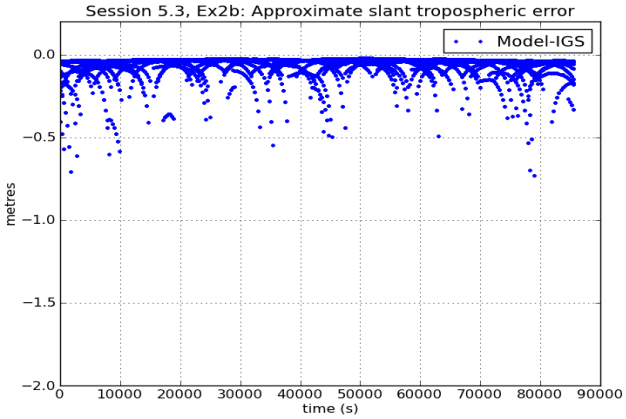


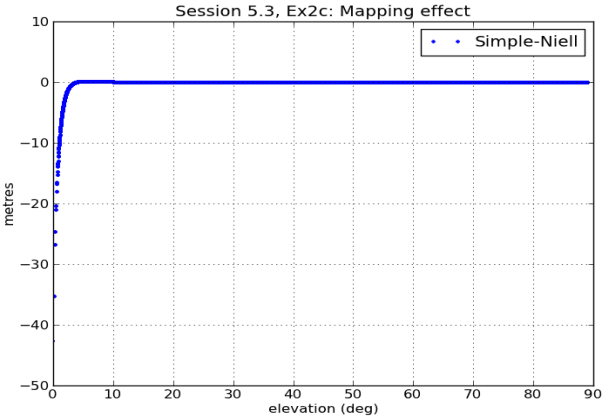


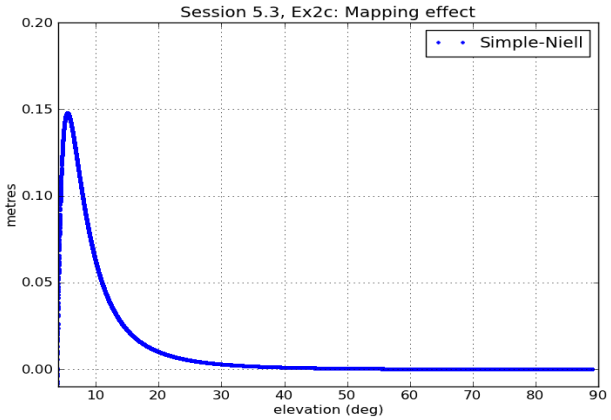


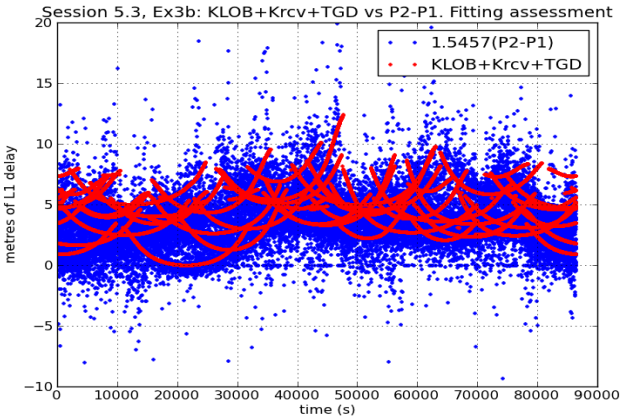


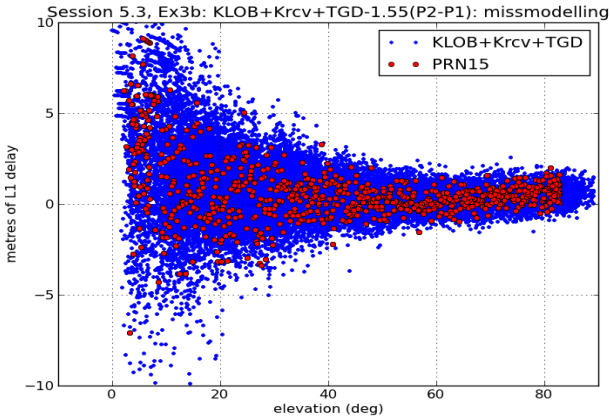


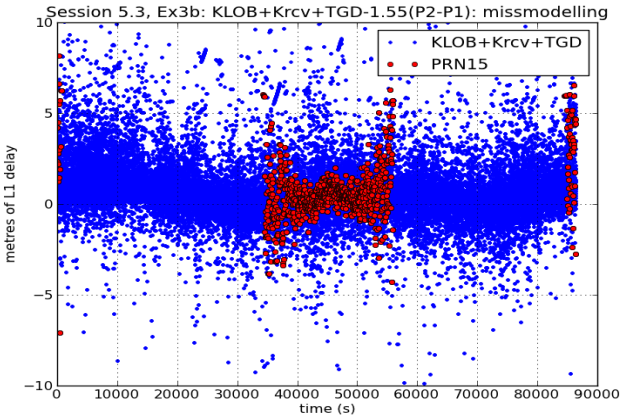


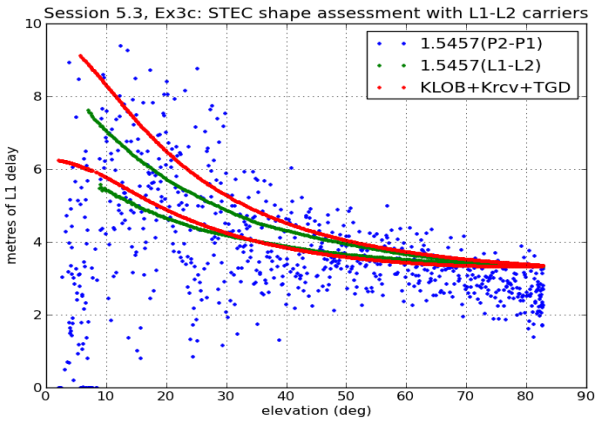


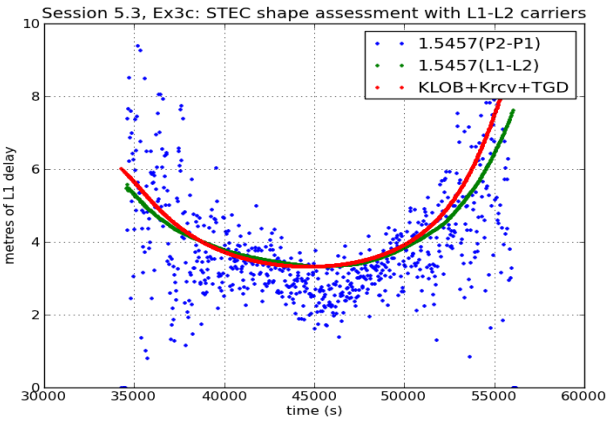

Session 5.3, Ex2c: Mapping effect
Simple-Niell
metres
elevation (deg)

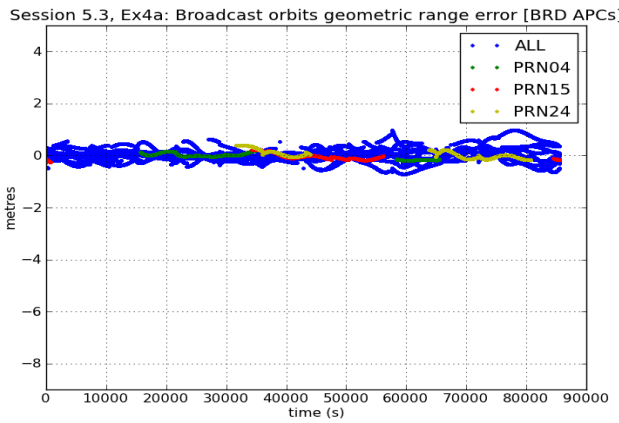

Session 5.3, Ex3b: KLOB+Krcv+TGD vs P2-P1. Fitting assessment
1.5457(P2-P1)
KLOB+Krcv+TGD
metres of L1 delay
time (s)

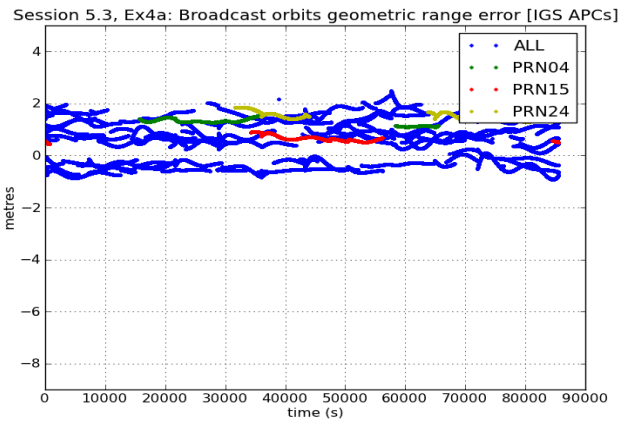

Session 5.3, Ex3b: KLOB+Krcv+TGD-1.55(P2-P1): missmodelling
KLOB+Krcv+TGD
PRN15
metres of L1 delay
elevation (deg)

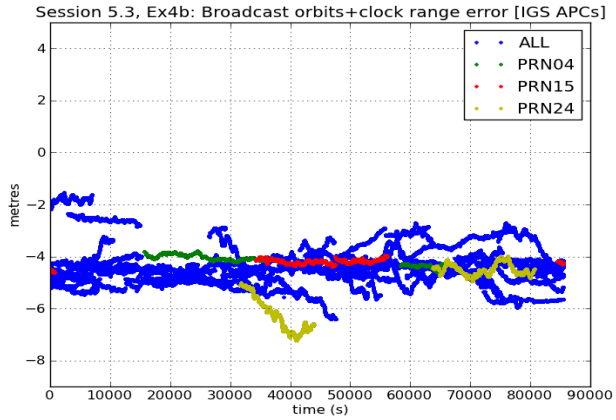

Session 5.3, Ex3b: KLOB+Krcv+TGD-1.55(P2-P1): missmodelling
KLOB+Krcv+TGD
PRN15
metres of L1 delay
time (s)

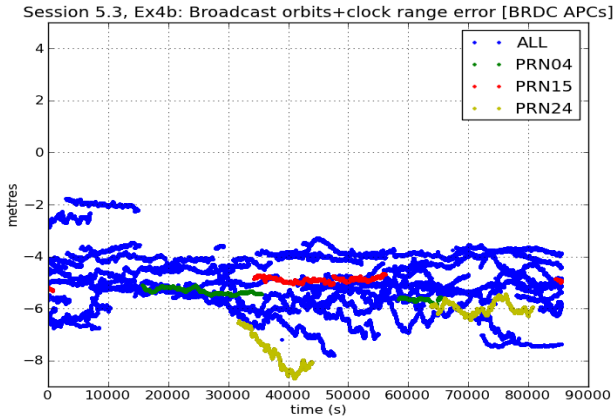

Session 5.3, Ex3c: STEC shape assessment with L1-L2 carriers
1.5457(P2-P1)
1.5457(L1-L2)
KLOB+Krcv+TGD
metres of L1 delay
elevation (deg)

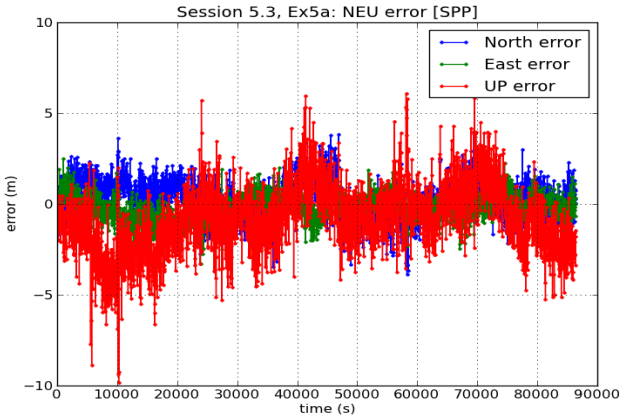

Session 5.3, Ex3c: STEC shape assessment with L1-L2 carriers
1.5457(P2-P1)
1.5457(L1-L2)
KLOB+Krcv+TGD
metres of L1 delay
time (s)

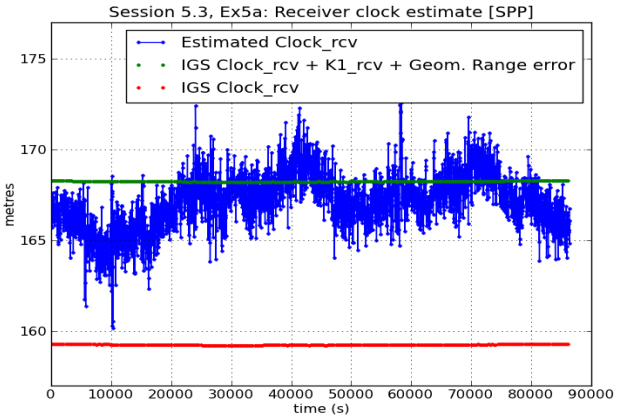

Session 5.3, Ex4a: Broadcast orbits geometric range error [BRD APCs]
ALL
PRN04
PRN15
PRN24
metres
time (s)

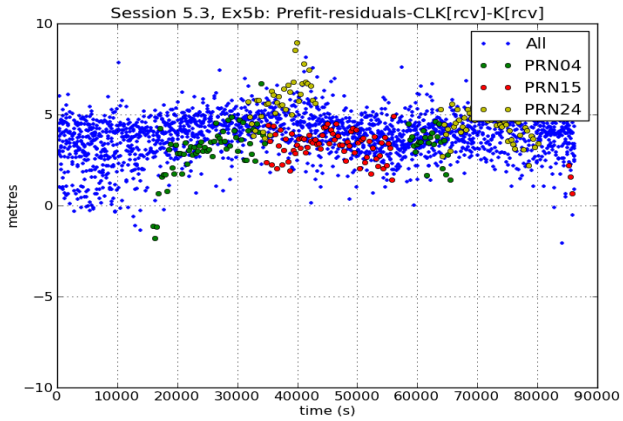

Session 5.3, Ex4a: Broadcast orbits geometric range error [IGS APCs]
ALL
PRN04
PRN15
PRN24
metres
time (s)

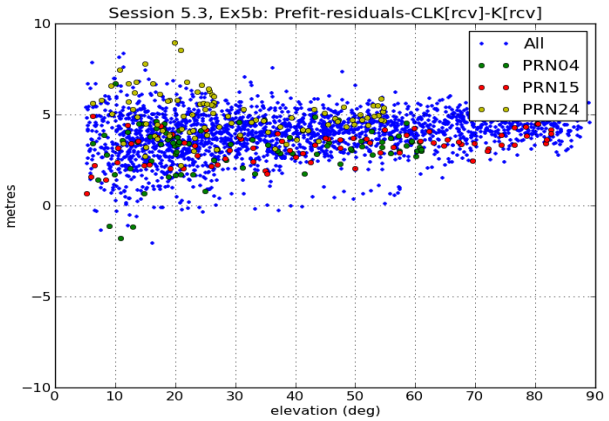

Session 5.3, Ex4b: Broadcast orbits+clock range error [IGS APCs]
ALL
PRN04
PRN15
PRN24
metres
time (s)

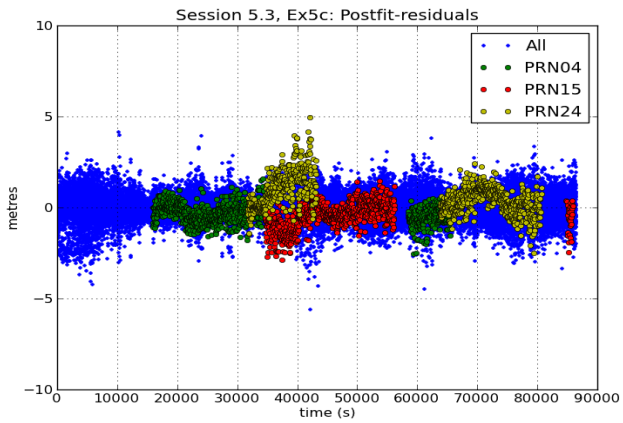

Session 5.3, Ex4b: Broadcast orbits+clock range error [BRDC APCs]
ALL
PRN04
PRN15
PRN24
metres
time (s)

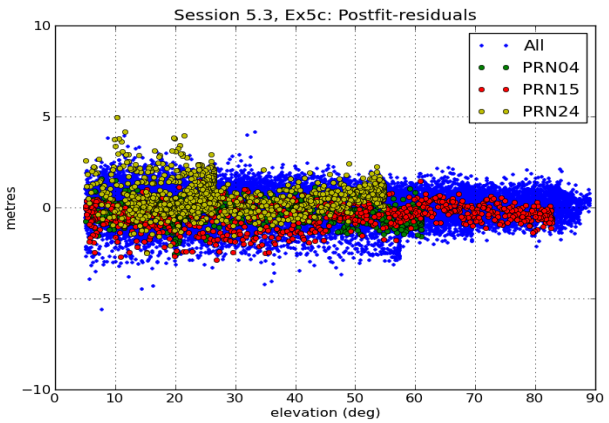

Session 5.3, Ex5a: NEU error [SPP]
North error
East error
UP error
error (m)
time (s)

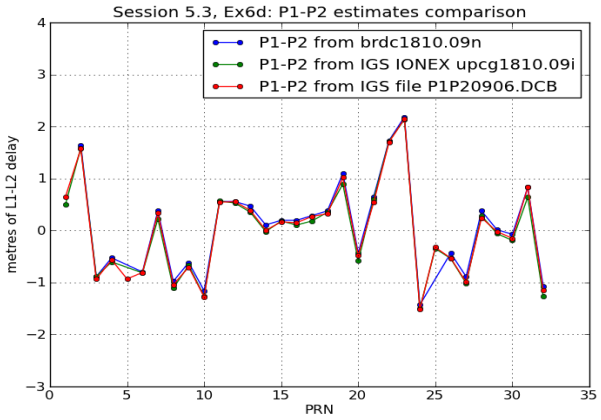

Session 5.3, Ex5a: Receiver clock estimate [SPP]
Estimated Clock_rcv
IGS Clock_rcv + K1_rcv + Geom. Range error
IGS Clock_rcv
metres
time (s)

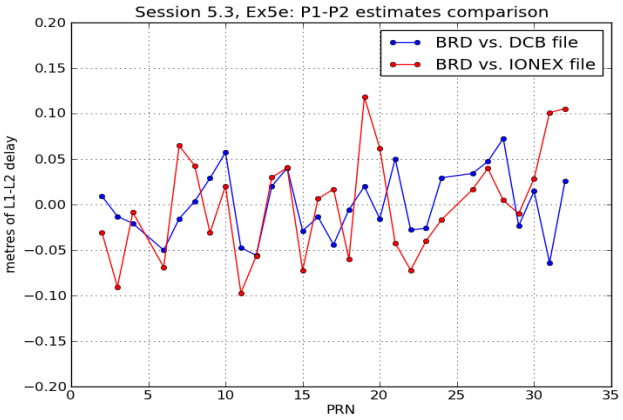

Session 5.3, Ex5b: Prefit-residuals-CLK[rcv]-K[rcv]
All
PRN04
PRN15
PRN24
metres
time (s)


Session 5.3, Ex5b: Prefit-residuals-CLK[rcv]-K[rcv]
All
PRN04
PRN15
PRN24
metres
elevation (deg)


Session 5.3, Ex5c: Postfit-residuals
All
PRN04
PRN15
PRN24
metres
time (s)


Session 5.3, Ex5c: Postfit-residuals
All
PRN04
PRN15
PRN24
metres
elevation (deg)


Session 5.3, Ex6d: P1-P2 estimates comparison
P1-P2 from brdc1810.09n
P1-P2 from IGS IONEX upcg1810.09i
P1-P2 from IGS file P1P20906.DCB
metres of L1-L2 delay
PRN


Session 5.3, Ex5e: P1-P2 estimates comparison
BRD vs. DCB file
BRD vs. IONEX file
metres of L1-L2 delay
PRN

6. Analysis of GPS SVN49 Anomaly

This last exercise is proposed as a work of synthesis, tying together the background and skills acquired on GNSS data processing in the previous sessions.

Motivation

The GPS satellite SVN49, also known as PRN01, was launched on 24 March 2009. This satellite carried an experimental demonstration payload to allow the transmission of the new GPS signal on L5, in addition to the standard L1 and L2 signals.

When the transmission of the L5 demonstration signal started, on 10 April, an anomaly producing elevation-angle-dependent pseudorange errors in the L1 signal was reported.

After a detailed analysis, the source of the problem was identified as the way in which the L5 payload was added to the satellite. The payload was integrated into a special auxiliary port (designated as J2) which caused signal reflections and produced a phase shift between the transmitted signals. This effect appeared as a permanent static multipath within the satellite.

This signal distortion has different impacts depending on the design of the user's receiver. Some receivers implementing special multipath mitigation techniques are able to filter out the satellite multipath component.

As one way to try to mitigate the effect of this problem in most of the receivers on Earth, the broadcast navigation message parameters were modified on 1 May 2009 to shift the SVN49 antenna phase centre by 150 m.

Session 6.1. Analysis of GPS SVN49 Anomaly

Objectives

As a work of synthesis of the methodology developed in the previous sessions, this exercise aims to reconstruct some of the analysis carried out by [Springer and Dilssner, 2009] on the GPS SVN49 anomaly on the L1 and L2 signals. The study is also extended to the new L5 signal.

Files to use
`l5dt1410.09o`, `brdc1410.09n`, `will1200.09o`, `will1210.09o`, `brdc1200.09n`, `brdc1210.09n`, `cod15294.eph`, `cod15295.eph`, `cod15295.clk`, `igs05_1525.atx`

Programs to use
`gLAB_linux`

Development

Session files: Copy and uncompress these session files in the working directory:

```
cp ~/GNSS/PROG/SES61/* .
cp ~/GNSS/FILES/SES61/* .
gzip -d *.gz *.Z
```

Documentation for the problem: The paper 'SVN49 and Other GPS Anomalies' by [Springer and Dilssner, 2009] provides the necessary background on the analysis of the problem. It is available on the website *http://www.insidegnss.com*. Reading this paper is the first step in the analysis of the anomaly. Further analysis can be found in [Thoelert et al., 2009].

1. Multipath analysis on the `will1200.09o` measurement file

The data file `will1200.09o` was collected by an AOA Benchmark ACT receiver with an Allen Osborne Associates Dorne Margolin Model_T antenna (AOAD/M_T). The file is in RINEX-2.11 format[1] containing GPS L1 and L2 code and carrier data. The receiver coordinates were kept fixed during the data collection at coordinates $\varphi = 52.^{\circ}2$, $\lambda = -122.^{\circ}2$.

Complete the following steps:

(a) Read the RINEX file using `gLAB` and generate the `MEAS` file with the measurements given in columns. The broadcast navigation file `brdc1200.09n` can be used for computing the satellite coordinates.

[1] This file was downloaded from the Crustal Dynamics Data Information System (CDDIS) server: *ftp://cddis.gsfc.nasa.gov/pub/gps/data/daily/2009/120/09o/*.

(b) Produce a skyplot to view the PRN01 satellite track and show that PRN30 is the closest satellite to PRN01.

(c) Produce a plot to analyse the multipath on C1, P1 and P2 codes for PRN01. Add the satellite's elevation to the plots.

i. Are the C1 and P1 codes affected in a similar way? And the P2 code?

ii. Repeat the plots for satellite PRN30 and compare results.

iii. Discuss if the observed patterns for PRN01 can be related to L1 and/or L2 antenna phase centre biases (see exercise 2).[2]

(d) Plot the Melbourne–Wübbena combination of the L1 and L2 signals for both satellites (PRN01 and PRN30) and compare the results with the previous ones.

(e) Plot the difference in the code and carrier ionosphere-free combinations (PC–LC) and compare the results with those for the C1 and P1 codes.

Discuss if the observed patterns in PC–LC for PRN01 can be related to L1 and/or L2 antenna phase centre biases (see exercise 2b).

(f) Justify the different amplitudes of the elevation-dependent patterns seen in satellite PRN01 with the different combinations.

2. Receiver APC offset effect

Let $(0, 0, \Delta_{1_{UP}})$ and $(0, 0, \Delta_{2_{UP}})$ be the Antenna Phase Centre (APC) offsets of the L1 and L2 signals (in ENU coordinates) relative to the Antenna Reference Point (ARP).

Show that the relative bias $\Delta_{1_{UP}} - \Delta_{2_{UP}}$ between the L1 and L2 APC signals produces an elevation-dependent effect $\Delta\rho$ on the combination

$$\mathcal{M}_{C_1} = C_1 - L_1 - \frac{2}{\gamma_{12} - 1}(L_1 - L_2) \tag{6.1}$$

according to the equation

$$\Delta\rho = \frac{2}{\gamma_{12} - 1}(\Delta_{1_{UP}} - \Delta_{2_{UP}}) \sin\varepsilon \tag{6.2}$$

where $\gamma_{12} = (f_1/f_2)^2 = (77/60)^2$.

Hint: From Fig. 6.1, it follows that $\rho_1 = \rho - \Delta_{1_{UP}} \sin\varepsilon$ *and* $\rho_2 = \rho - \Delta_{2_{UP}} \sin\varepsilon$. *Thus, substituting these in equation (6.1) leads to expression (6.2).*

[2] The L1 and L2 antenna phase centres for the (AOAD/M_T) antenna can be found in the ANTEX file `igs05_1525.atx`. The values are (North/East/Up): L1=[0.60,−0.46,91.24], L2=[−0.10,−0.62,120.06], expressed in millimetres.

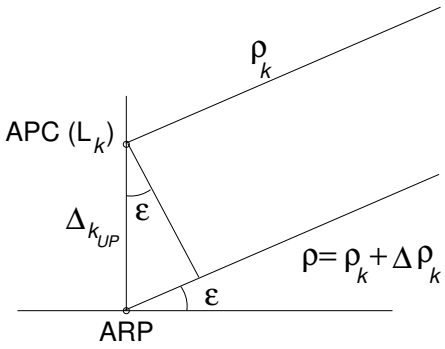


Figure 6.1: Layout associated with equation (6.1). The projection in range of APC $\Delta_{k_{UP}}$ of the L_k signal ($k = 1, 2$) is given by $\Delta\rho_k = \Delta_{k_{UP}} \sin(\varepsilon)$.

(a) Show that the next expression applies for the Melbourne–Wübbena combination:

$$\Delta\rho = -\frac{2\sqrt{\gamma_{12}}}{\gamma_{12}-1}(\Delta_{1_{UP}} - \Delta_{2_{UP}})\sin\varepsilon \tag{6.3}$$

Hint: Take $\rho_1 = \rho - \Delta_{1_{UP}}\sin\varepsilon$ *and* $\rho_2 = \rho - \Delta_{2_{UP}}\sin\varepsilon$ *in the Melbourne–Wübbena combination.*

(b) Justify that a receiver phase centre offset $(\Delta_N, \Delta_E, \Delta_{UP})$ between L1 and L2 signals will not produce any elevation-dependent effect on the difference of the code and carrier ionosphere-free combinations (i.e. PC–LC).

Hint: Note that code PC and carrier LC have the same APC. This does not happen with the Melbourne–Wübbena combination LW–PN, where code and carrier are multiplied by different factors.

(c) Taking into account that $\Delta_{1_{UP}} - \Delta_{2_{UP}} \simeq -3\,\text{cm}$ for the AOAD/M_T antenna used in the data collection, evaluate the effect of this offset in the patterns seen previously.

(d) Show that similar expressions to (6.1) and (6.3) can be applied for the satellite, changing the elevation ε by $90° - \eta$, where η is the deviation angle from a direction pointing to Earth's centre (see Fig. 6.2).

(e) Show that the effect on range produced by a given satellite APC is so much smaller than the effect produced by a receiver APC.
Hint: For a GPS satellite $|\Delta\eta| \leq \arcsin(6\,400/26\,560) < 14°$*, while in a ground receiver* $\varepsilon \leq 90°$*.*

3. Satellite multipath effect

In the previous exercise it was shown that an APC offset cannot produce an elevation-dependent effect on the LC–PC combination, because LC and PC share the same APC.[3]

In order to explain the previously seen elevation-dependent effect on the combination LC–PC, it is proposed to analyse the effect of a large satellite multipath producing a stable bias on code over time. In this context, it will be assumed that the effect of multipath on the carrier is negligible

[3]The same APCs are usually taken in the satellite for the L1 and L2 signals (see, for instance, the ANTEX file `igs05_1525.atx`).

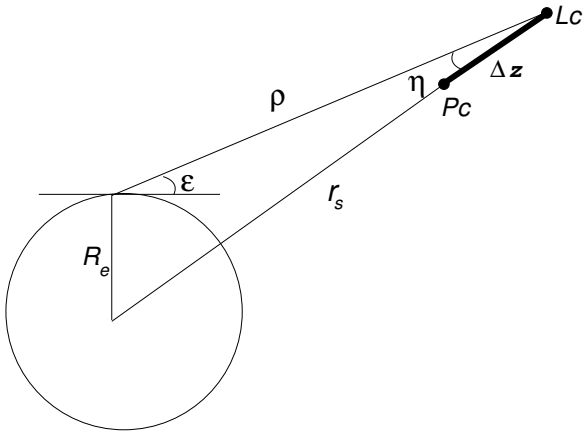


Figure 6.2: Layout associated with equation (6.4). A large Δz bias between the *Pc* and *Lc* APCs produces an elevation-angle-dependent effect on range $\Delta\rho(\varepsilon)$.

(see exercise 4) with respect to the code, which reaches up to 150 m, see [Springer and Dilssner, 2009].[4]

The following questions are posed:

(a) Show that a large offset Δz on code PC (but not on carrier LC) would produce an elevation-angle-dependent effect $\Delta\rho$ on the PC–LC combination, according to the equation

$$\Delta\rho = \Delta z \left[1 - \sqrt{1 - \left(\frac{R_e}{r_s}\cos\varepsilon\right)^2}\right] \tag{6.4}$$

where R_e is the radius of Earth and r_s the distance of the satellite from Earth's centre.

Hint: From Fig. 6.2 it follows that $\Delta\rho = \Delta z\,(1 - \cos\eta)$*, where the* nadir angle η *is related to the elevation* ε *by* $\eta = \arcsin\left((R_e/r_s)\cos\varepsilon\right)$*. Thus, taking into account that* $\cos(\arcsin x) = \sqrt{1 - x^2}$ *leads to equation (6.4).*

(b) From equation (6.4), show that an offset Δz of about 150 m will produce an elevation-dependent effect of up to 4.5 m on the pseudorange with a quite similar pattern to that seen in the previous exercise.
Note: The following values can be taken for the numerical application: $\Delta z = 150\,m$*,* $R_e = 6378\,km$ *and* $r_s = 26\,600\,km$*.*

(c) Produce a plot overlapping the code and carrier differences in the ionosphere-free combination (PC–LC) and the curve defined by equation (6.4). Use the previous values for Δz, R_e and r_s.

(d) Repeat the previous plot, but for the P1 code.

(e) Discuss the previous results.

4. Position domain analysis of the `will1200.09o` file

(a) Compute the Standard Point Positioning solution for the measurement file `will1200.09o` using the broadcast navigation message file `brdc1200.09n`.

[4] Note that the multipath effect on the carrier is less than a quarter of the signal wavelength, while in the code it can reach a theoretical value of 1.5 times the 'chip length', see section 4.2.2 in Volume I.

Hint: Starting from the default configuration of `gLAB` *in* `SPP` *mode, uncheck option* `[☐ Discard unhealthy satellites]`*, set* `Data decimation` *to* 30 *seconds and uncheck satellite PRN16*[5] *in section* `[Preprocess]`*. Uncheck option* `[☐ P1-C1 correction]` *in section* `[Modelling]`*, to be sure that the C1 code is always used (and not the P1 one).*

i. Plot the NEU positioning error as a function of time.
ii. Plot the postfit residuals as a function of elevation, emphasising the PRN01 residuals. As in the previous case, overlap on the same plot the curve defined by equation (6.4).
iii. Plot the postfit residuals as a function of time.

(b) Repeat the previous study but positioning with the ionosphere-free combination of codes (PC).

Note: In section `[Preprocess]`*, set the* `[☒ P1-C1 correction]` *option again to this computation.*

(c) Repeat the study but positioning in kinematic PPP mode (i.e. with the ionosphere-free combination of codes and carrier measurements (PC, LC)). The precise orbit and clock files `cod15294.eph` and `cod15294.clk`, and the ANTEX file `igs05_1525.atx`, can be used for this processing. Draw different plots to depict the code PC and carrier LC postfit residuals.

i. What is the magnitude of the multipath effect on the PC code?
ii. Are the carrier phase measurements affected in the same way as the code?

(d) Discuss the suitability of a solution to compensate the problem of 4 m of multipath, based on using an 'artificial' APC offset of 150 m for this satellite in the GPS broadcast navigation message.

The mitigation effect would be the same for PC and C1 code-based positioning (see exercise 6)?

5. SVN49 (PRN01) navigation message analysis

On 1 May 2009, the SVN49 (PRN01) navigation message was modified by placing the APC about 150 m above the satellite (rather than slightly below as usual). The satellite clock bias was also updated by a similar amount to ensure that the user range error was almost unaffected by this change.

This ephemeris update can be depicted as follows:

(a) *30 April 2009: Satellite coordinates and clock comparison*
Compare the broadcast satellite coordinates and clock offset computed from the navigation message file `brdc1200.09n` with the IGS precise coordinates of file `cod15294.eph`. Uncheck the broadcast satellite healthy check of `gLAB` by setting `--model:satellitehealh`.
Note: Exclude satellite PRN16, which showed anomalous behaviour with a large coordinate error due to repositioning manoeuvres on 30 April (this satellite is set as unhealthy in the broadcast message).

[5] This satellite is set as unhealthy in the GPS broadcast message of 30 April, due to repositioning manoeuvres.

(b) *1 May 2009: Satellite coordinates and clock comparison*
Repeat the previous comparison using the navigation message file `brdc1210.09n` and the IGS precise coordinates of file `cod15295.eph`, which corresponds to the 1 May 2009.

6. User domain effect of the APC update on SVN49

The data file `will1200.09o` processed in exercise 4 corresponds to 30 April, that is before the navigation message update. Repeat the exercise now, but processing with the data files of 1 May 2009. That is, use the RINEX measurement file `will1210.09o`, the broadcast orbit and clock file `brdc1210.09n` and the precise orbit and clock files `cod15295.eph` and `cod15295.clk`.

(a) Read the RINEX file using `gLAB` and generate the `MEAS` file with the measurements given in columns. The broadcast navigation file `brdc1210.09n` can be used for computing the satellite coordinates.

(b) Produce a skyplot to view the PRN01 and PRN30 satellite tracks.

(c) Repeat the plot of exercise 1 to depict the multipath in the C1 and C2 codes, in the Melbourne–Wübbena combination and in the ionosphere-free combination. Verify the stationarity of the satellite multipath, by overlapping, for instance, the Melbourne–Wübbena combination plots of 30 April and 1 May.

(d) Compute the positioning error with the ionosphere-free combination of codes PC. Compare the error found for both days and evaluate the impact of the adopted solution in the positioning domain. Compare also the postfit residuals for both days.

(e) Compare the results computed with LC, PC using the IGS precise products. Why are no changes found using precise products instead of broadcast ones?

(f) Compute the positioning error with the C1 code. Compare the error found for both days and evaluate the impact of the adopted solution in the positioning domain. Compare also the postfit residuals for both days.

(g) To confirm again the stationarity of this signal anomaly, generate a plot overlapping the C1 code multipath for both days (shift the second-day plot by $3^{m}56^{s}$ to account for the delay in sidereal time with respect to solar time in 24 h). Repeat the plot for the ionosphere-free combination PC multipath and for the Melbourne–Wübbena combination.

(h) Discuss the pros and cons of the adopted solution by the GPS control segment (i.e. modification of the broadcast message by adding 150 m of APC to satellite PRN01). For instance, does it work in the same way for all receivers? Do single-frequency users experience similar benefits to dual-frequency users?

Note: Reading document 'GPS IIR-20 (SVN-49) Panel Discussion' [Goldstein, 2010] is recommended. It contains a discussion among the GPS community of this signal problem.

7. Anomaly analysis of a three-frequency measurement file

The data file `l5dt1410.09o` was collected by a Trimble NETR8 receiver with a TRM59800.00 antenna. This receiver was located at coordinates $\varphi = 40^{\circ}.1$, $\lambda = -105^{\circ}.2$, which were kept fixed during the data collection.

This data file[6] follows the standard RINEX-2.11 format containing GPS L1, L2 and L5 code and carrier data for satellite PRN01 and L1 and L2 measurements for the remaining satellites.

Complete the following steps:

(a) Read the RINEX file using `gLAB` and generate the `MEAS` file with the measurements given in columns. Use the orbits file `brdc1410.09n`.

(b) Produce a skyplot to view the PRN01, PRN02 and PRN30 satellite tracks, among the others.

(c) Produce plots to depict the multipath for the three ionosphere-free combinations of codes for PRN01: $PC_{[12]} - LC_{[12]}$, $PC_{[15]} - LC_{[15]}$ and $PC_{[25]} - LC_{[25]}$. Plot the $PC_{[12]} - LC_{[12]}$ one for satellite PRN30 and compare the results with those of PRN01.

i. Are the patterns seen in the plots affected by any APC offset?

Note: Compare the maximum satellite elevation with that seen in the previous exercise with file `will1200.09o`.

ii. In which combinations does an elevation-dependent pattern appear?

iii. Do the results suggest that the satellite multipath affects only the P1 code?

(d) In order to analyse the effect of this signal anomaly on carrier phases, plot the *ionosphere-free and geometry-free* combinations of carriers (see session 4.3). What delay units are being considered in the plot?

i. A clear elevation-dependent pattern appears in the plot. Discuss possible sources of this pattern. What is the most plausible hypothesis: carrier phase multipath or APC biases on L1 and/or L2 and/or L5?

ii. Taking into account that the L1 and L2 APC biases are given, respectively (in NEU coordinates), by L1=$[0.37, 0.86, 90.02]$ and L2=$[0.09, 0.01, 119.89]$, in millimetres,[7] and considering that L5 is unknown (and assumed quite similar to the L1 and L2 APCs), discuss if the pattern seen in the figure could be associated with these APC biases.

iii. Assuming only APC bias in the vertical (Up) component for the L1, L2 and L5 APCs, estimate an APC offset in the L5 signal that could explain the observed elevation-angle-dependent effect.

Solution: $L5 = [0, 0, 274.74]\,mm$.

Note that Octave or MATLAB can be used as well to solve the Least Squares fitting.

[6] This file was gathered from the CDDIS server: *ftp://cddis.gsfc.nasa.gov/pub/gps/data/l5test/rinex2/daily/2009/141/*.

[7] These values are given in the ANTEX file `igs05_1525.atx` for the antenna 'TRM59800.00 NONE' referred to in the header of the RINEX file `l5dt1410.09o`.

iv. Repeat the previous plot, but removing all APC patterns. Discuss what might be the source of the residual pattern seen in the plot after removing such APC effects.

(e) Produce different plots to analyse the Melbourne–Wübbena combinations for the three possible pairs of signals: [12], [15] and [25].

Why does a stronger elevation-dependent pattern appear in the [25] combination of signals than in the [12] and [15], while, according to the previous results, the satellite multipath only affects the C1 code?

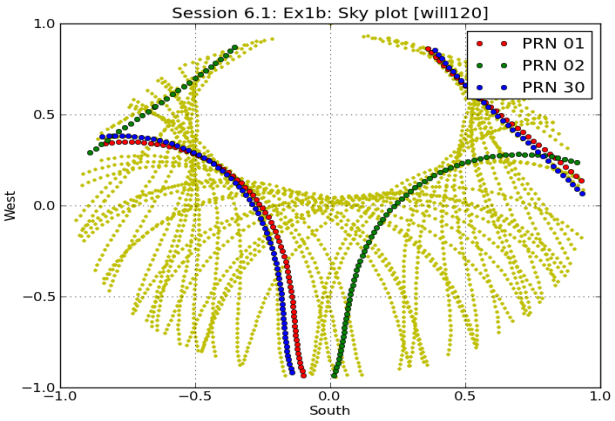
Hint: Consider the different amplification factors affecting the combinations and explore the APC as a possible source of this pattern.

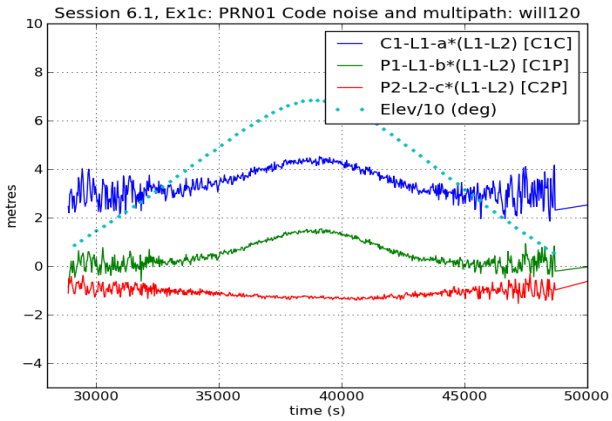
i. Taking into account the expression (6.3), compute the factors $2\sqrt{\gamma_{ij}}/(\gamma_{ij}-1)$ (where i,j indicate the pairs of frequencies involved, i.e., $i,j \in \{1,2,5\}$) multiplying the APC biases for the different combinations.

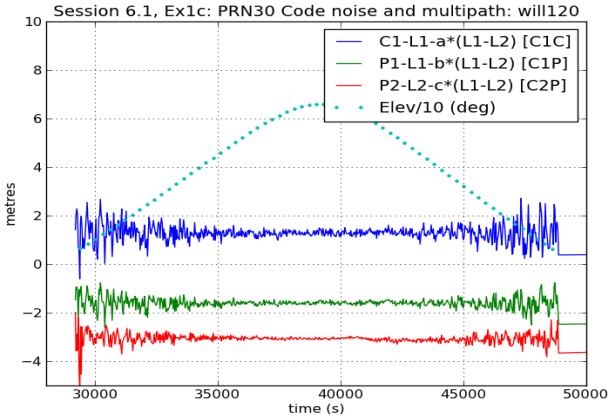
ii. Applying the antenna L1, L2 and L5 phase centre values of the previous exercise to expression (6.3), compute the APC effect on the Melbourne–Wübbena combinations and remove this effect from the plot.

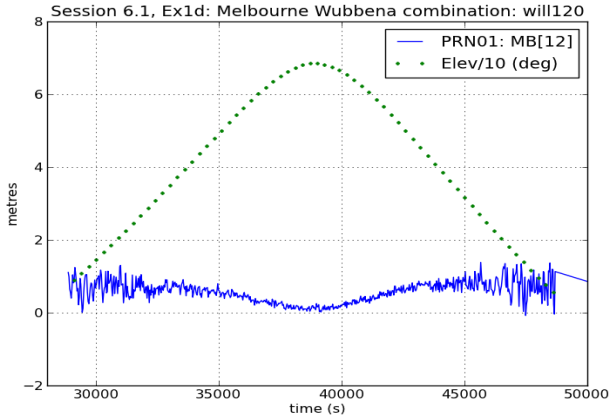
iii. Has the pattern of combination [25] been mitigated?

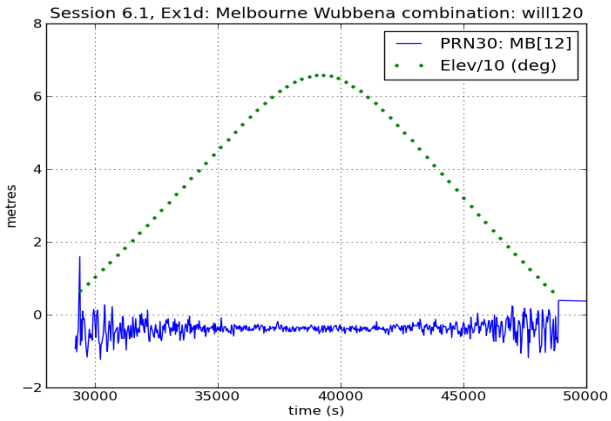
Graphs Session 6.1

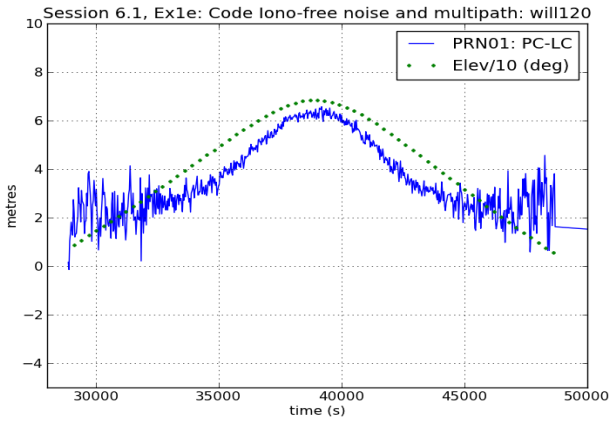


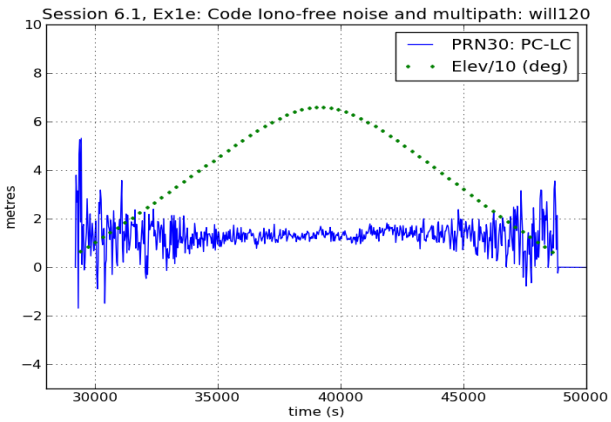


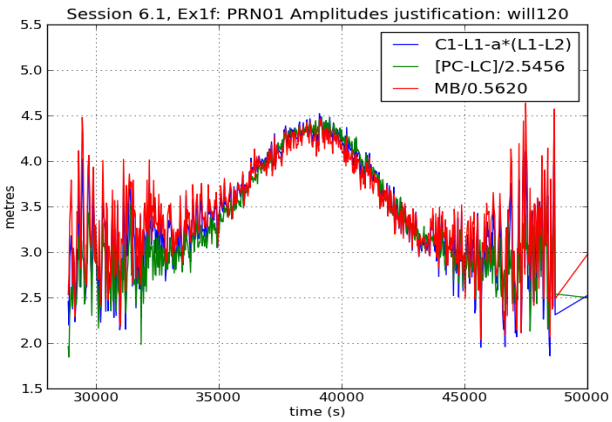


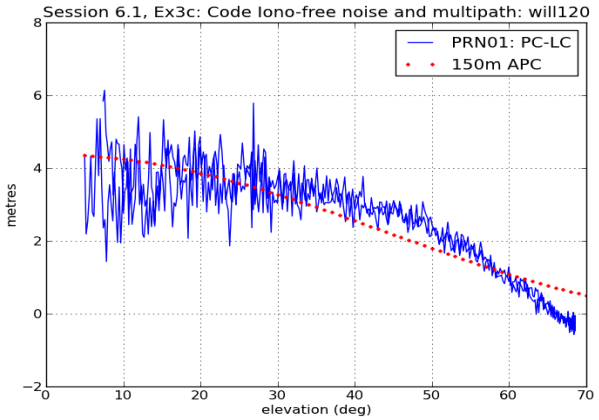


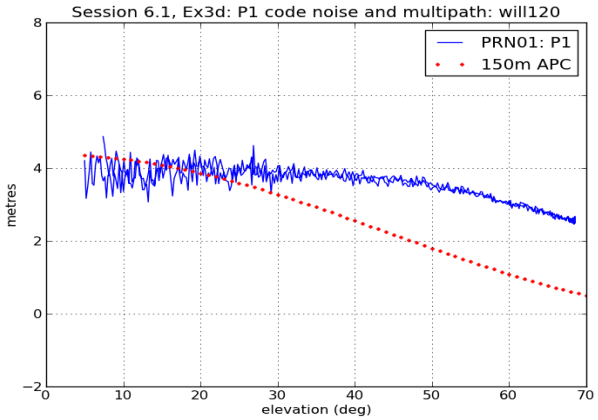


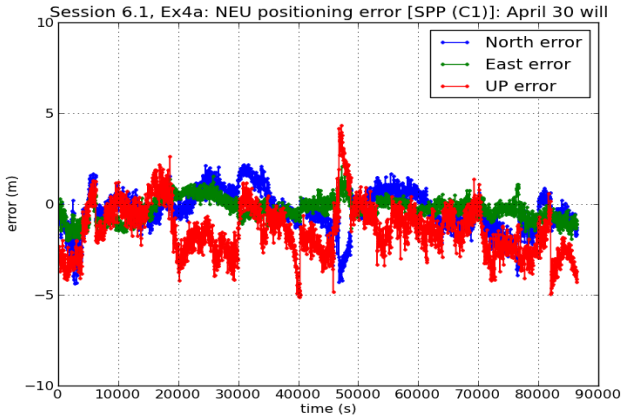


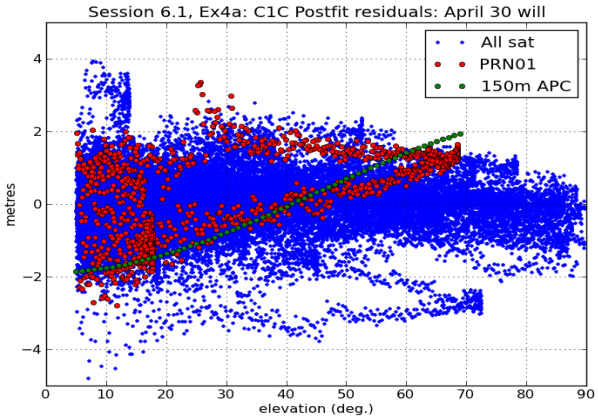


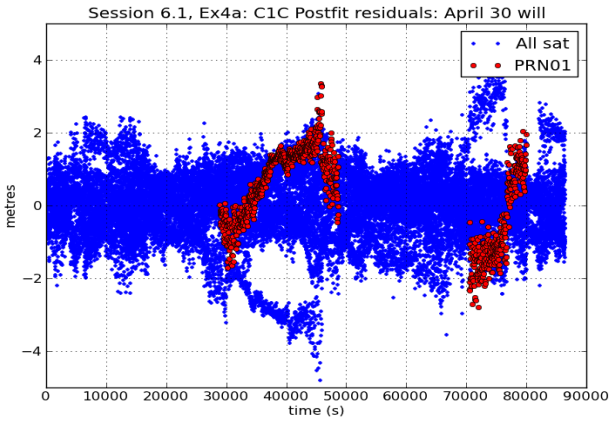


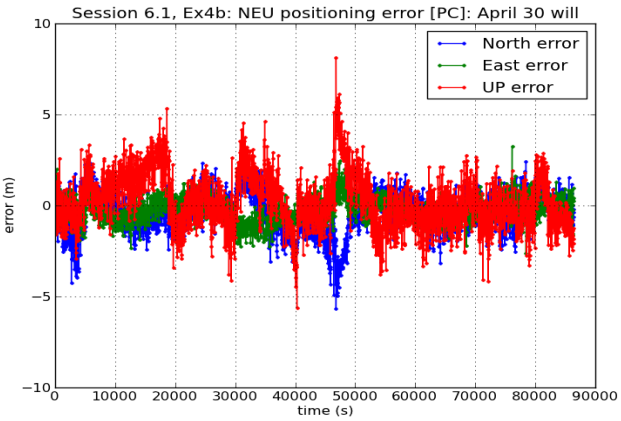

Session 6.1, Ex3c: Code Iono-free noise and multipath: will120
PRN01: PC-LC
150m APC
metres
elevation (deg)

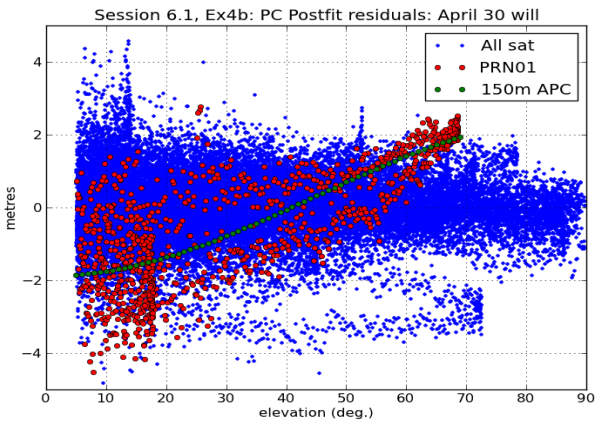

Session 6.1, Ex3d: P1 code noise and multipath: will120
PRN01: P1
150m APC
metres
elevation (deg)

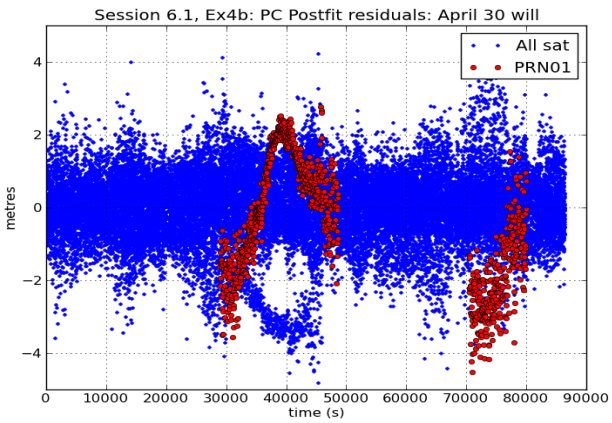

Session 6.1, Ex4a: NEU positioning error [SPP (C1)]: April 30 will
North error
East error
UP error
error (m)
time (s)

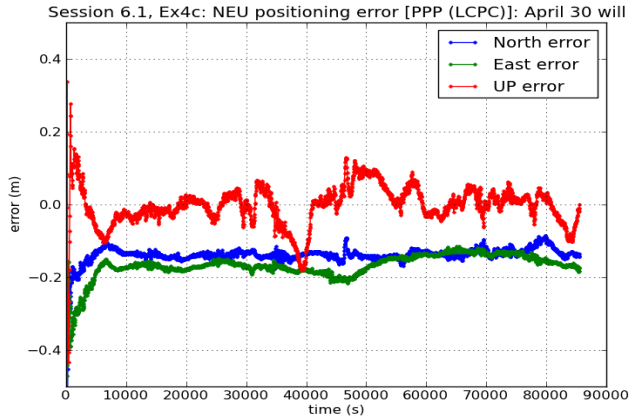

Session 6.1, Ex4a: C1C Postfit residuals: April 30 will
All sat
PRN01
150m APC
metres
elevation (deg.)

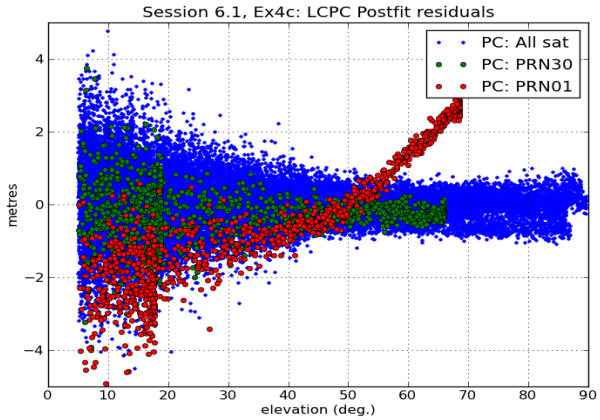

Session 6.1, Ex4a: C1C Postfit residuals: April 30 will
All sat
PRN01
metres
time (s)

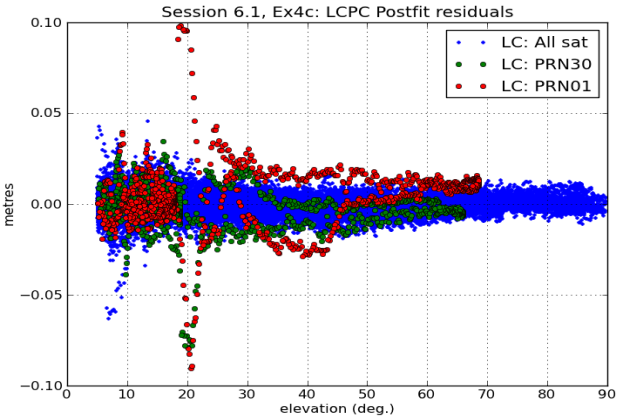

Session 6.1, Ex4b: NEU positioning error [PC]: April 30 will
North error
East error
UP error
error (m)
time (s)

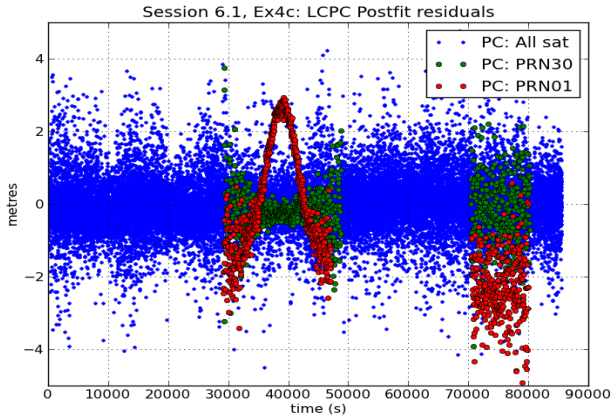

Session 6.1, Ex4b: PC Postfit residuals: April 30 will
All sat
PRN01
150m APC
metres
elevation (deg.)

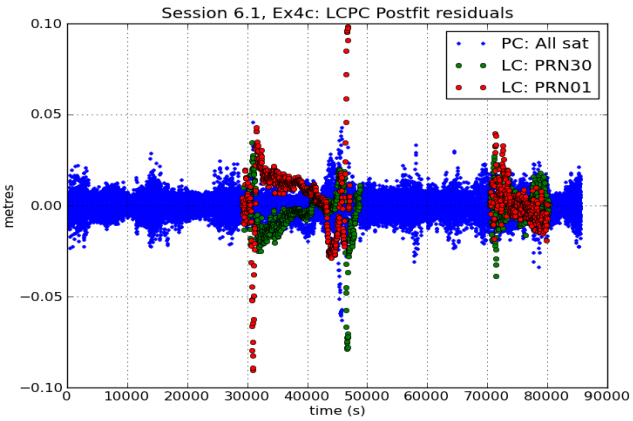

Session 6.1, Ex4b: PC Postfit residuals: April 30 will
All sat
PRN01
metres
time (s)

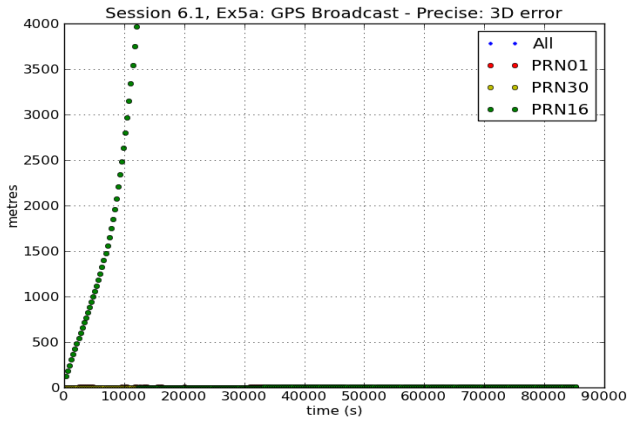

Session 6.1, Ex4c: NEU positioning error [PPP (LCPC)]: April 30 will
North error
East error
UP error
error (m)
time (s)

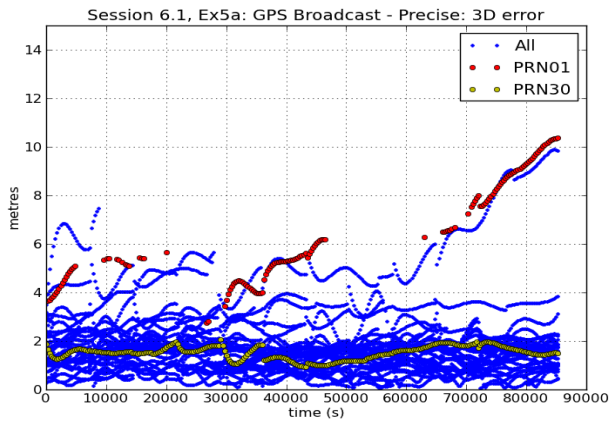

Session 6.1, Ex4c: LCPC Postfit residuals
PC: All sat
PC: PRN30
PC: PRN01
metres
elevation (deg.)

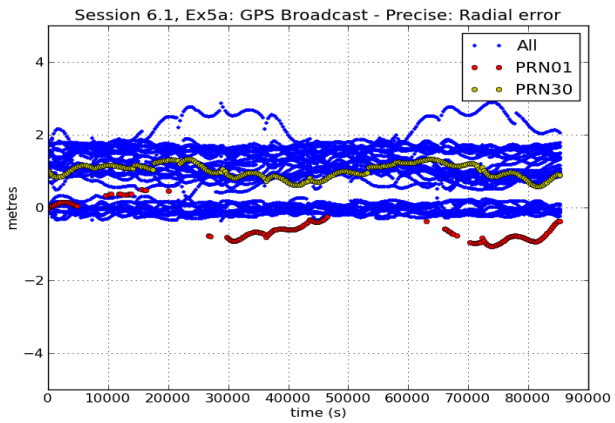

Session 6.1, Ex4c: LCPC Postfit residuals
LC: All sat
LC: PRN30
LC: PRN01
metres
elevation (deg.)

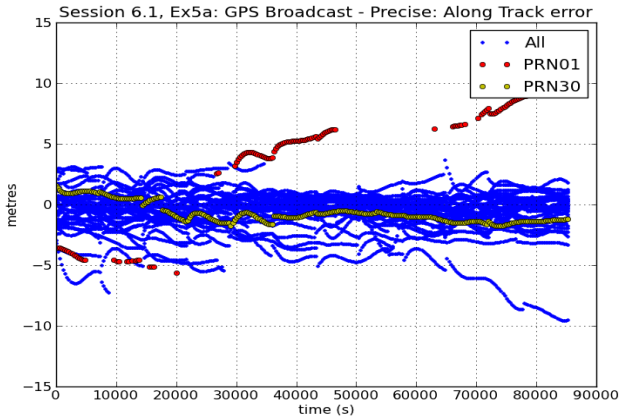

Session 6.1, Ex4c: LCPC Postfit residuals
PC: All sat
PC: PRN30
PC: PRN01
metres
time (s)

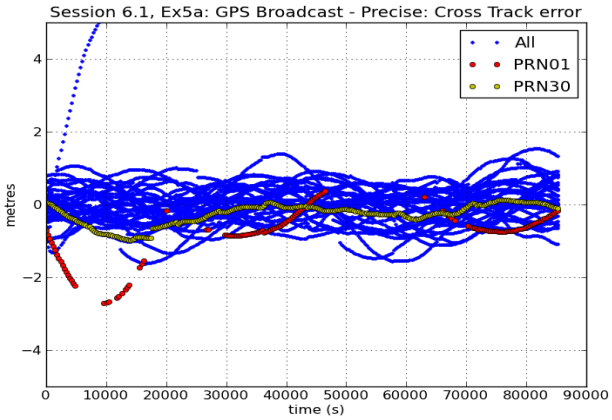

Session 6.1, Ex4c: LCPC Postfit residuals
PC: All sat
LC: PRN30
LC: PRN01
metres
time (s)

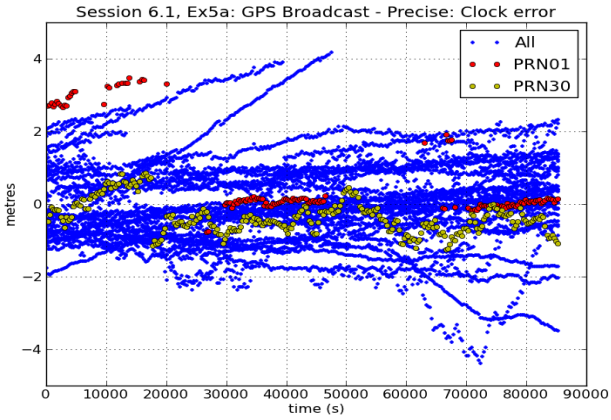

Session 6.1, Ex5a: GPS Broadcast - Precise: 3D error
All
PRN01
PRN30
PRN16
metres
time (s)

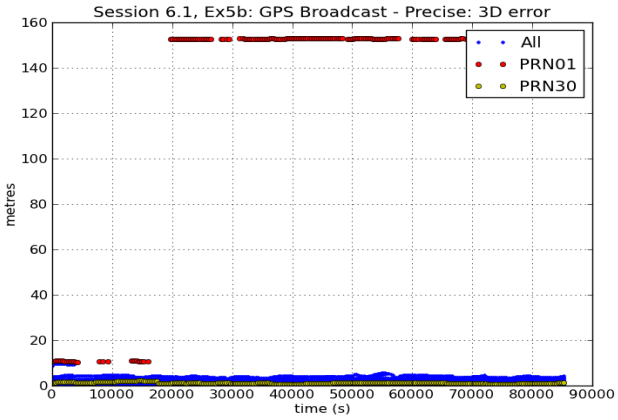

Session 6.1, Ex5a: GPS Broadcast - Precise: 3D error
All
PRN01
PRN30
metres
time (s)

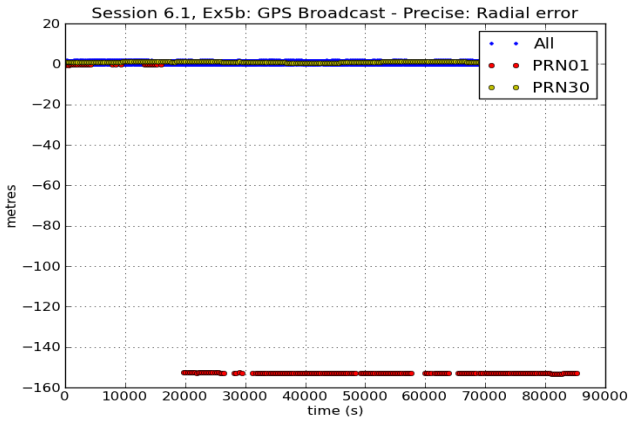

Session 6.1, Ex5a: GPS Broadcast - Precise: Radial error
All
PRN01
PRN30
metres
time (s)

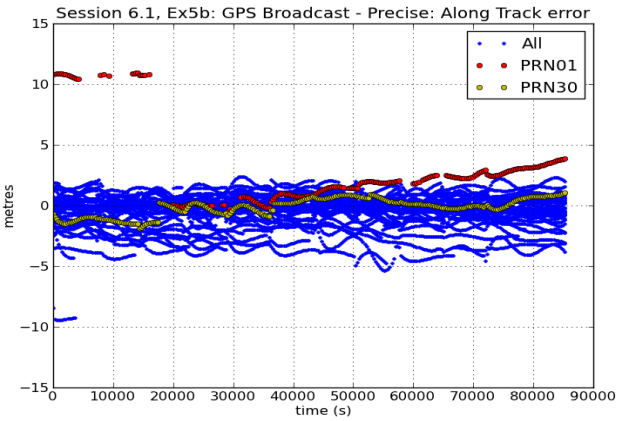

Session 6.1, Ex5a: GPS Broadcast - Precise: Along Track error
All
PRN01
PRN30
metres
time (s)

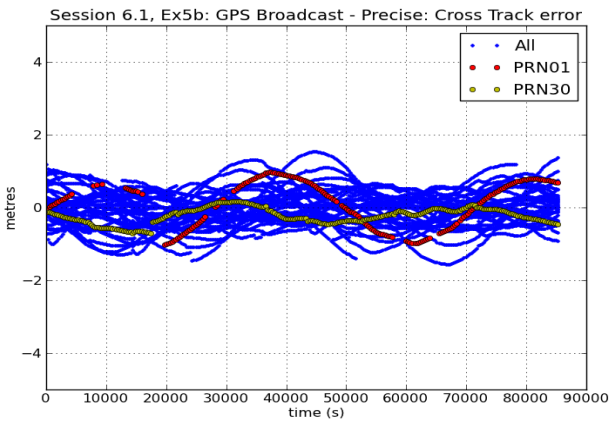

Session 6.1, Ex5a: GPS Broadcast - Precise: Cross Track error
All
PRN01
PRN30
metres
time (s)

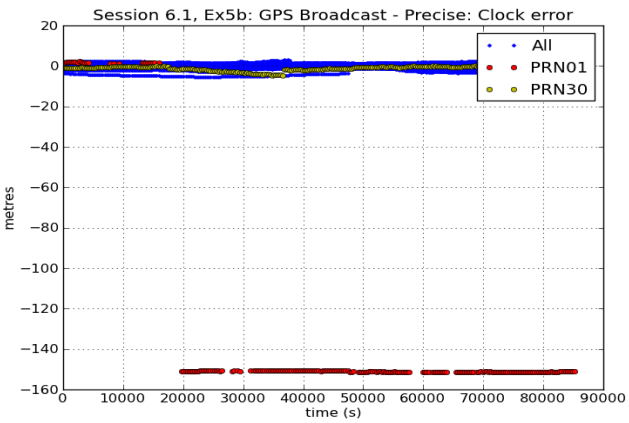

Session 6.1, Ex5a: GPS Broadcast - Precise: Clock error
All
PRN01
PRN30
metres
time (s)

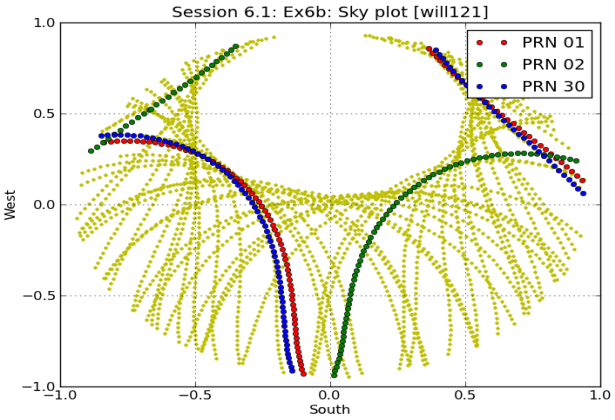

Session 6.1, Ex5b: GPS Broadcast - Precise: 3D error
All
PRN01
PRN30
metres
time (s)

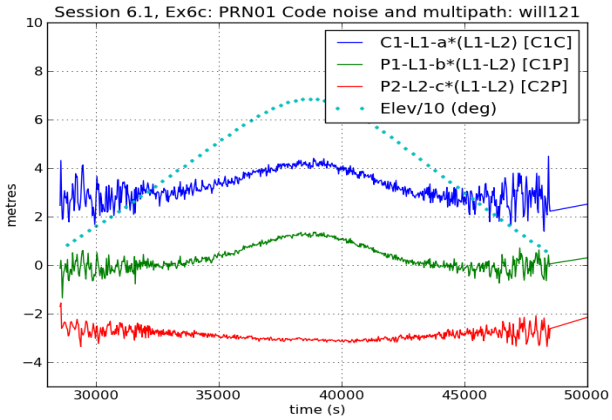

Session 6.1, Ex5b: GPS Broadcast - Precise: Radial error
All
PRN01
PRN30
metres
time (s)

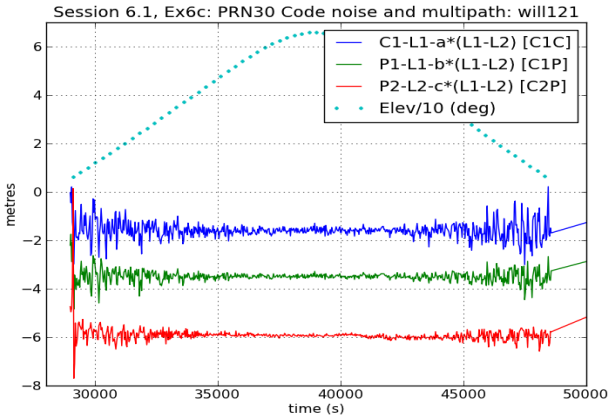

Session 6.1, Ex5b: GPS Broadcast - Precise: Along Track error
All
PRN01
PRN30
metres
time (s)

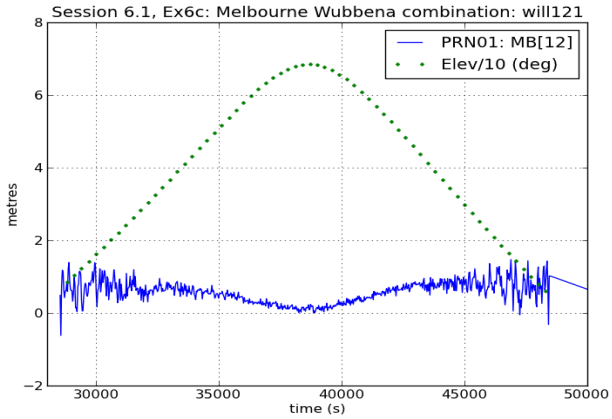

Session 6.1, Ex5b: GPS Broadcast - Precise: Cross Track error
All
PRN01
PRN30
metres
time (s)

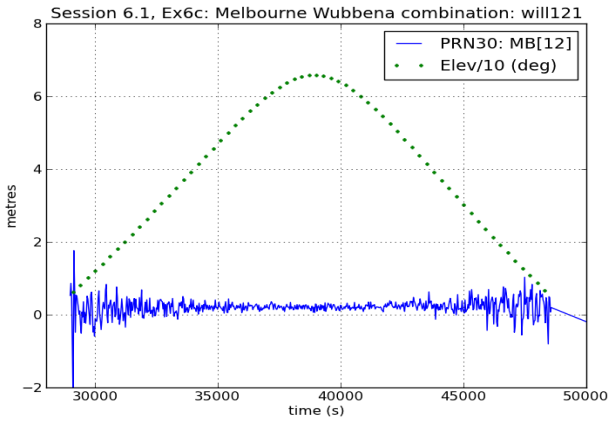

Session 6.1, Ex5b: GPS Broadcast - Precise: Clock error
All
PRN01
PRN30
metres
time (s)

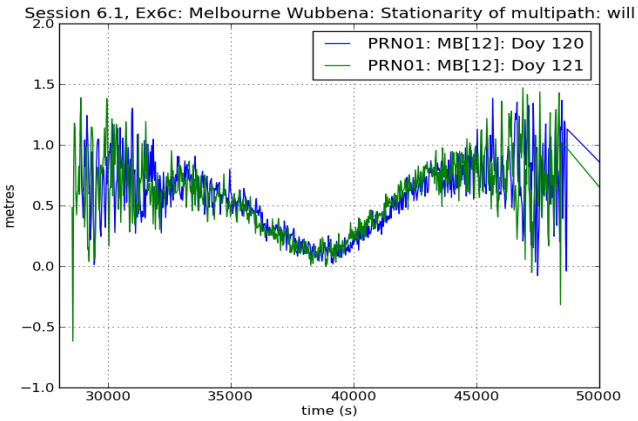

Session 6.1: Ex6b: Sky plot [will121]
PRN 01
PRN 02
PRN 30
West
South

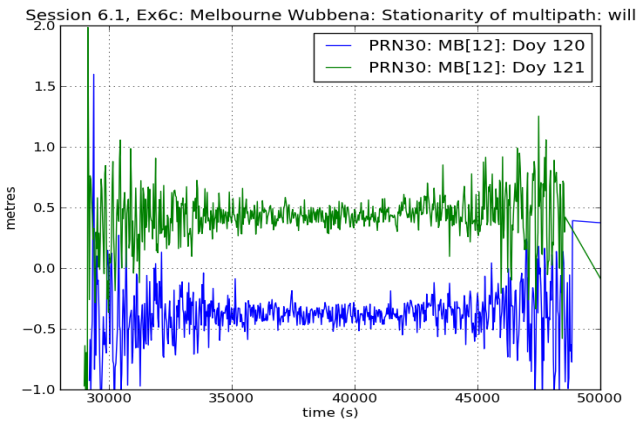

Session 6.1, Ex6c: PRN01 Code noise and multipath: will121
C1-L1-a*(L1-L2) [C1C]
P1-L1-b*(L1-L2) [C1P]
P2-L2-c*(L1-L2) [C2P]
Elev/10 (deg)
metres
time (s)

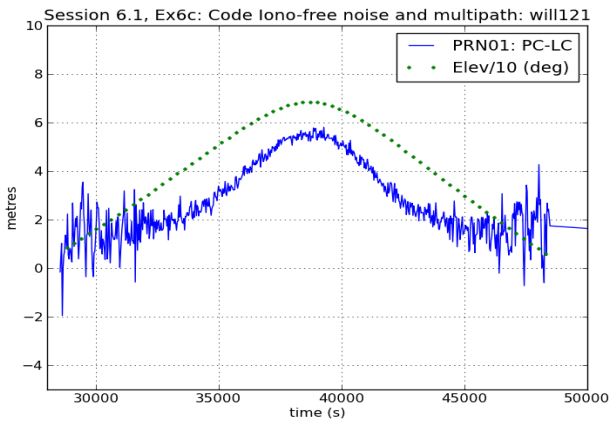

Session 6.1, Ex6c: PRN30 Code noise and multipath: will121
C1-L1-a*(L1-L2) [C1C]
P1-L1-b*(L1-L2) [C1P]
P2-L2-c*(L1-L2) [C2P]
Elev/10 (deg)
metres
time (s)

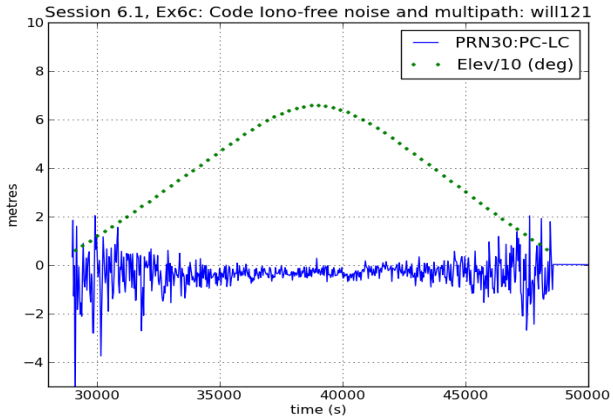

Session 6.1, Ex6c: Melbourne Wubbena combination: will121
PRN01: MB[12]
Elev/10 (deg)
metres
time (s)

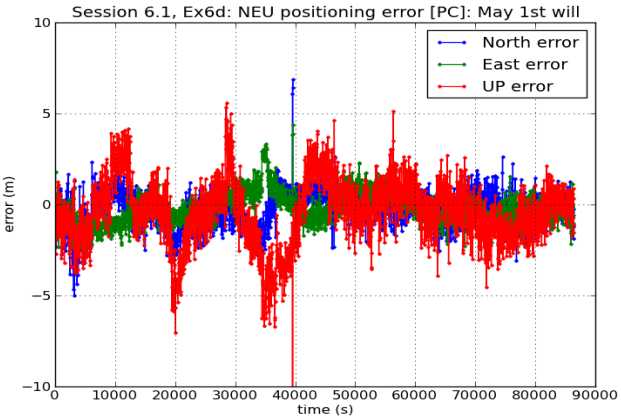

Session 6.1, Ex6c: Melbourne Wubbena combination: will121
PRN30: MB[12]
Elev/10 (deg)
metres
time (s)

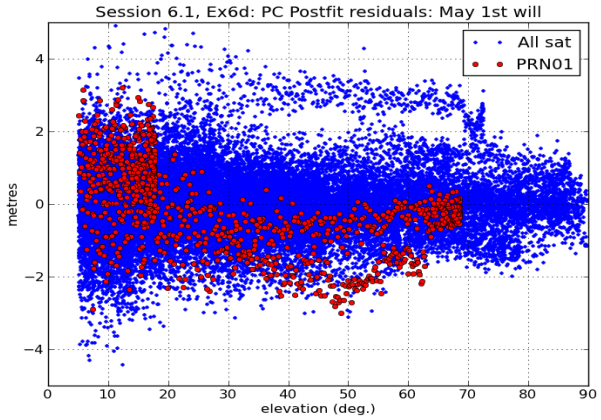

Session 6.1, Ex6c: Melbourne Wubbena: Stationarity of multipath: will
PRN01: MB[12]: Doy 120
PRN01: MB[12]: Doy 121
metres
time (s)

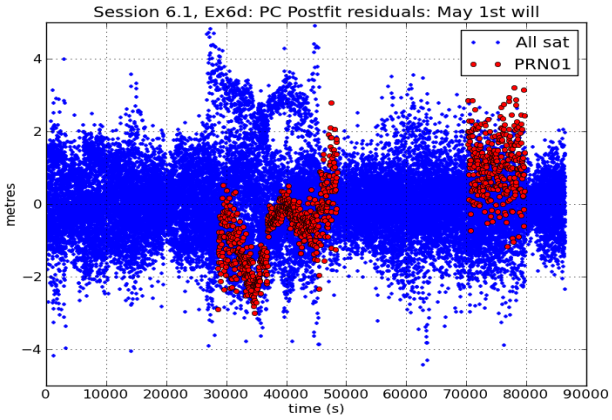

Session 6.1, Ex6c: Melbourne Wubbena: Stationarity of multipath: will
PRN30: MB[12]: Doy 120
PRN30: MB[12]: Doy 121
metres
time (s)

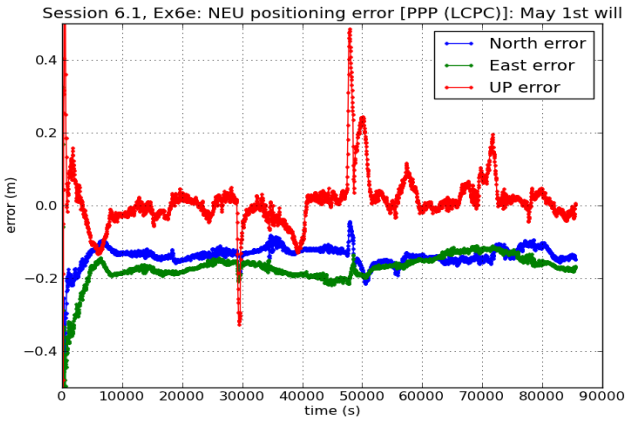

Session 6.1, Ex6c: Code Iono-free noise and multipath: will121
PRN01: PC-LC
Elev/10 (deg)
metres
time (s)

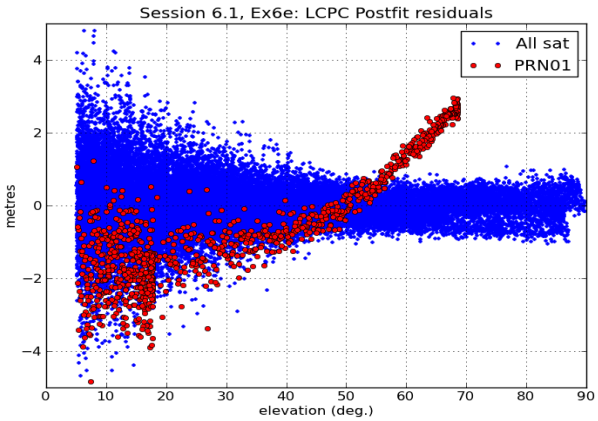

Session 6.1, Ex6c: Code Iono-free noise and multipath: will121
PRN30:PC-LC
Elev/10 (deg)
metres
time (s)

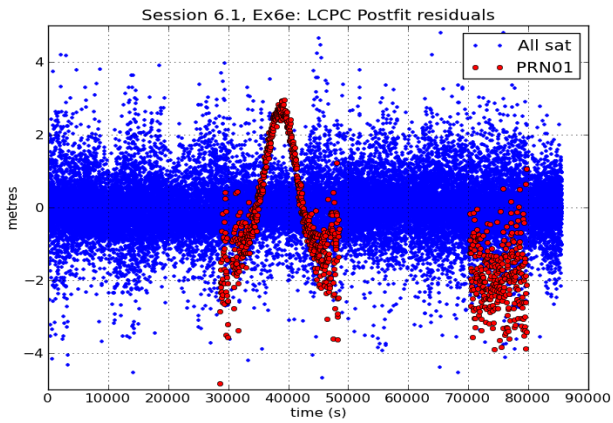

Session 6.1, Ex6d: NEU positioning error [PC]: May 1st will
North error
East error
UP error
error (m)
time (s)

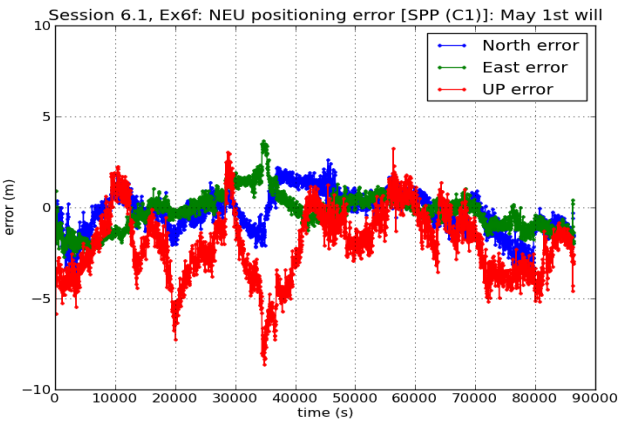

Session 6.1, Ex6d: PC Postfit residuals: May 1st will
All sat
PRN01
metres
elevation (deg.)

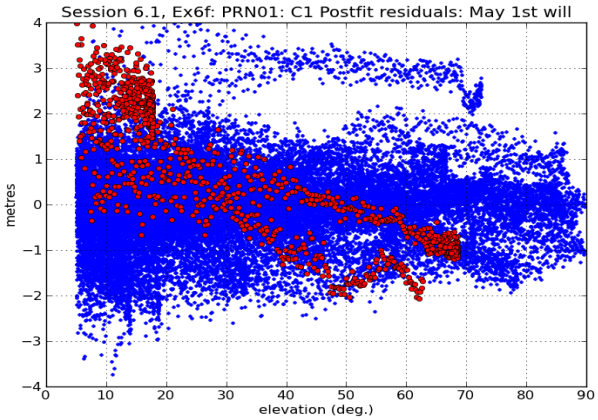

Session 6.1, Ex6d: PC Postfit residuals: May 1st will
All sat
PRN01
metres
time (s)

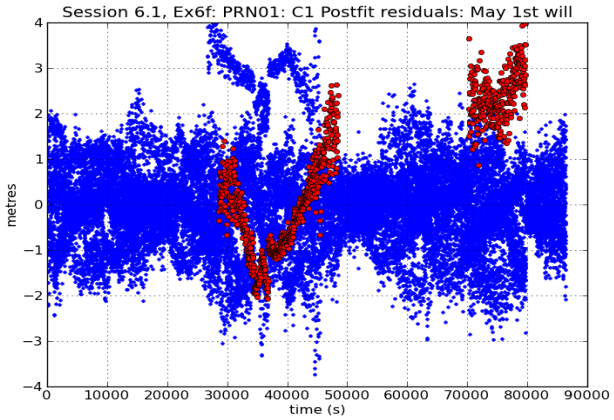

Session 6.1, Ex6e: NEU positioning error [PPP (LCPC)]: May 1st will
North error
East error
UP error
error (m)
time (s)

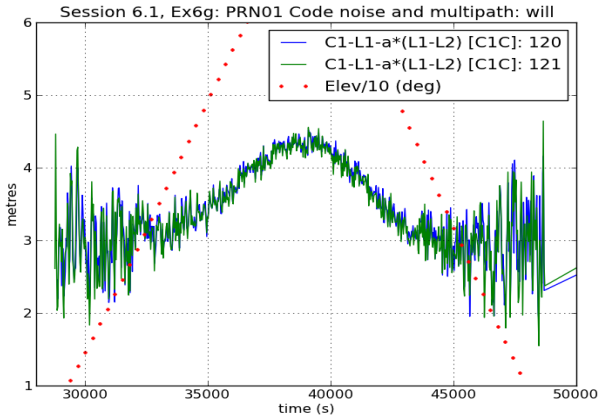

Session 6.1, Ex6e: LCPC Postfit residuals
All sat
PRN01
metres
elevation (deg.)

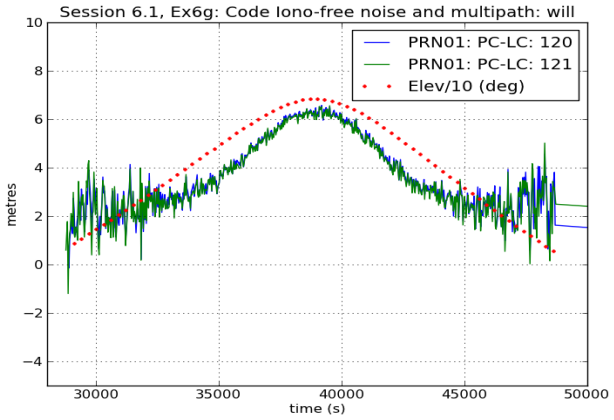

Session 6.1, Ex6e: LCPC Postfit residuals
All sat
PRN01
metres
time (s)

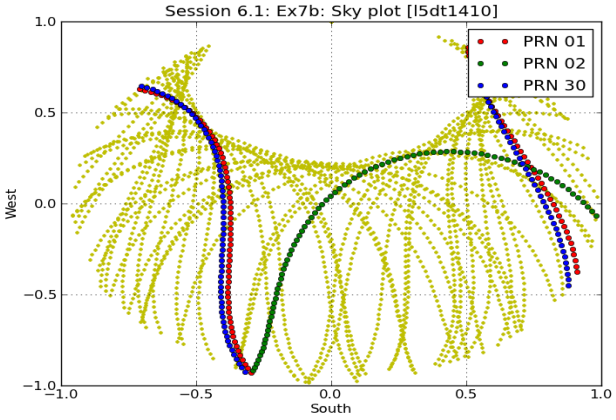

Session 6.1, Ex6f: NEU positioning error [SPP (C1)]: May 1st will
North error
East error
UP error
error (m)
time (s)

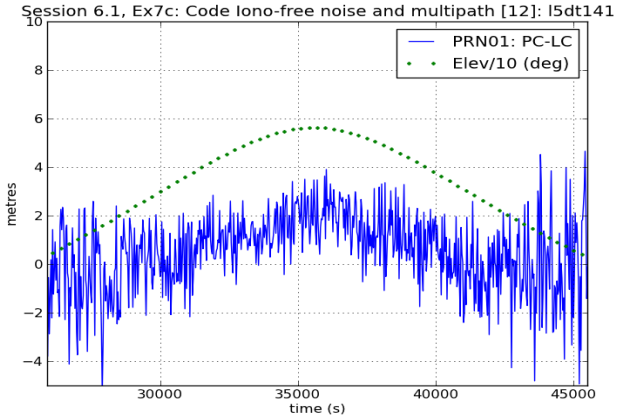

Session 6.1, Ex6f: PRN01: C1 Postfit residuals: May 1st will
metres
elevation (deg.)

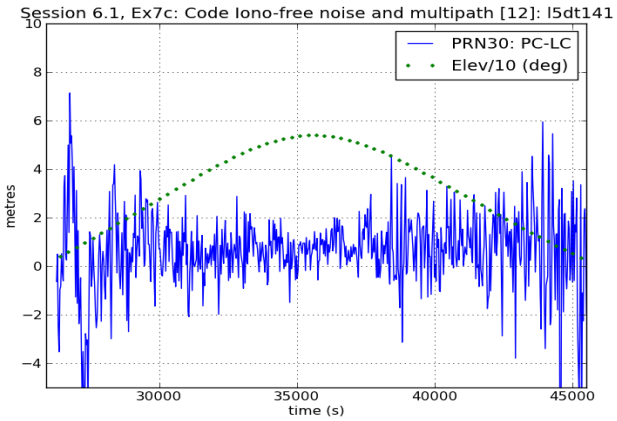

Session 6.1, Ex6f: PRN01: C1 Postfit residuals: May 1st will
metres
time (s)

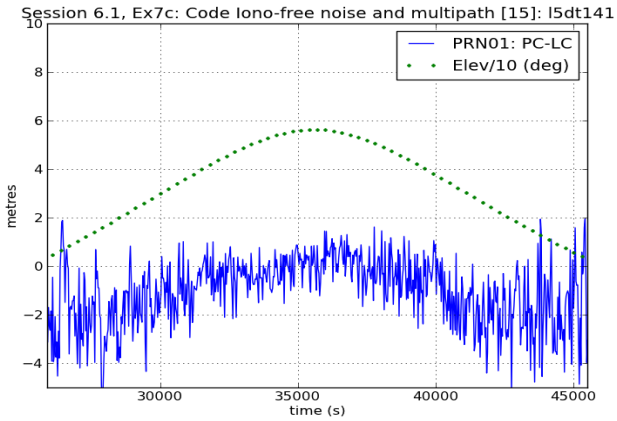

Session 6.1, Ex6g: PRN01 Code noise and multipath: will
C1-L1-a*(L1-L2) [C1C]: 120
C1-L1-a*(L1-L2) [C1C]: 121
Elev/10 (deg)
metres
time (s)

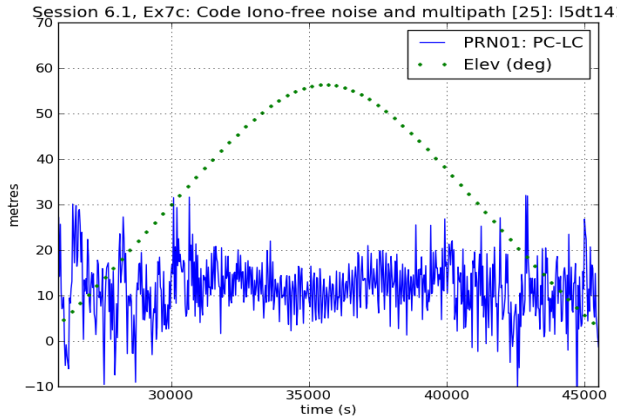

Session 6.1, Ex6g: Code Iono-free noise and multipath: will
PRN01: PC-LC: 120
PRN01: PC-LC: 121
Elev/10 (deg)
metres
time (s)

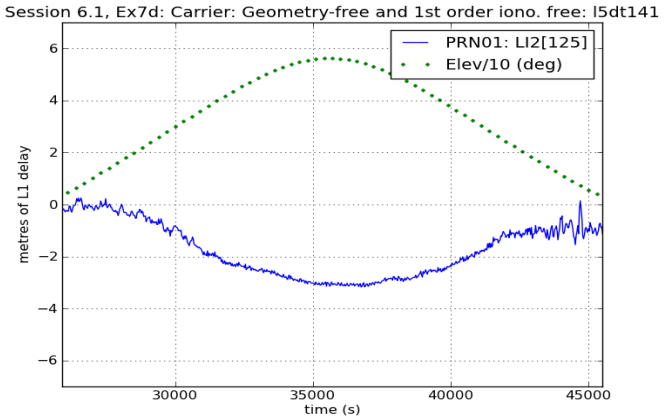

Session 6.1: Ex7b: Sky plot [l5dt1410]
PRN 01
PRN 02
PRN 30
West
South

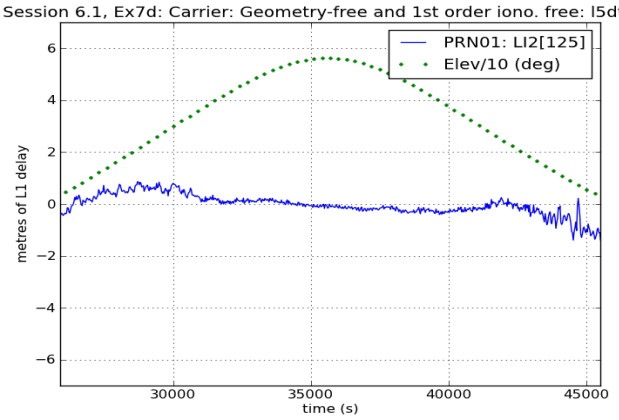

Session 6.1, Ex7c: Code Iono-free noise and multipath [12]: l5dt141
PRN01: PC-LC
Elev/10 (deg)
metres
time (s)

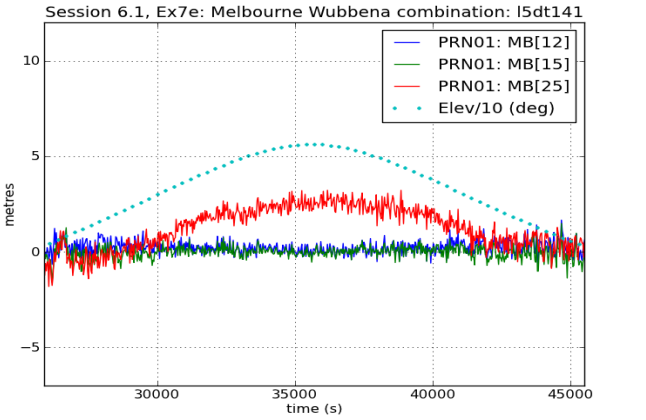

Session 6.1, Ex7c: Code Iono-free noise and multipath [12]: l5dt141
PRN30: PC-LC
Elev/10 (deg)
metres
time (s)

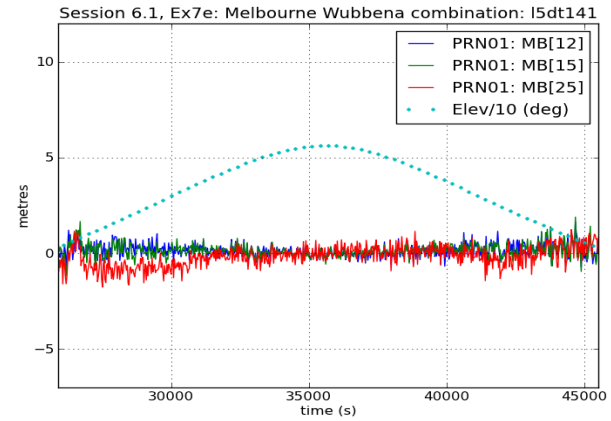

Session 6.1, Ex7c: Code Iono-free noise and multipath [15]: l5dt141
PRN01: PC-LC
Elev/10 (deg)
metres
time (s)


Session 6.1, Ex7c: Code Iono-free noise and multipath [25]: l5dt141
PRN01: PC-LC
Elev (deg)
metres
time (s)


Session 6.1, Ex7d: Carrier: Geometry-free and 1st order iono. free: l5dt141
PRN01: LI2[125]
Elev/10 (deg)
metres of L1 delay
time (s)


Session 6.1, Ex7d: Carrier: Geometry-free and 1st order iono. free: l5dt14
PRN01: LI2[125]
Elev/10 (deg)
metres of L1 delay
time (s)


Session 6.1, Ex7e: Melbourne Wubbena combination: l5dt141
PRN01: MB[12]
PRN01: MB[15]
PRN01: MB[25]
Elev/10 (deg)
metres
time (s)


Session 6.1, Ex7e: Melbourne Wubbena combination: l5dt141
PRN01: MB[12]
PRN01: MB[15]
PRN01: MB[25]
Elev/10 (deg)
metres
time (s)

A. GNSS Elemental Routines and gLAB Libraries

This appendix is aimed at helping the reader to develop his or her own GNSS software tools. It provides two additional laboratory sessions focusing on algorithm implementation and use of `gLAB` source code as a GNSS library.

The first session provides an end-to-end data processing chain for the computation of the Standard Point Positioning (SPP) solution. A set of elemental routines is provided as simple examples to illustrate the implementation of the algorithms in C. Although the session is for single-frequency users, examples of dual-frequency cycle-slip detectors are also included, besides the single-frequency detector.

The second session describes in detail three examples of programs built using `gLAB` source code as a GNSS library. These examples illustrate how to use `gLAB` to develop new programs and tools for GNSS data processing. Other exercises on coordinates transformation, writing RINEX files, and RINEX-3.0 to RINEX-2.11 formatted conversion are also included as complementary examples.

Session A.1. Examples of GNSS Elemental Routines

By Adrià Rovira García
gAGE/UPC & gAGE-NAV, S.L.

Objectives

To help the reader develop his or her own software tools for GNSS data processing by means of a set of elemental examples of algorithm implementation in C.

The examples provided in this session cover a complete end-to-end data processing chain (i.e. from pseudorange measurements to solution of the navigation equations) for GPS single-frequency positioning with broadcast orbits and clocks. They are simple software modules, built as independent units to allow the reader easily to trace and better understand algorithm implementation for Standard Point Positioning (SPP) (the algorithms were explained in detail in Volume I).

Files to use
`brus1810.09o`, `upc10600.11o`, `brdc1810.09n`, `upc10600.11n`, `upc41580.05o`, `brdc1200.09n`

Programs to use
`ObsRinex2txt`, `NavRinex2txt`, `PreProcess`, `Sat2Orb`, `Model`, `FilterLS`, `gLAB_linux`, `graph.py`, `Txt2Rnx`

Overview

As explained in the previous chapters, `gLAB` is the main tool for the exercises in this book. However, `gLAB` is a large program involving several thousand code lines, which makes it difficult to learn it quickly.

In order to simplify the understanding of algorithm implementation and help the reader to develop software, simple examples of elemental routines are provided in this section covering the full processing chain for SPP.

Figure A.1 shows a layout of the data flow indicating all the different programs involved. It takes two different inputs: the receiver measurements (codes and carriers) and the navigation messages broadcast by the satellites.

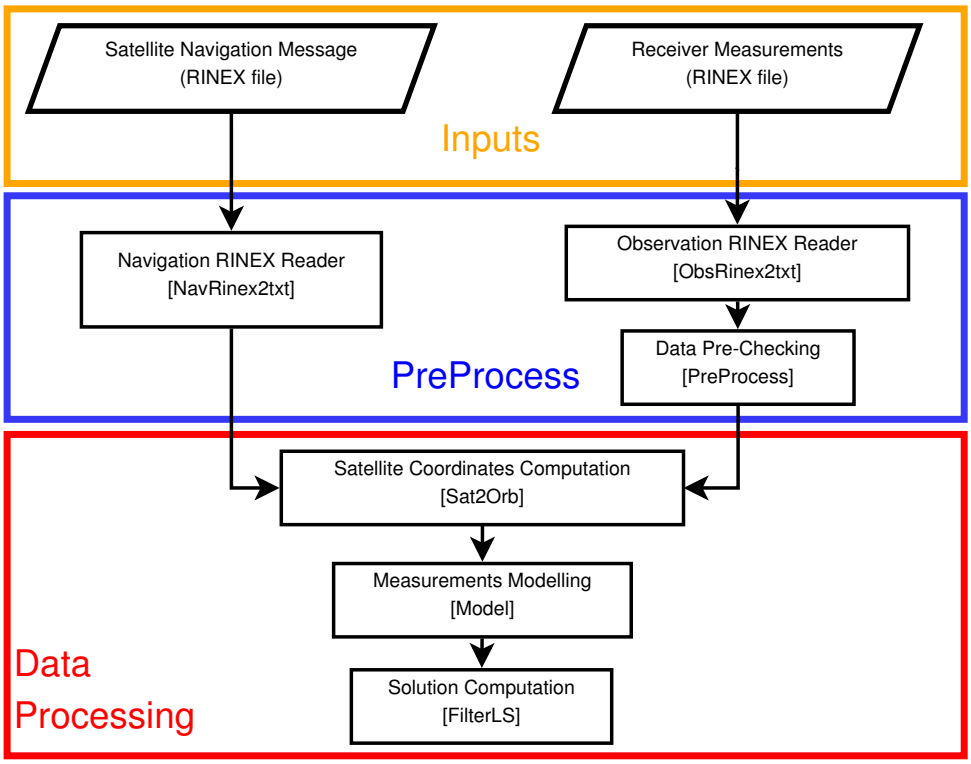


Figure A.1: Overview of the SPP data processing modules: After reading the RINEX data files using programs ObsRinex2txt and NavRinex2txt, the PreProcess module is applied to discard anomalous measurements and for cycle slip detection (section 4.3, Volume I). Then, satellite coordinates are computed using program Sat2Orb (section 3.3, Volume I), and the measurements are modelled by Model (Chapter 5, Volume I). Finally the navigation equations are posed and solved by the FilterLS module (section 6.1.1, Volume I).

Development

Session files:
Copy and uncompress these session files in the working directory:

```
cp ~/GNSS/PROG/SESA1/* .
cp ~/GNSS/FILES/SESA1/* .
gzip -d *.gz *.Z
```

Preliminary

Use the `gLAB` tool suite to compute the SPP solution to be used as the reference solution (i.e. the truth).

The following command-line sentence executes the required processing from the RINEX measurement file `brus1810.09o` and broadcast navigation message file `brdc1810.09n`.

Execute (in a single line):

```
gLAB_linux -input:obs brus1810.09o -input:nav brdc1810.09n
                                   -input:cfg gLAB.cfg > gLAB.out
```

Note: The file `gLAB.cfg` *sets the SPP default configuration of* `gLAB` *with the following predefined options: (1) the nominal values for the tropospheric model are set to* `Simple Nominal` *instead of the* `UNB3` *model; (2) the sampling rate is set to* 30 *seconds; (3) the elevation mask is set to* 5°.

1. RINEX observation file reading

Program `ObsRinex2txt` reads a standard RINEX-2.10 observation file according to the scheme shown in Fig. A.2. First, generic information is extracted from its header and the approximate receiver coordinates (X,Y,Z) are saved in the `station.pos` file for later use by the `Sat2orb`, `Model` and `Filter` modules. Then, pseudoranges are read epoch by epoch up to the end of the file. The maximum set of measurements that can be read from the RINEX file is: [L1, L2, C1, C2, P1, P2, D1, D2, S1, S2]. By default, all 10 measurements are output (missing data are set to 0.0000) but can be shortened by specifying a number as a program argument.[1]

The following sentence reads the first six measurements (i.e. [L1, L2, C1, C2, P1, P2]) of file `brus1810.09o`:

```
ObsRinex2txt brus1810.09o 6 > Observations.txt
```

Each row of output file `Observations.txt` contains all measurements associated with a given satellite and epoch, according to Table A.1.

Table A.1: Formatted output of `ObsRinex2txt` module.

Field	Content
1	Station name
2	Year
3	DoY (Day of Year)
4	Seconds of day (s)
5	GNSS (GPS, GAL, GLO or GEO)
6	PRN satellite identifier
7	Carrier phase measurement: L1 (m)
8	Carrier phase measurement: L2 (m)
9	Pseudorange measurement: C1 (m)
10	Pseudorange measurement: C2 (m)
11	Pseudorange measurement: P1 (m)
12	Pseudorange measurement: P2 (m)
13	Doppler measurement: D1 (Hz)
14	Doppler measurement: D2 (Hz)
15	SNR measurement: S1 (rec. dependent)
16	SNR measurement: S2 (rec. dependent)

[1] Despite the large set of possible measurements provided by the RINEX file, only the C1 code pseudorange will be used for positioning in this example.

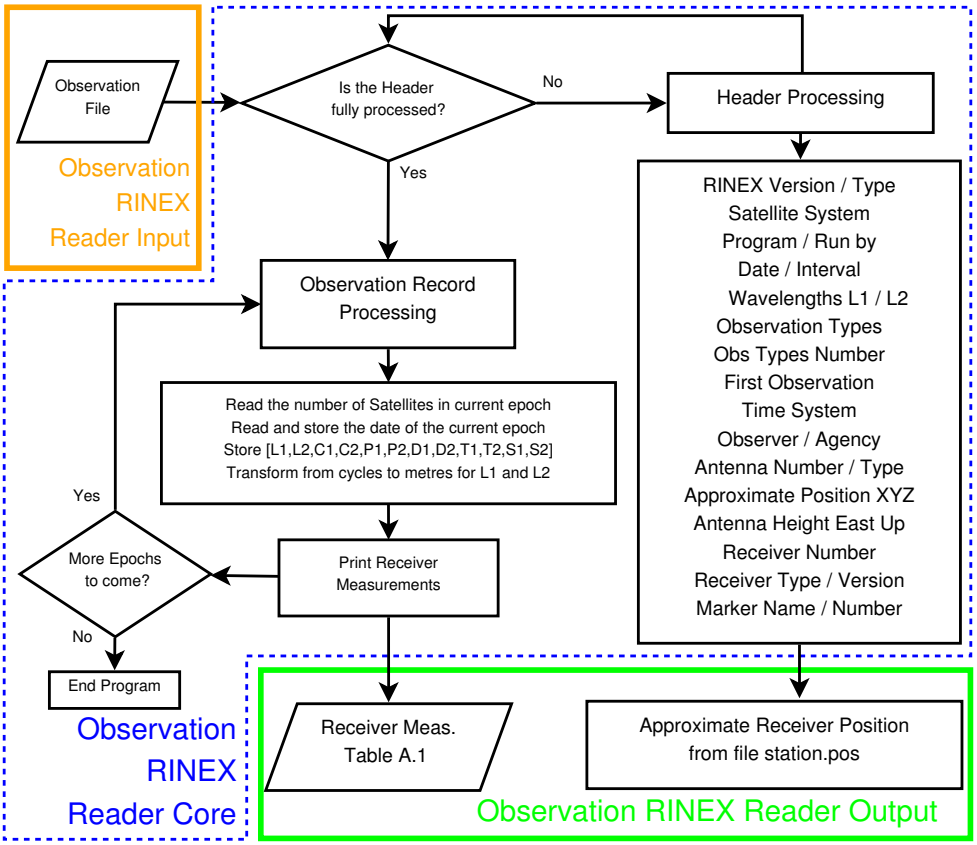


Figure A.2: Observation RINEX reader flowchart.

(a) *Source code inspection*

Edit the source code `ObsRinex2txt.c` and identify the different code lines used for reading the RINEX file and writing the output file.

```
less ObsRinex2txt.c
```

(b) *Checking results*

The results output by the RINEX observation reader can be compared with the `gLAB` output results in the following plot:

```
graph.py -f Observations.txt -x4 -y9 -so -l'C1 SPP Chain'
      -f gLAB.out -x4 -y11 -l 'C1 gLAB' -c '($1=="MEAS")'
           --xl "time (s)" --yl "C1 Code Measurement (m)"
```

2. RINEX navigation file reading

Program `NavRinex2txt` reads a standard RINEX-2.11 navigation file, according to the scheme of Fig. A.3. This file's header contains the coefficients needed to compute the ionospheric correction of the Klobuchar model (section 5.4.1.2.1, Volume I). According to Fig. A.3, these parameters are stored in a `IonoParameters` file, for later use in the `Model` module. The body of the RINEX navigation file contains, among other data, the pseudo-Keplerian elements to allow later calculation of the satellite's position in the `Sat2Orb` module.

The next sentence generates a file named `Ephemerides.txt` whose content is described in Table A.2:

```
NavRinex2txt brdc1810.09n  > Ephemerides.txt
```

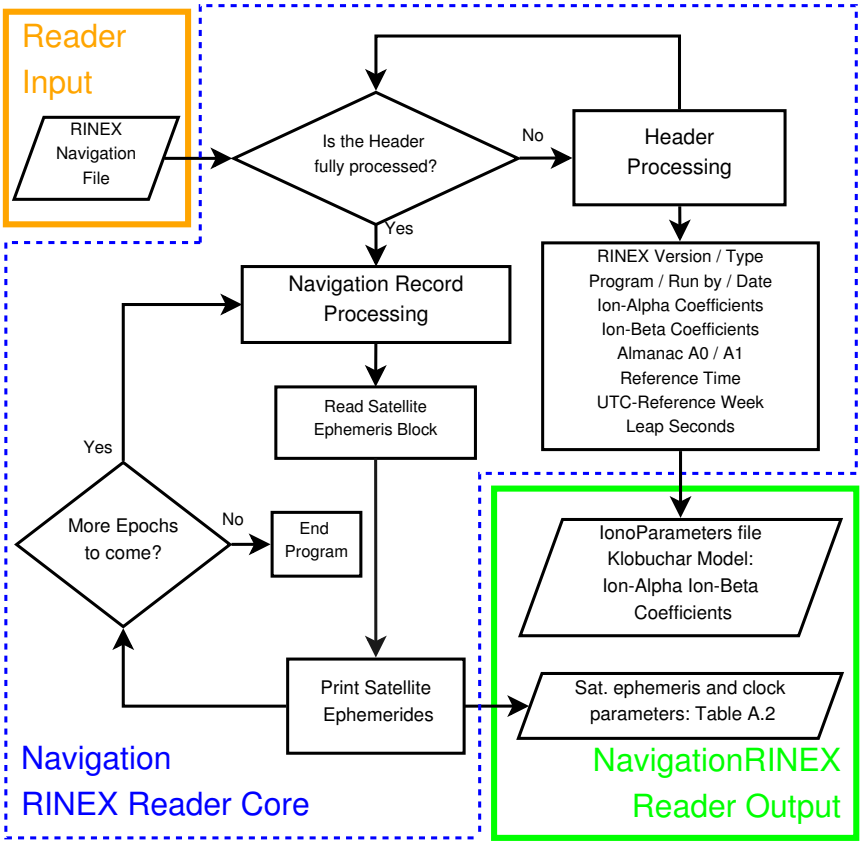


Figure A.3: Navigation RINEX reader flowchart.

Table A.2: Formatted output of `NavRinex2txt` module.

Field	Content
1	Year
2	DoY (Day of Year)
3	Ephemerides time (seconds in DoY)
4	GNSS (GPS, GAL, GLO or GEO)
5	PRN satellite identifier
6	Satellite clock BIAS (s)
7	Satellite clock DRIFT (s/s)
8	Satellite clock RATE (s/s^2)
9	IODE (Issue Of Data Ephemerides)
10	Sin correction radius (m)
11	DeltaN (rad/s)
12	Mo (rad)
13	Cos correction lat (rad)
14	Eccentricity
15	Sin correction lat (rad)
16	sqrt(a) (sqrt(m))
17	Time-of-ephemeris (seconds of GPS week)
18	Cos correction inc (rad)
19	OMEGAo (rad)
20	Sin correction inc (rad)
21	Inclination o (rad)
22	Cos correction radius (m)
23	OmegaO (rad)
24	Omega dot (rad/s)
25	Inclination dot (rad/s)
26	L2 code channel
27	GPS week
28	L2P data flag
29	SV accuracy (m)
30	SV health
31	Satellite TGD (s)
32	IOD clock
33	Transmission time (seconds of GPS week)
34	Fit interval (h)
35	Spare
36	Spare

(a) *Source code inspection*

Edit the source code `NavRinex2txt.c` and identify the different code lines used for reading the RINEX file and writing the output file:

```
less NavRinex2txt.c
```

(b) *Space vehicle health*

The exclusion of satellites flagged as unhealthy[2] in the navigation message protects the user against ephemeris and clock data errors.

```
Plot the SV health flag:
graph.py -f Ephemerides.txt -x 5 -y 30 -so
                    --yl "SV Health" --xl "PRN Number"
```

Which PRNs should be excluded according to the broadcast message of file `brdc1200.09n`?

(c) The Issue Of Data Ephemerides (IODE) identifies the ephemeris block. Check the different values associated with each satellite data set:

```
Visualise IODE:
graph.py -f Ephemerides.txt -x 5 -y 9 -l 'SPP Chain'
                          --xl "PRN Number" --yl "IODE"
```

3. Preprocessing of GPS measurements

Program `PreProcess` performs a single data pre-checking and cycle slip detection. According to Fig. A.4, the C1 code is (roughly) checked[3] to discard corrupted pseudorange measurements. The L1 cycle slips are identified with a single-frequency detector (see Volume I section 4.3.2, example 2).[4] Once a cycle slip is detected, a new carrier phase arch is declared and the '`In arch counter`' is reseted (see Table A.3). Then, the L1 carriers are pre-aligned with the C1 codes (by shifting the carriers by an integer number of cycles).

Previously generated measurements are preprocessed by means of the following sentence, which outputs the file `Obs_Preprocessed.txt` from the previously generated `Observations.txt` file. The output file contents are described in Table A.3.

```
PreProcess Observations.txt > Obs_Preprocessed.txt
```

[2]The SV health check is enabled in `gLAB` by default. It can be disabled by the input argument `--model:satellitehealth` (with double `-`).

[3]Because nominal GPS satellites orbit at an altitude of 20 000 km, a simple test is applied to check if C1 pseudorange values lie between 15 000 and 30 000 km.

[4]This module also implements dual-frequency cycle slip detectors, using the geometry-free combination (section 4.3.1.1, Volume I) and the Melbourne–Wübbena combination (section 4.3.1.2, Volume I). See exercises 9 and 10 at the end of this session.

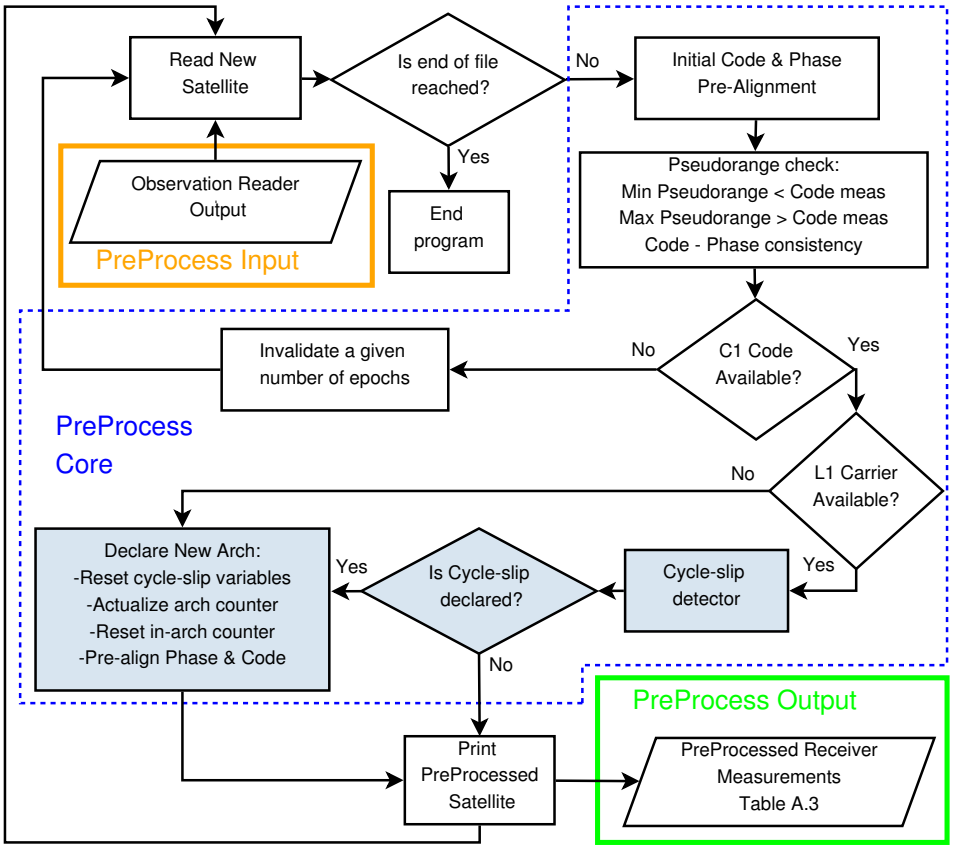


Figure A.4: PreProcess module flowchart.

Table A.3: `PreProcess` module formatted output.

Field	Content
1	Station name
2	Year
3	DoY (Day of Year)
4	Seconds of day (seconds)
5	GNSS (GPS, GAL, GLO or GEO)
6	PRN satellite identifier
7	In arch counter
8	Carrier phase measurement: L1 (m)
9	Carrier phase measurement: L2 (m)
10	Pseudorange measurement: C1 (m)
11	Pseudorange measurement: C2 (m)
12	Pseudorange measurement: P1 (m)
13	Pseudorange measurement: P2 (m)

(a) *Aligned L1 phase with C1 code*

The L1 carrier pre-alignment with the C1 code done by the module `PreProcess` can be visualised with the following plot (for an arch segment for the PRN07 satellite):

```
graph.py -f Obs_Preprocessed.txt -c '($6==7)' -x4 -y8
              -so --cl r -l 'PRN07: L1 aligned to C1'
         -f Obs_Preprocessed.txt -c '($6==7)' -x4 -y10
                  --cl b -l 'PRN07: C1 code' --yn 2e7
```

(b) *Source code inspection*

Edit the source code `PreProcess.c` and identify the different code lines which implement the single- frequency cycle slip detection:

```
less PreProcess.c
```

4. Satellite coordinates computation

Program `Sat2Orb` computes the ECEF satellite coordinates and velocities at signal transmission time, together with the satellite clock offsets and TGDs, see flowchart in Fig. A.5 and section 3.3.1 in Volume I. The main input data are the broadcast navigation message and the measurements. Transmission time is computed using the pseudorange-based method (section 5.1.1, Volume I). After this module has been executed, code and carrier phase measurements are combined in a common file together with satellite coordinates and velocities.

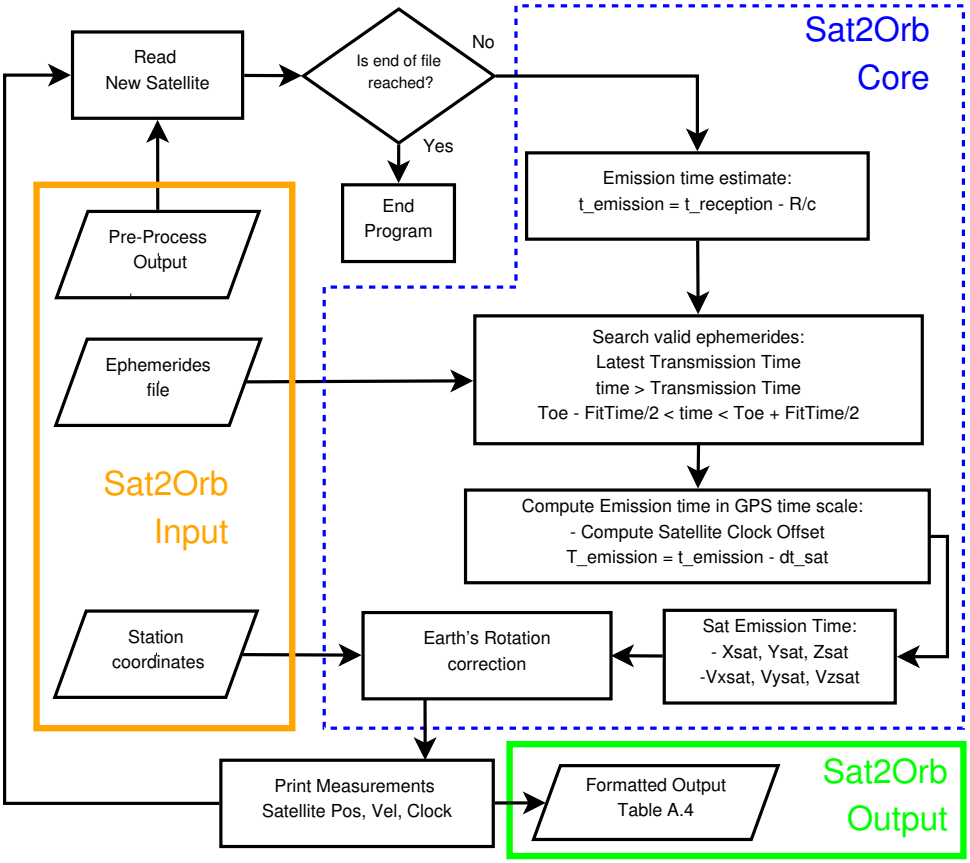


Figure A.5: Sat2Orb module flowchart.

The next sentence computes the satellite coordinates and velocities from the `Ephemerides.txt` data and combines these results with the measurement file `Obs_Satellited.txt` in a common file Obs_Satellited.txt:

```
Sat2Orb Obs_Preprocessed.txt Ephemerides.txt station.pos
                                        > Obs_Satellited.txt
```

Table A.4 shows the contents of the generated file `Obs_Satellited.txt`.

Table A.4: `Sat2Orb` module formatted output.

Field	Content
1	Station name
2	Year
3	DoY (Day of Year)
4	Seconds of day (seconds)
5	GNSS (GPS, GAL, GLO or GEO)
6	PRN satellite identifier
7	In arch counter
8	Carrier phase measurement: L1 (m)
9	Carrier phase measurement: L2 (m)
10	Pseudorange measurement: C1 (m)
11	Pseudorange measurement: C2 (m)
12	Pseudorange measurement: P1 (m)
13	Pseudorange measurement: P2 (m)
14	Geometric flight time (s)
15	X satellite coordinate (m)
16	Y satellite coordinate (m)
17	Z satellite coordinate (m)
18	X satellite velocity (m/s)
19	Y satellite velocity (m/s)
20	Z satellite velocity (m/s)
21	Satellite Clock Offset (m)
22	Satellite TGD (m)

(a) *Satellite coordinates assessment with* `gLAB`

Plot the computed satellite coordinates and assess the discrepancies with the `gLAB` results.

Complete the following steps:

- Plot the satellite coordinates:

```
graph.py -f Obs_Satellited.txt -x 4 -y 15 -l 'X'
         -f Obs_Satellited.txt -x 4 -y 16 -l 'Y'
         -f Obs_Satellited.txt -x 4 -y 17 -l 'Z'
                    --xl "time (s)" --yl "metres"
```

- Compute the discrepancies with the gLAB results:

```
grep MODEL gLAB.out| grep -v INFO |
   gawk '{print $3,$4,$6,$11,$12,$13}'|sort -u > r0.tmp

cat Obs_Satellited.txt|
       gawk '{print $3,$4,$6,$15,$16,$17,"T"}' > r1.tmp

cat r0.tmp r1.tmp|gawk '{i=$1*1" "$2*1" "$3*1;if (NF==6)
      {x[i]=$4;y[i]=$5;z[i]=$6}else{if (length(x[i])!=0)
              print i,$4-x[i],$5-y[i],$6-z[i]}}' > r.dif
```

- Plot the discrepancies with gLAB:

```
graph.py -f r.dif -x2 -y4 -l'X' -f r.dif -x2 -y5 -l'Y'
     -f r.dif -x2 -y6 -l'Z' -xl "time (s)" --yl "metres"
```

(b) *Source code inspection*

Edit the source code `Sat2Orb.c` and identify the different code lines where the emission time and the satellite coordinates and velocities are computed:

```
less Sat2Orb.c
```

5. GPS measurement modelling

The `Model` program implements the measurement modelling for SPP described in Chapter 5 of Volume I. It includes the geometric range computation, relativistic clock correction and the tropospheric and ionospheric corrections. According to Fig. A.6, this module takes three different inputs: the output of the `Sat2Orb` module, the approximate receiver coordinates and the ionospheric parameters for the Klobuchar model.

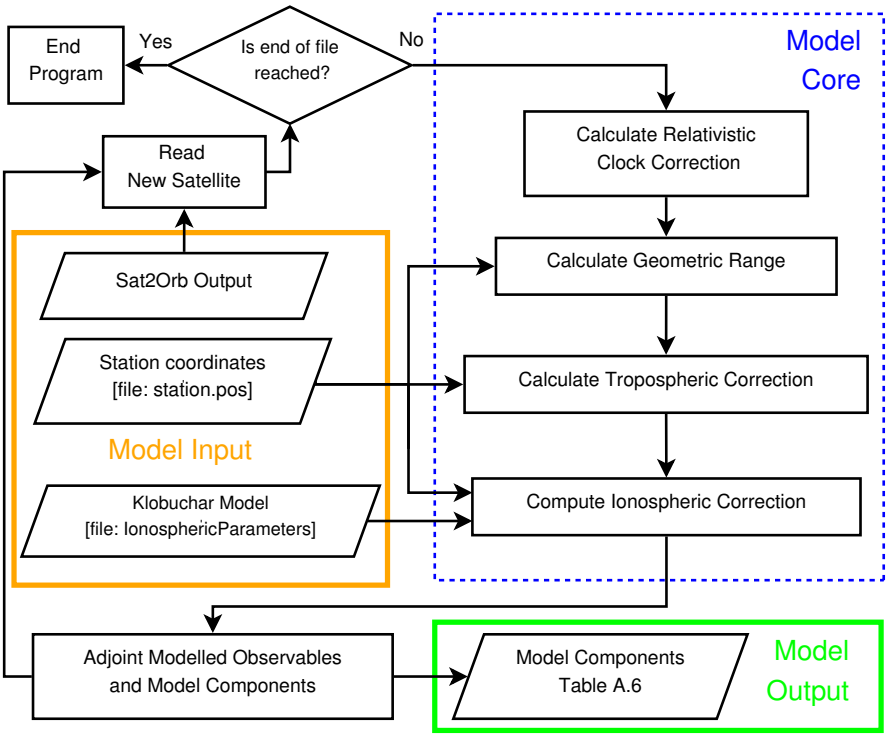


Figure A.6: Model module flowchart.

The next sentence computes the measurement modelling and outputs the model components and satellite coordinates according to Table A.5:

```
Model Obs_Satellited.txt station.pos IonoParameters
                                        > Obs_Modeled.txt
```

Table A.5: Model module formatted output.

Field	Content
1	Station name
2	Year
3	DoY (Day of Year)
4	Seconds of day (s)
5	GNSS (GPS, GAL, GLO or GEO)
6	PRN satellite identifier
7	In arch counter
8	Carrier phase measurement: L1 (m)
9	Carrier phase modelled: L1 (m)
10	Carrier phase measurement: L2 (m)
11	Carrier phase modelled: L2 (m)
12	Pseudorange measurement: C1 (m)
13	Pseudorange modelled: C1 (m)
14	Pseudorange measurement: C2 (m)
15	Pseudorange modelled: C2 (m)
16	Pseudorange measurement: P1 (m)
17	Pseudorange modelled: P1 (m)
18	Pseudorange measurement: P2 (m)
19	Pseudorange modelled: P2 (m)
20	Signal flight time (s)
21	X satellite coordinate (m)
22	Y satellite coordinate (m)
23	Z satellite coordinate (m)
24	X satellite velocity (m/s)
25	Y satellite velocity (m/s)
26	Z satellite velocity (m/s)
27	Satellite clock offset (m)
28	Satellite TGD (m)
29	Rho sat-station module (m)
30	Relativity component (m)
31	Tropospheric component (m)
32	Ionospheric component (L1 m of delay)
33	Satellite elevation (degrees)
34	Satellite azimuth (degrees)

(a) *Signal flight time*

The flight time is estimated from the geometric range between satellite and receiver coordinates at the transmission and reception times, respectively ($dt = \rho/c$), (see section 5.1.1.2, Volume I).

An assessment of the results can be done as follows:

- Plot the signal flight time:

```
graph.py -f Obs_Modeled.txt -x 33 -y 20
     --xl "sat. elev. (deg)"--yl  "flight time (s)"
```

- Compute the discrepancies with the gLAB results:

```
grep MODEL gLAB.out| grep C1C |
                gawk '{print $3,$4,$6,$8}' > r0.tmp

cat Obs_Modeled.txt|gawk '{print $3,$4,$6,$20,$33}'
                                          > r1.tmp

cat r0.tmp r1.tmp|gawk '{i=$1*1" "$2*1" "$3*1;
      if (NF==4) {x[i]=$4}else{if (length(x[i])!=0)
                      print i,$4-x[i],$5}}' > r.dif
```

- Plot the discrepancies with gLAB:

```
graph.py -f r.dif -x5 -y4 --xl degrees --yl seconds
```

(b) *Relativistic clock correction*

The relativistic clock correction is implemented following section 5.2.1 of Volume I. The results can be assessed as in the previous case:

- Plot the relativistic clock correction:

```
graph.py -f Obs_Modeled.txt -x 4 -y 30
                      --xl "time (s)" --yl "metres"
```

- Compute the discrepancies with the gLAB results:

```
grep MODEL gLAB.out| grep C1C |
               gawk '{print $3,$4,$6,$22}' > r0.tmp

cat Obs_Modeled.txt|gawk '{print $3,$4,$6,$30,$33}'
                                          > r1.tmp

cat r0.tmp r1.tmp|gawk '{i=$1*1" "$2*1" "$3*1;
      if (NF==4) {x[i]=$4}else{if (length(x[i])!=0)
                      print i,$4-x[i],$5}}' > r.dif
```

- Plot the discrepancies with gLAB:

```
graph.py -f r.dif -x2 -y4 --xl "time (s)" --y l"m"
```

(c) *Ionospheric correction*

The ionospheric correction is implemented following section 5.4.1.2.1 of Volume I (i.e. the Klobuchar model).

Following a similar procedure as in exercise (5a), assess the computed ionospheric corrections as a function of the elevation.

(d) *Tropospheric correction*

A simple tropospheric correction model is implemented in the `Model` program. The mapping function is given by equation (5.61) in Volume I, and the nominal value for the dry component is given by equation (5.66) in Volume I. The nominal value for the wet component is 10 cm. This model corresponds to selecting the `gLAB` options: `Simple Nominal` and `Simple Mapping`.

Following a similar procedure as in the previous case, assess the computed tropospheric corrections as a function of the elevation.

(e) *Prefit residual computation*

The prefit residual is obtained as the difference between the measured and modelled pseudorange C1, as described in section 6.1 of Volume I.

The results can be assessed as in the previous cases:

- Plot the Prefit residuals:

```
graph.py -f Obs_Modeled.txt -x 33 -y '($12-$13)'
          --xl "Prefit (m)" --yl "elevation (deg)"
```

- Compute the discrepancies with the `gLAB` results:

```
grep PREFIT gLAB.out| grep C1C |
              gawk '{print $3,$4,$6,$8}' > r0.tmp

cat Obs_Modeled.txt|gawk '{print $3,$4,$6,
                           $12-$13,$33}' > r1.tmp

cat r0.tmp r1.tmp|gawk '{i=$1*1" "$2*1" "$3*1;
    if (NF==4) {x[i]=$4}else{if (length(x[i])!=0)
                      print i,$4-x[i],$5}}' > r.dif
```

- Plot the discrepancies with `gLAB`:

```
graph.py -f r.dif -x5 -y4 --xl degrees --yl metres
```

(f) *Source code inspection*

Edit the source code `Model.c` and identify the different lines where previous model components are implemented. Check the algorithm implementation regarding the equations given in Chapter 5 of Volume I:

```
less Model.c
```

6. Least squares navigation equations solver

The navigation equations given in section 6.1 of Volume I are posed and solved by means of a Least Squares (LS) adjustment described in section 6.1.1. This final module takes two different inputs (see Fig. A.7): the output of the `Model` program and the a priori receiver position, stored in the `station.pos` file.

The next sentence poses and solves the navigation equations system and outputs the user solution and postfit residuals according to Table A.6:

```
FilterLS Obs_Modeled.txt station.pos  > Solution.txt
```

Table A.6: FilterLS module formatted output.

Field	Content
1	Station name
2	Year
3	DoY (Day of Year)
4	Seconds of day (s)
5	X receiver coordinate (m)
6	Y receiver coordinate (m)
7	Z receiver coordinate (m)
8	Receiver Clock Offset (m)
9	X receiver solution – a priori (m)
10	Y receiver solution – a priori (m)
11	Z receiver solution – a priori (m)
12	Receiver latitude (degrees)
13	Receiver longitude (degrees)
14	Receiver height (metres)
15	Receiver North a priori difference (m)
16	Receiver East a priori difference (m)
17	Receiver Up a priori difference (m)
18	Measurement identifier
19	Satellites in epoch
20	Satellites postfits (m)

(a) *SPP performance*

The computed receiver coordinates and clock can be assessed with `gLAB` as in the previous exercise 5.

- Positioning error plots:

Horizontal components:

```
graph.py -f Solution.txt -x 15 -y 16
         --xl "East error(m)" --yl "North error(m)"
                  --xn -10 --xx 10 --yn -10 --yx 10
```

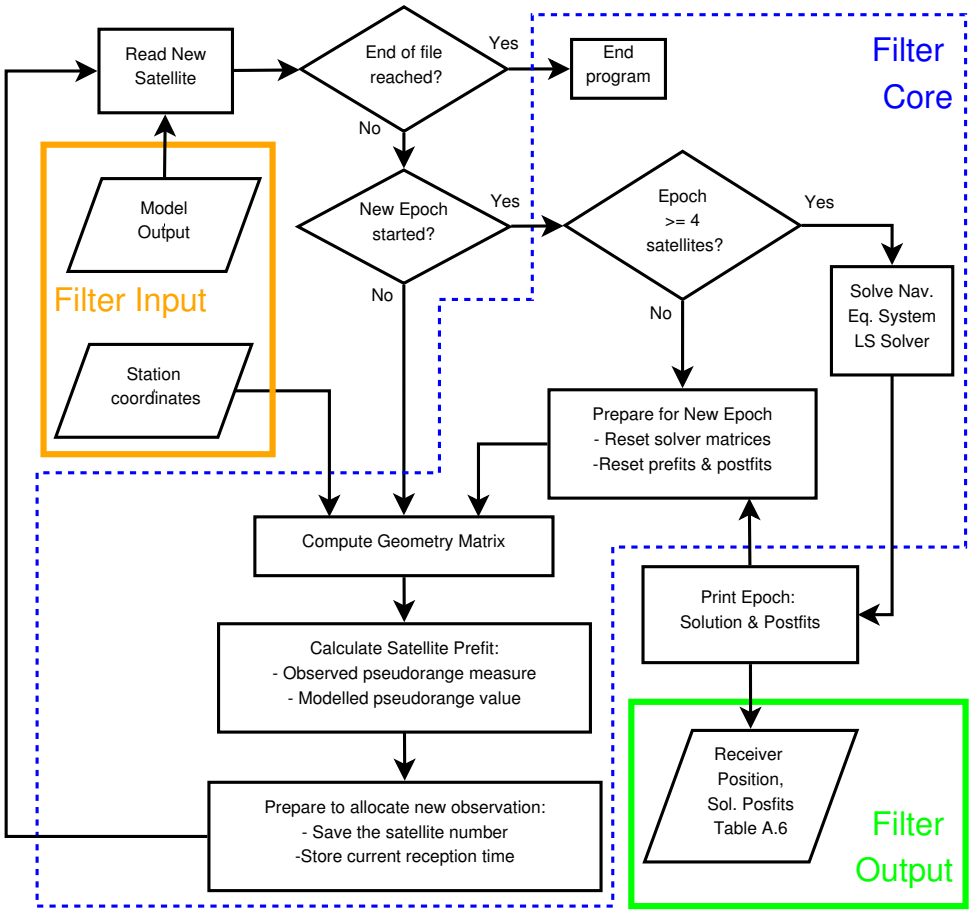


Figure A.7: Filter module flowchart.

Vertical component:

```
graph.py -f Solution.txt -x4 -y17 --yn -15 --yx 15
                --xl "time (s)" --yl "Up error (m)"
```

- Compute the discrepancies with the gLAB results:

```
grep OUTPUT gLAB.out|
         gawk '{print $3,$4,$18,$19,$20}' > r0.tmp

cat Solution.txt | gawk '{print $3,$4,$15,
                              $16,$17,"T"}' > r1.tmp

cat r0.tmp r1.tmp|gawk '{i=$1*1" "$2*1;
       if (NF==5) {x[i]=$3;y[i]=$4;z[i]=$5}
       else {if (length(x[i])!=0)
        print i,$3-x[i],$4-y[i],$5-z[i]}}' > r.dif
```

- Plot the discrepancies with gLAB:

```
graph.py -f r.dif -x2 -y3 -l'North' -f r.dif
          -x2 -y4 -l'East' -f r.dif -x2 -y5 -l'Up'
                  --xl "time (s)" --yl "error (m)"
```

(b) *Receiver clock offset estimate*

The receiver clock offset estimate can be assessed following a similar approach as in the previous case:

- Plot the receiver clock estimate:

```
graph.py -f Solution.txt -x 4 -y 8
       --xl "time (s)" --yl "Receiver Clock (m)"
```

- Compute the discrepancies with the gLAB results:

```
grep FILTER gLAB.out| grep -v INFO |
                gawk '{print $3,$4,$8}' > r0.tmp

cat Solution.txt|gawk '{print $3,$4,$8,"T"}'
                                        > r1.tmp

cat r0.tmp r1.tmp|gawk '{i=$1*1" "$2*1;
   if (NF==3) {x[i]=$3}else{if (length(x[i])!=0)
                        print i,$3-x[i]}}' > r.dif
```

- Plot the discrepancies with gLAB:

```
graph.py -f r.dif -x2 -y3 --xl seconds --yl m
```

(c) *Postfit residuals*

The computed postfit residuals can be compared with the gLAB output as follows, where the first command generates the `Postfits.txt` file containing postfits in a columnar format:

Columnar postfit generation:

```
cat Solution.txt | gawk '{for (i=1;i<=$19;i++)
           print "POSTFIT",$1,$2,$3,$4,$21,$(21+i),
                        $(21+$19+i)}' > Postfits.txt
```

Postfit comparison:

```
graph.py -f Postfits.txt -x 5 -y 8 -so -l
    'SPP Chain Postfits' -f gLAB.out -x 4  -y 8 -c
                '($1=="POSTFIT")' -l 'gLAB Postfits'
```

(d) *Source code inspection*

Edit the source code `FilterLS.c` and identify the different lines where the navigation equations described in section 6.1 of Volume I are built. Find the LS solver step-by-step implementation:

```
less FilterLS.c
```

7. Positioning with carrier-smoothed code

In this exercise, carrier phase measurements will be used to improve the SPP performance. As explained in section 4.2.1, Volume I, the noisy (but unambiguous) code pseudorange measurements can be smoothed with the precise (but ambiguous) carrier phase measurements.

The program `PreProcess` implements the Hatch filter to smooth the code and the single-frequency Cycle Slip (CS) detector (Volume I section 4.3.2,

example 2). This detector is used to reset the smoother after a CS. The performance of the smoothed and unsmoothed solutions is compared next.

(a) *Positioning with carrier-smoothed code*

The SPP data processing chain is executed again, but using the C1 code pseudorange smoothed with the L1 carrier phase measurement:

End-to-end data processing:

```
ObsRinex2txt upc10600.11o 6 > Obs.txt
PreProcess Obs.txt 1Fcs Smooth > Obs_Smooth.txt
NavRinex2txt upc10600.11n > Ephemerides.txt
Sat2Orb Obs_Smooth.txt Ephemerides.txt  station.pos
                                  > Obs_Smooth_Sat.txt
Model Obs_Smooth_Sat.txt station.pos IonoParameters
                                  > Obs_Smooth_Mod.txt
FilterLS Obs_Smooth_Mod.txt station.pos
                                  > Coord_Smooth.txt
```

Note: The argument '`1Fcs Smooth`*' in program* `PreProcess` *is set to enable the single-frequency CS detector and the smoothing.*

(b) *Recomputing the solution, but without applying smoothing*

End-to-end data processing:

```
ObsRinex2txt upc10600.11o 6 > Obs.txt
PreProcess Obs.txt > Obs_no_smooth.txt
NavRinex2txt upc10600.11n > Ephemerides.txt
Sat2Orb Obs_no_smooth.txt Ephemerides.txt station.pos
                                       > Obs_Sat.txt
Model Obs_Sat.txt station.pos IonoParameters
                                       > Obs_Mod.txt
FilterLS Obs_Mod.txt station.pos > Coord.txt
```

(c) *Comparing the smoothed and unsmoothed solutions*:

Horizontal positioning error:

```
graph.py -f Coord.txt -x15 -y16 --cl r -l'Unsmoothed'
    -f Coord_Smooth.txt -x15 -y16 --cl b -l'Smoothed'
                  --xn -10 --xx 10 --yn -10 --yx 10
         --xl "East error (m)" --yl "North error (m)"
```

Vertical positioning error:

```
graph.py -f Coord.txt -x4 -y17 --cl r -l'Unsmoothed'
     -f Coord_Smooth.txt -x4 -y17 --cl b -l'Smoothed'
 --xn 28000 --xx 31000 --xl "time (s)" --yl "metres"
```

8. Simulating CS in L1 and L2 carriers

A procedure to generate a RINEX file with simulated CS 'a la carte' on the L1 and L2 carriers is given next. The input file used is the RINEX `upc41580.05o` with measurements at a sampling rate of 1 s. The output file, with the simulated CS, will be named `upc41580.05o.cs` and will be used as a test file in the next two exercises 9 and 10.

(a) *Generating a CS in the L1 and L2 carriers*

Complete the following steps to add a jump of 6 and 1 wavelengths on the L1 and L2 carriers, respectively, of satellite PRN03 at time 20 000 s:

- Generate a '`*.txt`' file (columnar format) with the following contents: `[name YYYY DoY sec PRN L1(m) L2(m) C1(m) P2(m) P1(m)]`.

Note: These fields are the input of the program `Txt2Rnx` *that converts a* `*.txt` *file to RINEX format (see exercise 5 on session A.2).*

Execute:

```
ObsRinex2txt upc41580.05o 6|gawk '{if ($5=="G") print
 "upc4",$2,$3,$4,$6,$7,$8,$9,$12,$11}'>upc41580.05.txt
```

- Add a jump of 6 and 1 wavelengths on the L1 and L2 carriers (i.e. 1.142 and 0.244 metres), respectively, of satellite PRN03 at time 20 000 s:

Execute:

```
cat upc41580.05.txt | gawk '{if ($5==03 && $4>=20000)
 {$6=$6+1.142;$7=$7+0.244}; printf "%s %s %s %s %03i
   %15.5f %15.5f %15.5f %15.5f %15.5f\n",$1,$2,$3,$4,
                $5,$6,$7,$8,$9,$10}' > upc41580.05.tmp
```

Note: The `printf` *with its explicit format has been used in* `gawk` *to avoid decimal truncation in the sum.*

- Convert the output `*.txt` file to RINEX format:

```
 Txt2Rnx upc41580.05.tmp > upc41580.05o.cs
```

(b) *Assessing results*

- Using the expressions (4.20) of Volume I, calculate the effect of the previous jumps over the geometry-free and the wide-lane combination of carriers.

- Plot the L1-C1, the geometry-free and the Melbourne–Wübbena combinations for PRN03 to depict the jump:

Execute:

Computing the combinations:

```
cat upc41580.05.tmp | gawk 'BEGIN{s12=154/120}
     {if ($5==03) printf "%s %14.4f %14.4f %14.4f \n",
                                        $4,$6-$8,$6-$7,
   (s12*$6-$7)/(s12-1)-(s12*$8+$9)/(s12+1)}'> comb.tmp
```

Plotting the L1-C1 combination:

```
graph.py -f comb.tmp -x1 -y2 --xn 17000 --xx 22000
                         --yn -28705945 --yx -28705935
             -l "L1-C1" --xl "time (s)" --yl "metres"
```

Plotting the geometry-free combination:

```
graph.py -f comb.tmp -x1 -y2 --xn 17000 --xx 22000
                            --yn -132506 --yx -132504
             -l "L1-L2" --xl "time (s)" --yl "metres"
```

Plotting the Melbourne-Wübbena combination:

```
graph.py -f comb.tmp -x1 -y2 --xn 17000 --xx 22000
                         --yn -29173610 --yx -29173600
                -l "MW" --xl "time (s)" --yl "metres"
```

(c) *Additional Comments: Receiver clock adjust*

The data file `upc41580.05o` has been collected with a Septentrio receiver (Polar-Rx2). These receivers make clock adjustments on the code pseudorange (of several milliseconds), but not on the carriers. These jumps, only on code, produce inconsistent code and carrier measurements. This inconsistency must be fixed before applying CS detectors and the smoother.

The following plot allows to depict the code jumps, which are simultaneous for all satellites in view:

Plotting the L1-C1 combination for all satellites in view:

```
graph.py -f upc41580.05.txt -x4 -y'($6-$8)' -l "L1-C1"
     --xl "time (s)" --yl "metres" -t "All satellites"
```

Remark: They are code jumps, but not carrier CS.

The program `PreProcess` makes a code-carrier consistency check to detect such jumps (simultaneous for all satellites). After detecting the jump, its magnitude is estimated (an integer number of milliseconds) and added to the carrier to make this measurement consistent with the code.

9. Single-frequency CS detector

As indicated previously, the program `PreProcess` implements a single-frequency CS detector based on example 2 of section 4.3.2 in Volume I. The performance of this detector is roughly assessed next using the RINEX file `upc41580.05o.cs` generated in the previous exercise.

(a) *Data processing: L1 CS detection*

```
ObsRinex2txt upc41580.05o.cs 6 > Obs.txt
PreProcess Obs.txt 1Fcs dbg > Obs_PreProcessed.txt
```

Note: The argument '`dbg`*' in program* `PreProcess` *is set to enable the debugging mode. In this mode, the program outputs internal information on the CS detection.*

(b) *L1 CS detection analysis*

CS information message:

```
grep CS_L1C1 Obs_PreProcessed.txt | grep "G 03"
```

L1 CS plotting results

```
grep DAT_L1C1 Obs_PreProcessed.txt | gawk '{if($6==3)
               print $4,$10,$12,$14}' > L1C1_PRN03.txt

graph.py -f L1C1_PRN03.txt -x1 -y2 --cl b  -l 'L1C1'
       -f L1C1_PRN03.txt -x1 -y3 --cl r -l 'L1C1 Mean'
 -f L1C1_PRN03.txt -x1 -y'($3+$4)' --cl r -s- -l 'Thr'
 -f L1C1_PRN03.txt -x1 -y'($3-$4)' --cl r -s-
                --xn 13000 --xx 21000 --yn -5 --yx 15
                         --xl "time (s)" --yl "metres"
```

Note: The program `PreProcess` *aligns the carrier with the code measurements after a data hole or CS detection.*

10. Dual-frequency CS detectors

The two examples of CS detectors given in section 4.3.1.1 and 4.3.1.2 of Volume I are also implemented in program `PreProcess`, among the previously analysed single-frequency detector. They are based on using the geometry-free combination of carriers (LI) and the Melbourne–Wübbena (MW) combination of carrier and code measurements. The performance of these detectors is roughly assessed as follows with the RINEX measurement file `upc41580.05o.cs` generated in the exercise 8.

(a) *Data processing: LI and LW CS detection*

```
ObsRinex2txt upc41580.05o.cs 6 > Obs.txt
PreProcess Obs.txt 2Fcs dbg > Obs_PreProcessed.txt
```

(b) *LI CS detection analysis*

CS information message:

```
grep CS_LI Obs_PreProcessed.txt | grep "G 03"
```

LI CS plotting results:

```
grep DAT_LI Obs_PreProcessed.txt | gawk '{ if($6==03)
             print $4, $10, $12, $14 }' > LI_PRN03.txt

graph.py -f LI_PRN03.txt -x 1 -y 2 --cl b -so -l 'LI'
  -f LI_PRN03.txt -x1 -y3 --cl r -l 'LI Prediction'
  -f LI_PRN03.txt -x1 -y'($3+$4)' --cl r -s- -l 'Thr'
  -f LI_PRN03.txt -x1 -y'($3-$4)' -s- --cl r
          --xn 19920 --xx 20010 --yn -5.82 --yx -5.73
                        --xl "time (s)" --yl "metres"
```

(c) *LW CS detection analysis*

CS information message:

```
grep CS_MW Obs_PreProcessed.txt | grep "G 03"
```

LW CS plotting results:

```
grep DAT_MW Obs_PreProcessed.txt | gawk '{ if($6==03)
            print $4, $10, $12, $14 }' > MW_PRN03.txt

graph.py -f MW_PRN03.txt -x1 -y2 --cl b -s o -l 'MW'
        -f MW_PRN03.txt -x 1 -y 3 --cl r -l 'MW Mean'
  -f MW_PRN03.txt -x1 -y'($3+$4)' --cl r -s- -l 'Thr'
  -f MW_PRN03.txt -x1 -y'($3-$4)' --cl r -s-
             --xn 19920 --xx 20010 --yn -21 --yx -14
                       --xl "time (s)" --yl "metres"
```

Note: The program `PreProcess` *aligns the carrier with the code measurements after a data hole or a CS detection 'by either of the two detectors' (i.e. LI or MW detector).*

11. CS detectors sensibility assessment

(a) Repeat the previous three exercises but adding only 1 wavelength to L1 carrier (i.e. 19 cm).

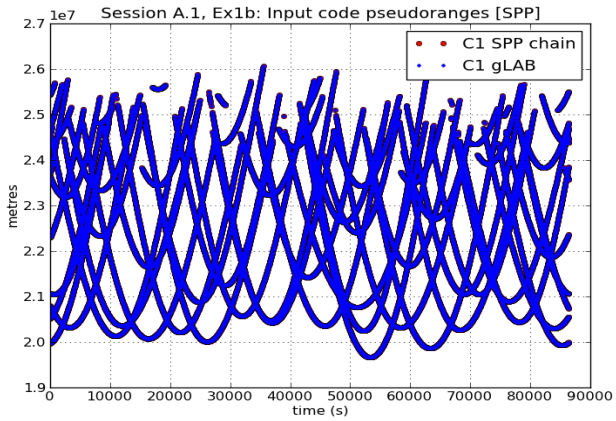
(b) Repeat the previous three exercises but adding 77 and 60 wavelengths to L1 and L2 carriers, respectively.

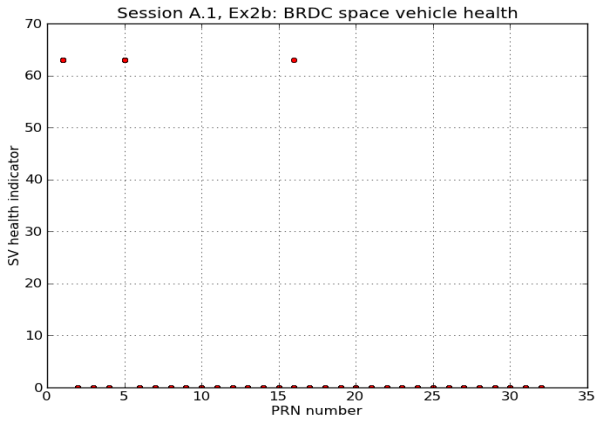
(c) Experiment yourself using different values for the jumps added to L1 and L2 carriers.

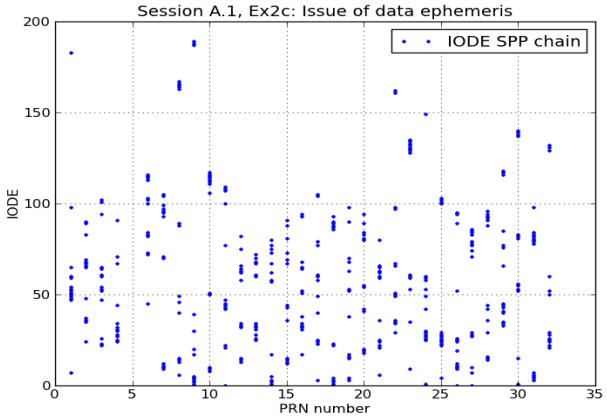
(d) Edit the `PreProcess.c` program and experiment yourself by changing the detection thresholds and other internal parameters used.

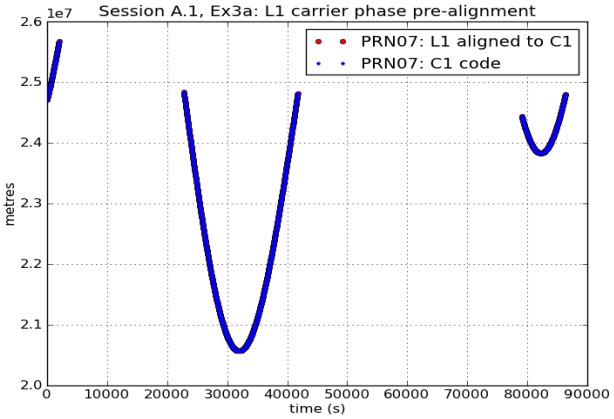
Comment:
The CS detectors presented in this book are shown only as basic examples. It is only intended to introduce the student to this topic, providing some ideas and approaches to follow, but not to fully solve the problem (that has an heuristic component).

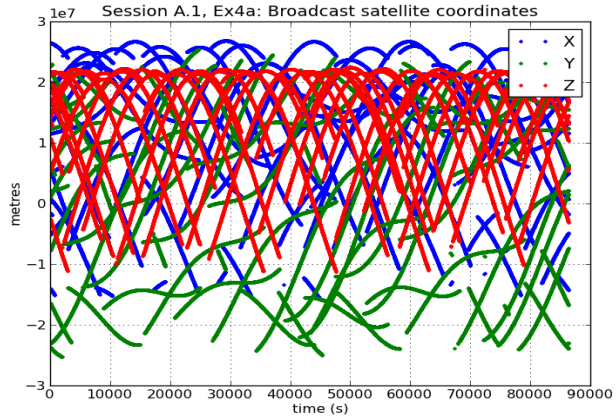
Graphs Session A.1

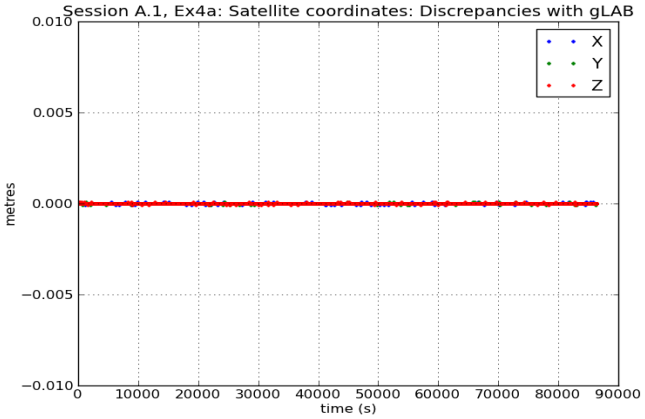


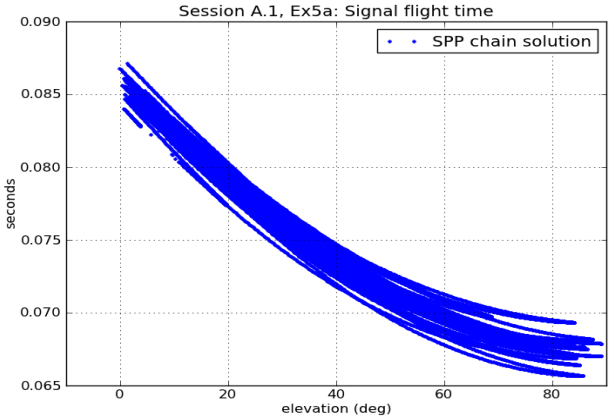


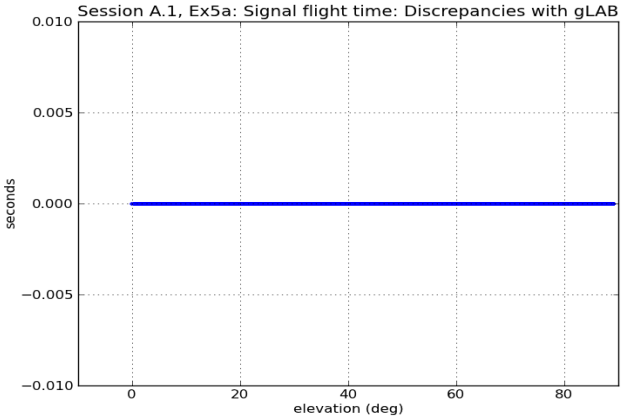


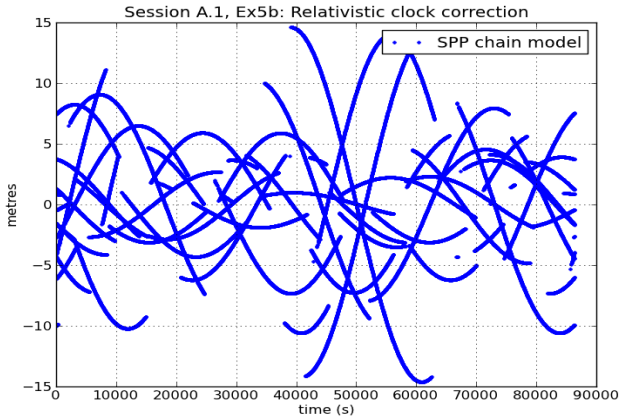


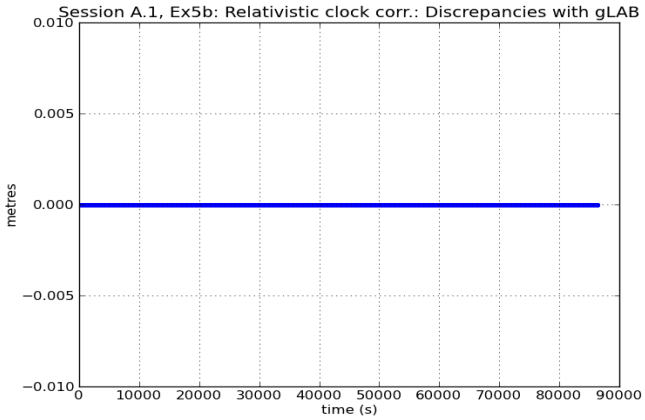


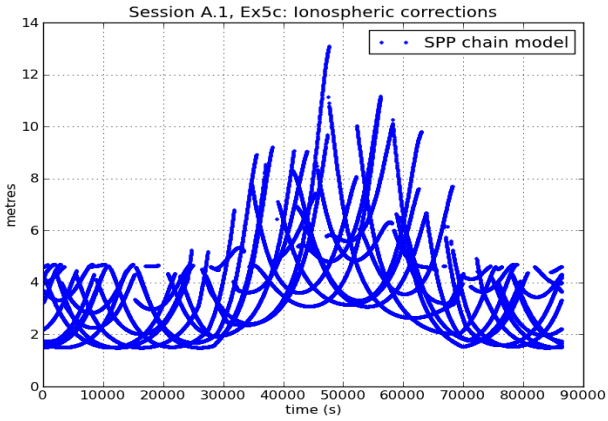


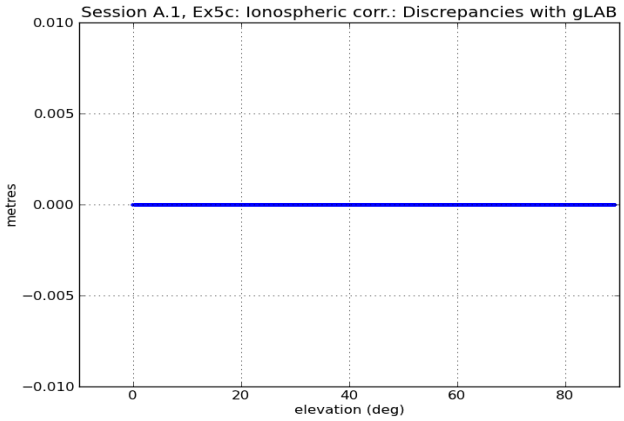


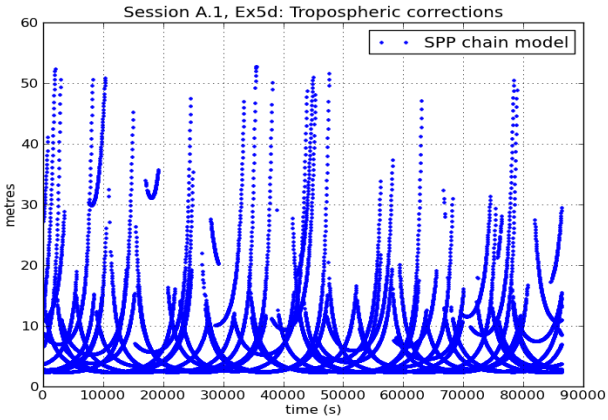


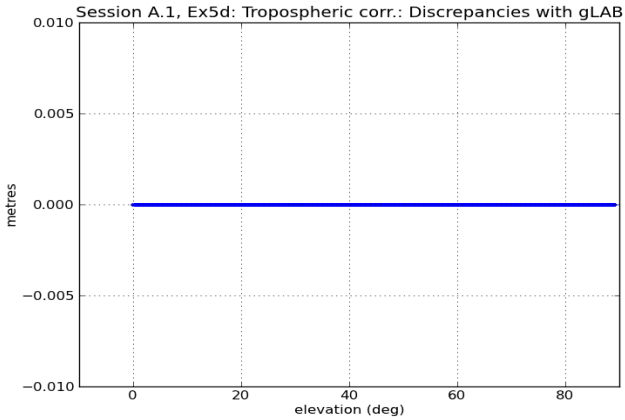

Session A.1, Ex5b: Relativistic clock correction
SPP chain model
metres
time (s)

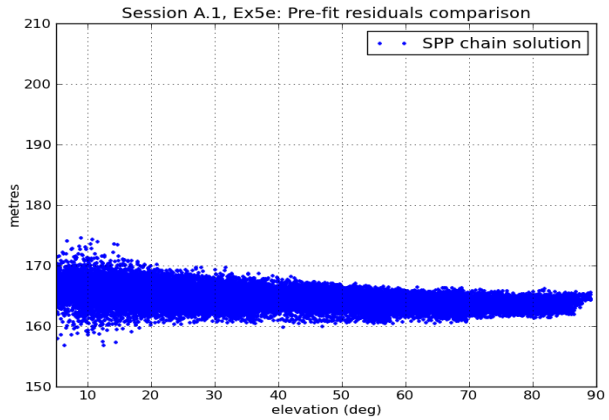

Session A.1, Ex5b: Relativistic clock corr.: Discrepancies with gLAB
metres
time (s)

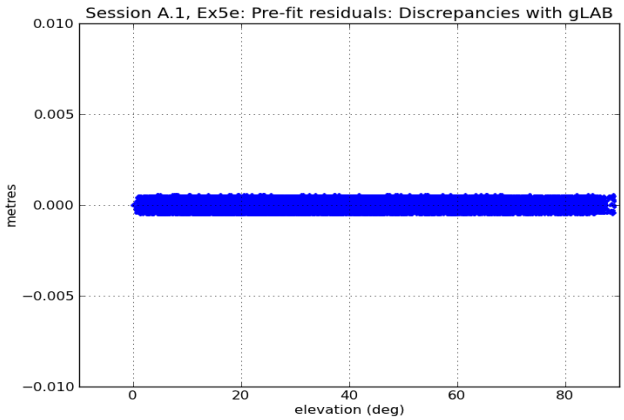

Session A.1, Ex5c: Ionospheric corrections
SPP chain model
metres
time (s)

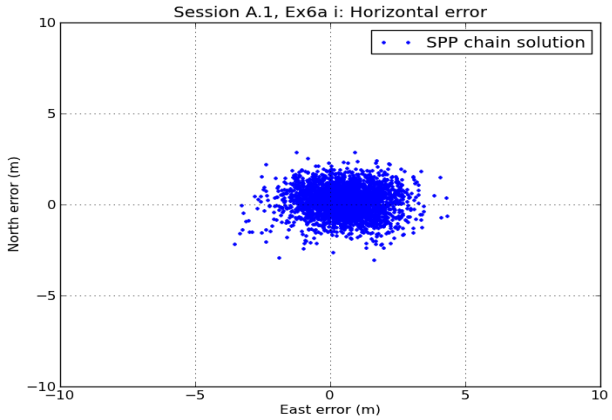

Session A.1, Ex5c: Ionospheric corr.: Discrepancies with gLAB
metres
elevation (deg)

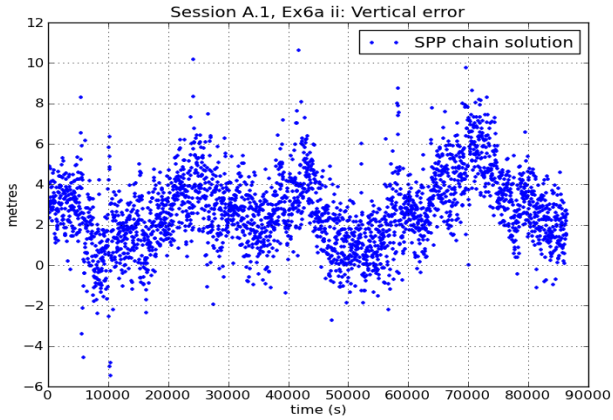

Session A.1, Ex5d: Tropospheric corrections
SPP chain model
metres
time (s)

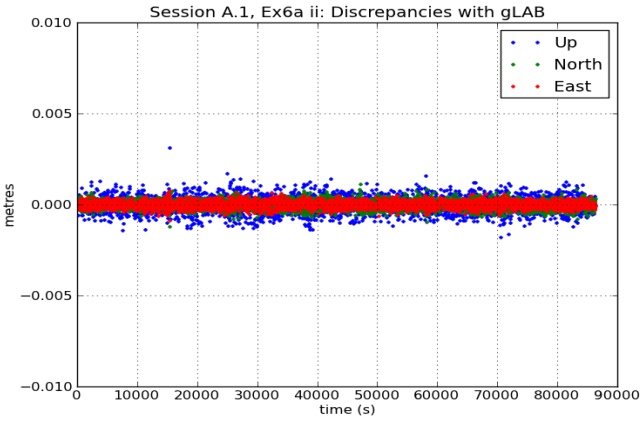

Session A.1, Ex5d: Tropospheric corr.: Discrepancies with gLAB
metres
elevation (deg)

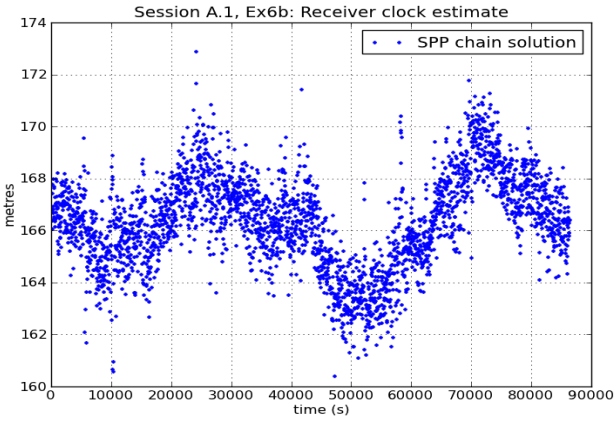

Session A.1, Ex5e: Pre-fit residuals comparison
SPP chain solution
metres
elevation (deg)

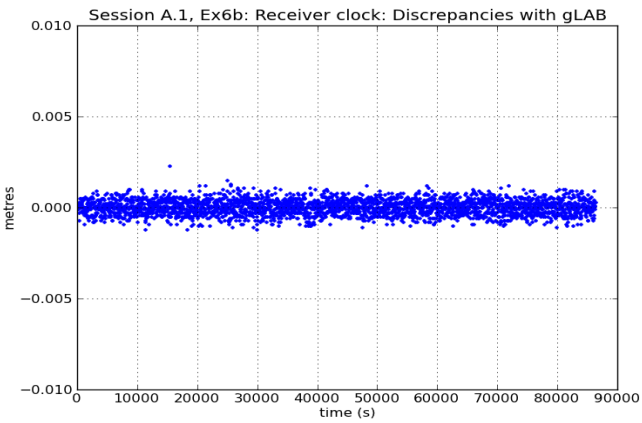

Session A.1, Ex5e: Pre-fit residuals: Discrepancies with gLAB
metres
elevation (deg)

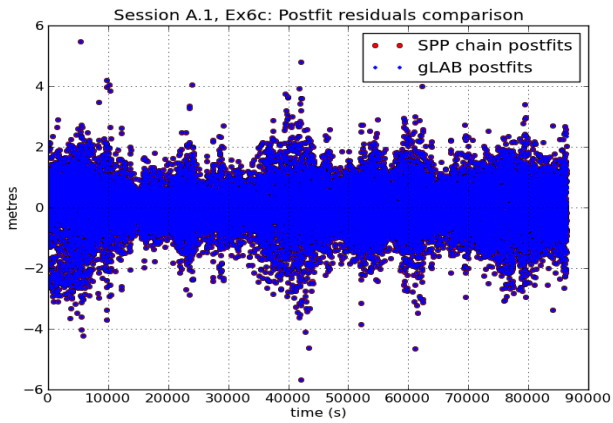

Session A.1, Ex6a i: Horizontal error
SPP chain solution
North error (m)
East error (m)

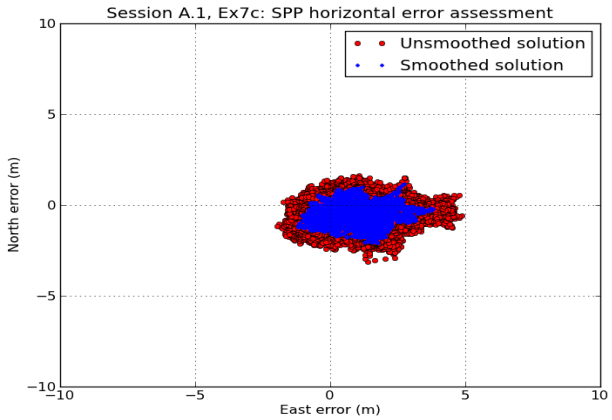

Session A.1, Ex6a ii: Vertical error
SPP chain solution
metres
time (s)

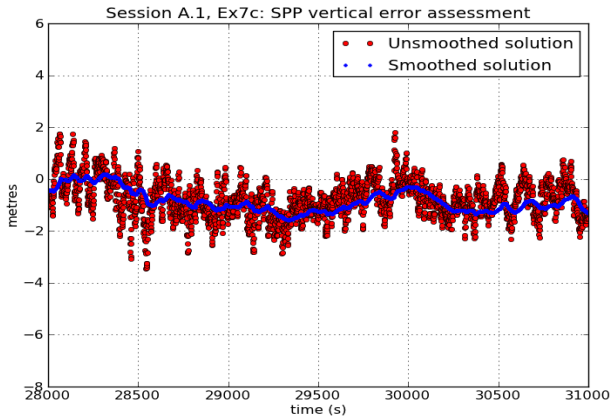

Session A.1, Ex6a ii: Discrepancies with gLAB
Up
North
East
metres
time (s)

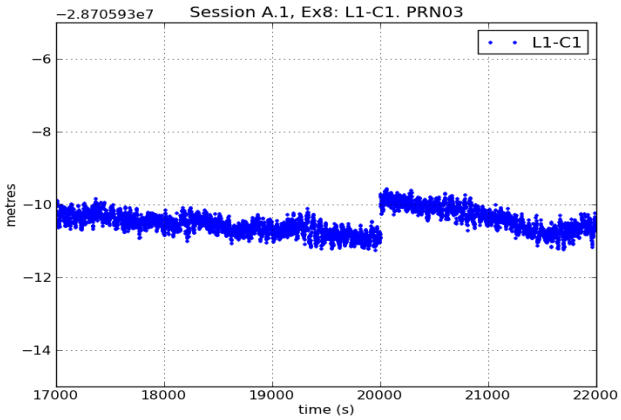

Session A.1, Ex6b: Receiver clock estimate
SPP chain solution
metres
time (s)

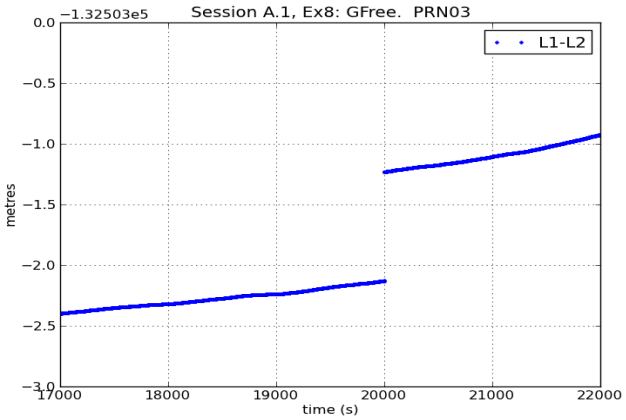

Session A.1, Ex6b: Receiver clock: Discrepancies with gLAB
metres
time (s)

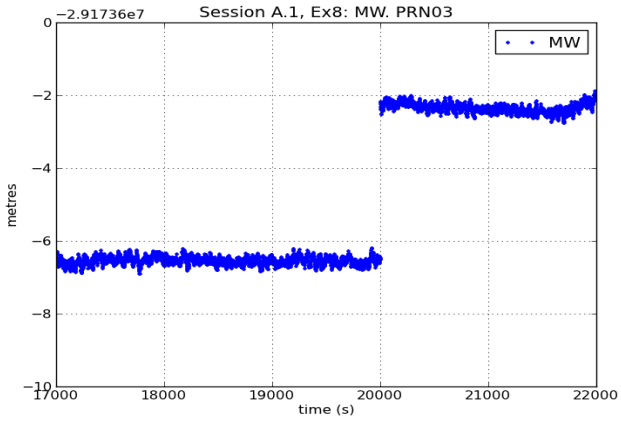

Session A.1, Ex6c: Postfit residuals comparison
SPP chain postfits
gLAB postfits
metres
time (s)

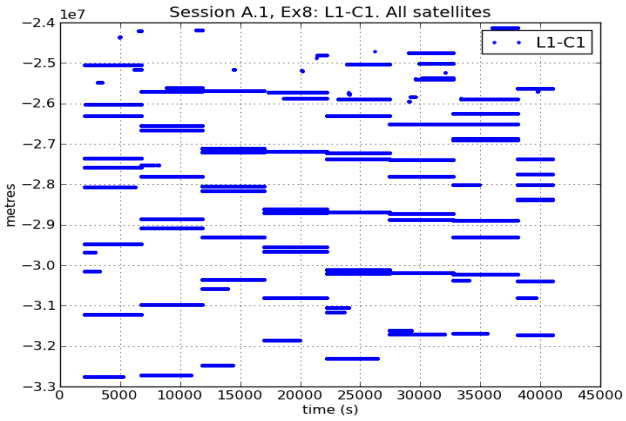

Session A.1, Ex7c: SPP horizontal error assessment
Unsmoothed solution
Smoothed solution
North error (m)
East error (m)

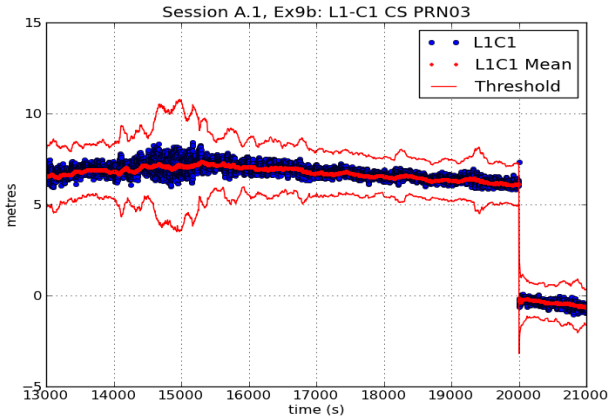

Session A.1, Ex7c: SPP vertical error assessment
Unsmoothed solution
Smoothed solution
metres
time (s)

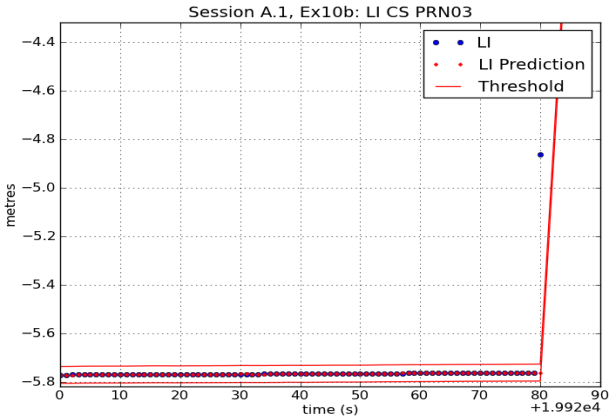

−2.870593e7
Session A.1, Ex8: L1-C1. PRN03
L1-C1
metres
time (s)

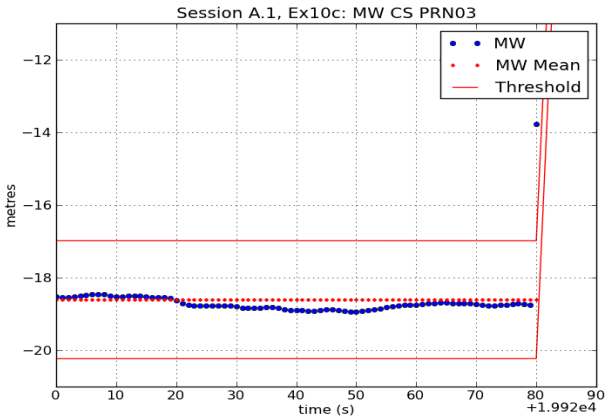

−1.32503e5
Session A.1, Ex8: GFree. PRN03
L1-L2
metres
time (s)

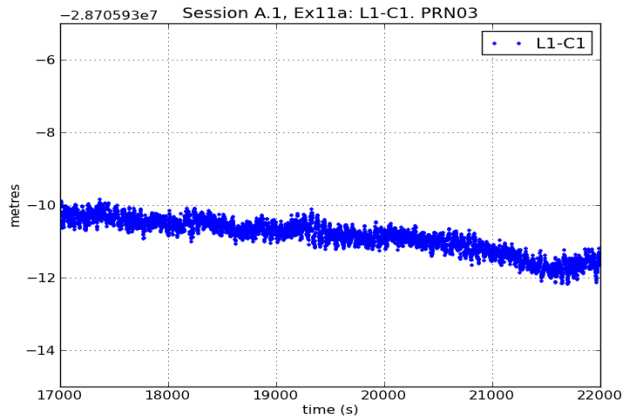

−2.91736e7
Session A.1, Ex8: MW. PRN03
MW
metres
time (s)

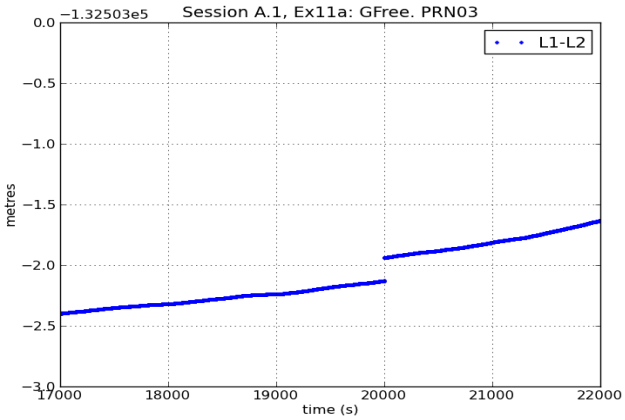

1e7
Session A.1, Ex8: L1-C1. All satellites
L1-C1
metres
time (s)

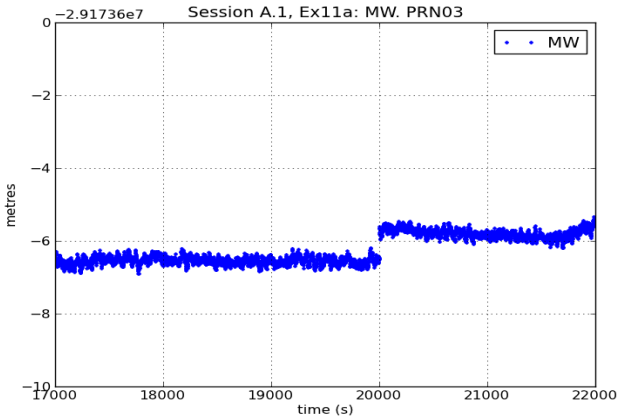

Session A.1, Ex9b: L1-C1 CS PRN03
L1C1
L1C1 Mean
Threshold
metres
time (s)

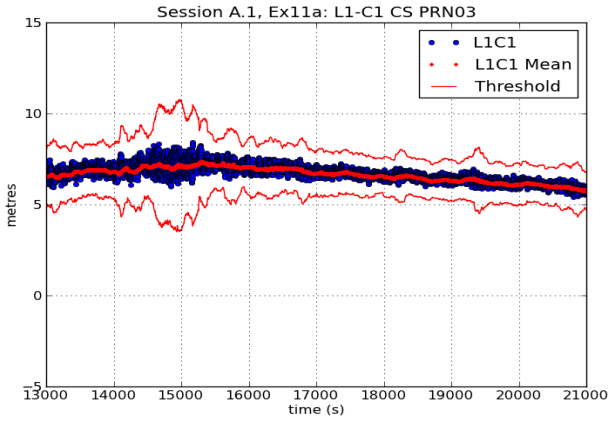

Session A.1, Ex10b: LI CS PRN03
LI
LI Prediction
Threshold
metres
time (s)
+1.992e4

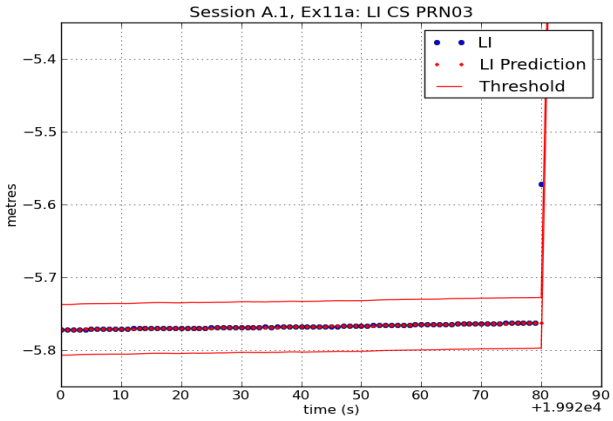

Session A.1, Ex10c: MW CS PRN03
MW
MW Mean
Threshold
metres
time (s)
+1.992e4

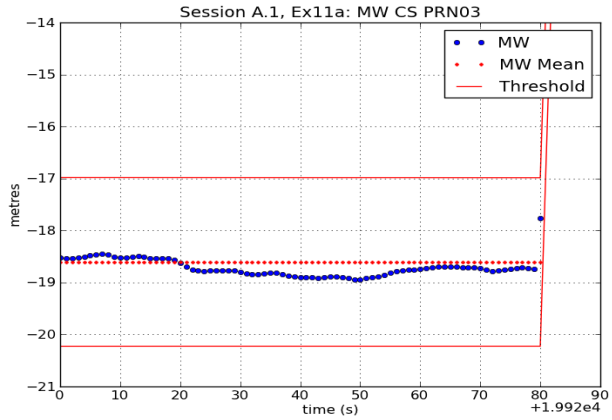

−2.870593e7
Session A.1, Ex11a: L1-C1. PRN03
L1-C1
metres
time (s)

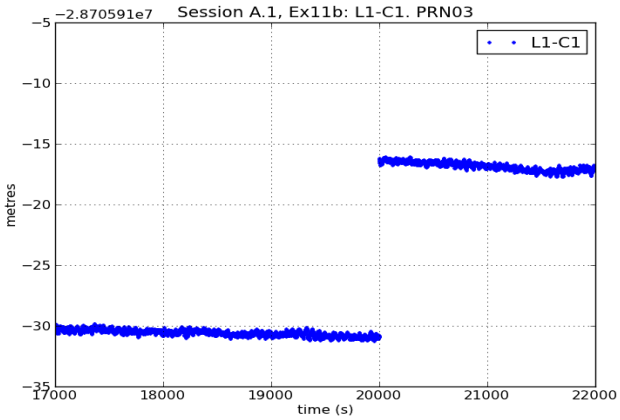

−1.32503e5
Session A.1, Ex11a: GFree. PRN03
L1-L2
metres
time (s)

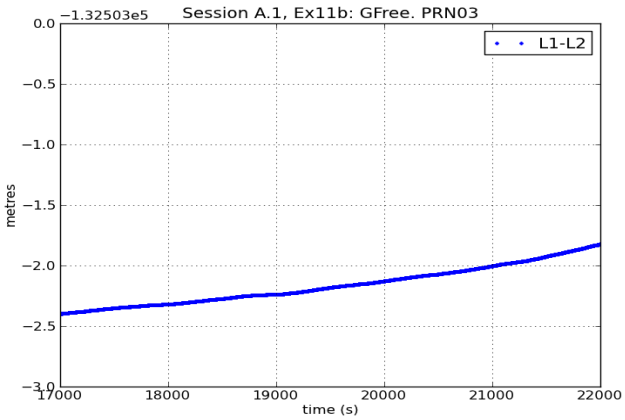

−2.91736e7
Session A.1, Ex11a: MW. PRN03
MW
metres
time (s)

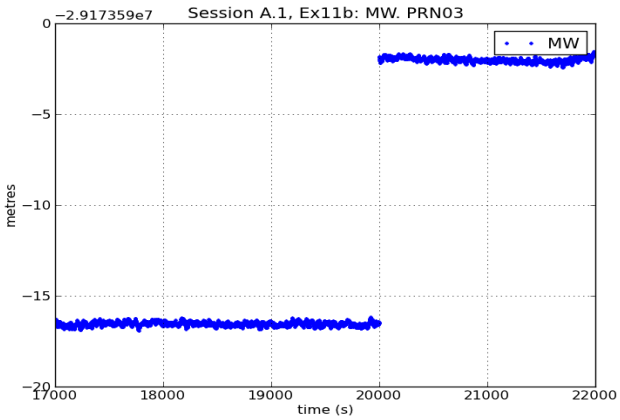

Session A.1, Ex11a: L1-C1 CS PRN03
L1C1
L1C1 Mean
Threshold
metres
time (s)


Session A.1, Ex11a: LI CS PRN03
LI
LI Prediction
Threshold
metres
time (s)
+1.992e4


Session A.1, Ex11a: MW CS PRN03
MW
MW Mean
Threshold
metres
time (s)
+1.992e4


−2.870591e7
Session A.1, Ex11b: L1-C1. PRN03
L1-C1
metres
time (s)


−1.32503e5
Session A.1, Ex11b: GFree. PRN03
L1-L2
metres
time (s)


−2.917359e7
Session A.1, Ex11b: MW. PRN03
MW
metres
time (s)

Session A.2. Programming with gLAB Modules

By Dagoberto Salazar
gAGE/UPC & gAGE-NAV, S.L.

Objectives

To present and explain in detail several examples of how to use gLAB-provided software modules to write new GNSS tools.

Files to use
`madr2000.06o`, `brdc2000.06n`, `iex3326f.12o`, `cart.dat`, `casa2910.95.txt`

Modules to use
`input.h`, `input.c`, `model.h`, `model.c`, `output.h`, `output.c`, `dataHandling.h`, `dataHandling.c`, `filter.h`, `filter.c`, `preprocessing.h`, `preprocessing.c`

Programs to use
`Car2Esf.c`, `Car2Geo.c`, `Trs2Crs.c`, `Txt2Rnx.c`, `Rinex3to2.c`, `handleTime.c`, `readRinexObs.c`, `printSatPosition.c`

Overview

This session introduces the reader to the process of writing his or her own GNSS data handling software tools by taking advantage of the libraries of functions provided by gLAB. Three examples are presented, with increasing levels of complexity.

The examples are written in C, and they work with any gLAB-supported operating system. However, the work in this session is developed using a Linux-based development environment. The user is therefore encouraged to research the specific characteristics of his or her development environment, in case it is different from the one used here.

Each example is carefully explained, showing the relationships with gLAB functions and some common pitfalls to avoid. Also, several tricks for ease of handling are presented. It is assumed that the user has a very basic understanding of C, at the beginner level.

Finally, it is important to emphasise that gLAB is licensed under the Apache License, Version 2.0, allowing the user to develop both free and commercial GNSS data processing tools.

Development

1. Basic time handling

Appropriate time handling is a very important task in GNSS data processing, and the first example (`handleTime.c`) will present some of the data structures and functions related to it.

```
 1  #include <stdio.h>
 2  #include <sys/time.h>
 3  #include <time.h>

 4  #include "dataHandling.h"

 5  int main (int argc, char *argv[]) {

 6     struct timeval tvVar;
 7     struct tm      tmVar;
 8     TTime          myTime;
 9     int            year, month, day, hour, minute;
10     double         second;
11     double         doy;

12     gettimeofday(&tvVar, NULL);

13     localtime_r(&tvVar.tv_sec, &tmVar);
14     year   = tmVar.tm_year1900;+
15     month  = tmVar.tm_mon1;+
16     day    = tmVar.tm_mday;
17     hour   = tmVar.tm_hour;
18     minute = tmVar.tm_min;
19     second = tmVar.tm_sec;

20     fprintf( stdout, "\nCurrent calendar time:
                %04d/%02d/%02d %02d:%02d:%f UTC\n",
                year, month, day, hour, minute, second);

21     fprintf( stdout, "Modified Julian Day  : %d\n",
                MJDN(&tmVar));

22     myTime = cal2t(year, month, day, hour, minute, second);

23     t2doy(&myTime, &year, &doy);

24     fprintf( stdout, "Year / DOY           :
                          %04d / %f\n\n", year, doy );

25  }
```

In this example, lines 1 to 3 include standard C libraries dealing with output (`stdion.h`) and system time handling (`sys/time.h` and `time.h`). The last two provide both functions and data structures needed to manage time. On the other hand, line 4 includes the non-standard, `gLAB`-provided `dataHandling.h` modules.

The `gLAB`-provided modules organise the functions and data structures roughly according to their goals. There are six separate files inside the `glab/` directory. Their names are self-explanatory:

(a) dataHandling
(b) filter
(c) input
(d) model
(e) output
(f) preprocessing

In this example, we only need to call `dataHandling.h` because we are interested in the time handling data structures of `gLAB`. However, it is important to note that often one module may require other modules, so it is common to find cross-references among them.

The program starts in line 5 and ends in line 25. The initial action is to declare the data structures and variables to be used (lines 6 to 11). The `timeval` structure (line 6) is defined in `<sys/time.h>`, while the `tm` structure (line 7) is defined in `<time.h>`. Both are internal system structures needed to handle time.

On the other hand, the `TTime` structure (line 8) is an internal `gLAB` time handling structure defined in the `dataHandling` module. The rest of the variables (lines 9 to 11) are integers and double- precision numbers to hold the values of year, month, day of year (`doy`), seconds, etc.

The next step is to get the current system time using the standard system function `gettimeofday()` on line 12. The current time information is stored in variable `tvVar` (a `timeval` system structure), and in order to break it down into its different components the `localtime()` standard function is invoked on line 13. Lines 14 to 19 take advantage of the resulting `tm` system structure (`tmVar`) to extract year, month, day of the month, hour, minutes and seconds. Note that years are reported starting in 1900 (line 14), and months start at '0' (line 15).

The former way of handling time is called 'calendar time', and in order to print the results line 20 is used. Note that for typesetting purposes the line is divided into three different lines, but it is indeed just one line of code.

The 'calendar time' is not practical for GNSS data processing, and alternative ways to keep track of time are used. One of them is the 'Modified Julian Day' (MJD), and line 21 uses the `gLAB` function `MJDN()` to compute this parameter from the `tm` system structure and print it.

As mentioned above, `gLAB` modules provide a specific time handling structure called `TTime`. This structure is very important in `gLAB` and can be built from calendar time using the function `cal2t()`, as shown in line 22.

Then, line 23 shows how to use the `t2doy()` function to obtain both the year and the day of year from a `TTime` structure, and finally line 24 prints the results. Note that the day-of-year parameter includes the elapsed time within the day in its fractional part.

There are several ways to compile the former program, depending on the compiler used. In a Linux environment this program should be compiled with a single command line,[5] issued from the same working directory where our program is located:

```
gcc -o handleTime handleTime.c input.c dataHandling.c
        output.c model.c filter.c preprocessing.c -lm
```

where:

- `gcc` is the name of the GNU's Not Unix (GNU) compiler;
- `handleTime.c.cpp` and `handleTime` are the names of the input source and output binary files, respectively;
- `input.c`, `dataHandling.c`, `output.c`, `model.c`, `preprocessing.c` are the names of the source files of the `gLAB` modules; and
- `-lm` is the flag to call the mathematical libraries.

It is *very important* to note that, in this simple example, all the `gLAB` module source files, namely both the `.h` and `.c` files, are stored in the same directory as the program we want to compile.

Once the `handleTime` program is compiled, it may be executed with the following command line:

```
./handleTime
```

where the `./` before the program name forces the operating system to execute a program that is *not* in the standard execution 'path'. This practice is common in all the UNIX/Linux-derived systems.

The output of `handleTime` follows similarly:

```
Current calendar time: 2011/02/18 10:47:42.000000 UTC
Modified Julian Day  : 55610
Year / DOY           : 2011 / 49.449792
```

The former compilation procedure, although simple, may not be appropriate when working with complex projects. Therefore, a better way to proceed is to compile all `gLAB` modules into a library and copy the new library and corresponding header files (`.h`) to the appropriate system directories.

First, we should compile `gLAB` modules into 'object' files (`.o`):

```
gcc -lm -c dataHandling.c
gcc -lm -c filter.c
gcc -lm -c input.c
gcc -lm -c model.c
gcc -lm -c output.c
gcc -lm -c preprocessing.c
```

[5] The compilation command line is a single line, although it may appear as two lines because of typesetting constraints.

Then we use the command `ar` to collect the resulting object files into a library:

```
ar r libglab.a dataHandling.o filter.o input.o
              model.o output.o preprocessing.o
```

Finally, the resulting library `libglab.a`, as well as `gLAB` header files (`.h`), must be copied to appropriate system directories. These operations need administrative rights, and they must be carried out as the **root** user:

```
cp libglab.a /usr/lib/
mkdir /usr/include/glab
cp dataHandling.h filter.h input.h model.h
    output.h preprocessing.h /usr/include/glab
```

As can be seen, the `libglab.a` library is copied into the `/usr/lib/` system directory, and then a new subdirectory called `glab` is created in the `/usr/include` system directory. All `gLAB` header modules are copied into the new subdirectory.

The `gLAB` header files were copied into their own subdirectory in order to keep a clean separation between the standard header files in `/usr/include` and the new `gLAB` header files. This is good practice; however, it forces a small change in line 4 of our program, because now the `include` line must reflect the relative position of the header file with respect to the `include` directory:

```
4  #include "glab/dataHandling.h"
```

With this change, the following calls to the compiler become very simple:

```
gcc -o handleTime handleTime.c -lglab -lm
```

Note: The following sentence can be executed as well,

```
gcc -o handleTime handleTime.c -L. -lglab -lm
```

if the library `libglab.a` *is in the directory where the compilation is being done.*

Here the `-lglab` flag calls the modules in the `libglab.a` library. Given that both the `gLAB` library and corresponding header files are located in standard system directories, the new `handleTime` program may be called from anywhere in the system, and may also be included in the 'path'.

For operating systems different from Linux the procedure varies, so the reader must research and follow the specific steps for his or her particular platform. For instance, in a Windows system a *Dynamic Link Library* (DLL) should be built and placed in the appropriate location.

2. Reading from a RINEX observation file

The second example shows how a RINEX observation file can be handled with gLAB modules, introducing the reader to more advanced gLAB functions and data structures.

```
 1  #include <stdio.h>
 2  #include <sys/time.h>
 3  #include <time.h>
 4  #include "glab/dataHandling.h"
 5  #include "glab/input.h"
 6  #include "glab/model.h"
 7  #include "glab/output.h"
 8  int void writeHeaderData( FILE *fd, TEpoch *epoch ) {
 9    fprintf( fd, "\nReceiver name:                %4s\n",
                  epoch->receiver.name );
10    fprintf( fd, "Receiver type:                 %4s\n\n",
                  epoch->receiver.type );

11    fprintf( fd, "Approximate position (XYZ):
                  %13.4lf %13.4lf %13.4lf\n",
                  epoch->receiver.aproxPosition[0],
                  epoch->receiver.aproxPosition[1],
                  epoch->receiver.aproxPosition[2] );

12    XYZ2NEU ( epoch->receiver.aproxPosition,;
               epoch->receiver.aproxPositionNEU );

13    fprintf( fd, "Approx. position (Lat/Lon/H):
                  %9.5lf deg %10.5lf deg %12.3lf m\n\n",
                  epoch->receiver.aproxPositionNEU[0]/d2r,
                  epoch->receiver.aproxPositionNEU[1]/d2r,
                  epoch->receiver.aproxPositionNEU[2] );

14    fprintf( fd, "Antenna type:                  %4s\n",
                   epoch->receiver.antenna.type_ant );

15    fprintf( fd, "Radome type:                   %4s\n\n",
                  epoch->receiver.antenna.type_radome );

16    fprintf( fd, "Antenna reference point (ARP):
                  %8.4lf %8.4lf %8.4lf\n\n",
                  epoch->receiver.ARP[0],
                  epoch->receiver.ARP[1],
                  epoch->receiver.ARP[2] );
17    fprintf( fd, "Measurements interval (s):     %8.4f\n\n",
                  epoch->receiver.interval );
18  }
```

```
19  int main (int argc, char *argv[]) {

20     FILE           *rinexFile;
21     TEpoch         epoch;
22     TOptions       options;
23     TConstellation constellation;

24     if(argc != 2) {
25        printf("\nYou MUST pass the RINEX file name as only
                  parameter. For instance:\n");
26        printf("\n    ./readRinexObs madr2000.06o\n\n");
27        exit(-1);
28     } else {
29        rinexFile = fopen(argv[1], "r");
30     }

31     initEpoch (&epoch);
32     initOptions(&options);

33     readRinexObsHeader( rinexFile, &epoch, &options );

34     writeHeaderData(stdout, &epoch);

35     while( readRinexObsEpoch( rinexFile, &epoch,
                                 &constellation, pFORWARD) ) {

36        printInput(&epoch, &options);
37        getc(stdin);
38     }

39     fclose(rinexFile);
40  }
```

The example is called `readRinexObs.c` and its initial lines (1 to 7) are very similar to the previous example. Note, however, that in this case more `gLAB` modules are included, and the `glab/header-file.h` convention is used.

The following lines, 8 to 18, will be explained later because they belong to a user subroutine called `writeHeaderData()`. Skipping to line 19, where the program really begins, it is important to note the variables `argc` and `argv[]` that are passed as parameters to function `main()`.

These variables are very useful when we want to develop a program that takes parameters from the command line. In this example, the name of the observation RINEX file to be processed will be passed as a parameter just after the name of the program. Integer variable `argc` will contain the number of parameters passed in the command line, and `argv[]` is an array of `chars` holding the names of the parameters.

Lines 20 to 23 declare the name of the main variables, including a reference to read an external file (line 20, file descriptor `rinexFile`) and three important `gLAB`-provided data structures to handle a given epoch of data (`TEpoch`), processing program options (`TOptions`) and satellite data (`TConstellation`). The definition of these and other important data structures may be found in the `dataHandling.h` header file provided by (`gLAB`).

Lines 24 to 30 deal with the command-line parameters. Line 24 checks if the program was invoked with two parameters (i.e. the name of the program itself and the name of the RINEX file). If the number of parameters is not correct, lines 25 to 27 print an error message and a short example of how to run the program correctly, and terminate the program with a negative return code (`exit(-1)`), signalling the operating system that the program aborted due to an error.

In case the program was called with the correct number of parameters, then line 29 will take the **second parameter** (`argv[1]`) and consider it as the RINEX file name, trying to open it in 'read-only' mode.

Proper initialisation is needed for some data structures, lines 31 and 32 taking care of this issue for structures `TEpoch` and `TOptions`. Afterwards, the `gLAB` function `readRinexObsHeader()`, from module `input`, will read the header of the RINEX observation file and store the data in the `TEpoch` data structure.

In order to print part of the data in the RINEX header, the user function `writeHeaderData()` (defined in lines 8 to 18) is called in line 34. It is interesting to realise that the first parameter of this function is a file descriptor, and in this case we are passing the special file descriptor `stdout`. This means that the information will be printed on the standard output (usually the screen), but with a minor modification we may also send the header information to any output file we want.

The function `writeHeaderData()` itself is quite simple. The only remarkable issues are:

- the way to access data inside the `TEpoch` structure (for instance, `epoch->receiver.name` to get the receiver name);
- the use of the `gLAB` function `XYZ2NEU()` to convert from an *Earth-centred, Earth-fixed* reference frame to a *Longitude, Latitude, Altitude* reference frame; and
- the use of the `gLAB` constant `d2r` to convert from radians to degrees.

When the function `writeHeaderData()` ends and control returns to the main program, the following lines (35 to 38) consist of a `while` loop controlled by the `gLAB` function `readRinexObsEpoch()`. This function, also provided by the `input` module, reads the RINEX observation file `one epoch at a time` and returns a non-null integer as long as there were data still to be read in the RINEX file. In this way, the `while` loop will automatically terminate when the RINEX file ends.

The `readRinexObsEpoch()` function will store the epoch data in the `TEpoch` structure, and then line 36 will use the handy `printInput()` function (from module `output`) to print all the available measurements

for that specific epoch. Given that the `while` loop would dump all the measurements to the screen, line 37 implements a small trick that requires pressing ENTER to continue with the next epoch, allowing the user to examine the contents of the RINEX file.

Finally, when the RINEX file ends and the program exits the `while` loop, line 39 closes the file and the program ends.

If the previous steps to create a `gLAB` library were carried out, the compilation of this program would be done with the following command line:

```
gcc -o readRinexObs readRinexObs.c -lglab -lm
```

A user can execute this program with a command line like:

```
./readRinexObs madr2000.06o
```

And the output will be similar to the following:

```
Receiver name:                 MADR
Receiver type:                 ASHTECH Z-XII3

Approximate position (XYZ):    4849202.3737 -360328.9388 4114913.2119
Approx. position (Lat/Lon/H):  40.42916 deg  -4.24966 deg   829.454 m

Antenna type:                  AOAD/M_T
Radome type:                   NONE

Antenna reference point (ARP):   0.0000   0.0000   0.0000

Measurements interval (s):      30.0000

INPUT    2006 200     0.00 GPS 25     1  24684477.9510  24684478.8110
INPUT    2006 200     0.00 GPS 15     1  20399128.4360  20399127.6010
INPUT    2006 200     0.00 GPS  3     1  20833257.3350  20833257.4290
INPUT    2006 200     0.00 GPS 19     1  23119003.9020  23119002.6110
INPUT    2006 200     0.00 GPS 16     1  20333004.2520  20333004.2540
INPUT    2006 200     0.00 GPS 18     1  22727958.1400  22727958.1820
```

3. Computing the position of a given satellite

The third and final example is called `printSatPosition.c`. It reads a RINEX ephemeris file and, with that information, computes the position of a given satellite during a full day.

```
 1  #include <stdio.h>
 2  #include <sys/time.h>
 3  #include <time.h>
 4  #include "glab/dataHandling.h"
 5  #include "glab/input.h"
 6  #include "glab/model.h"
 7  int main (int argc, char *argv[]) {
 8     char           *navfile="brdc2000.06n";
 9     TEpoch         epoch;
10     TGNSSproducts  products;
11     TOptions       options;
12     TTime          currentEpoch, lastEpoch;
13     int            prn=18;
14     double         position[3];
15     double         velocity[3];
16     int            exitCode;
17     initEpoch (&epoch);
18     initOptions(&options);
19     initGNSSproducts(&products);
20     exitCode = readRinexNavFile( navfile, &products );
21     if( !exitCode ) {
22        printf("\nRINEX ephemeris file is not available!!!.\n\n");
23        exit(-1);
24     }
25     currentEpoch = getProductsFirstEpochBRDC( &products );
26     lastEpoch    = getProductsLastEpochBRDC ( &products );
27     while( tdiff( &lastEpoch, &currentEpoch ) >= 0.0 ) {
28        exitCode = getSatellitePVTBRDCraw( &products,
                              &currentEpoch, GPS, prn, position,
                              velocity, NULL, NULL, &options );
29        if( exitCode ) {
30           printf( "GPS %d %d %9.3f %13.3lf %13.3lf %13.3lf\n",
                     prn, currentEpoch.MJDN, currentEpoch.SoD,
                     position[0], position[1], position[2] );
31        }
32        currentEpoch = tdadd( &currentEpoch, 900.0 );
33     }
34  }
```

As with the previous examples, the first lines (1 to 6) load the necessary modules (both standard and gLAB's), and the main program starts at line 7. Then, lines 8 to 16 declare the main variables to be used. Note the following issues:

- In line 8 the name of the RINEX ephemeris file is already fixed (`brdc2000.06n`). Remember that it may also have been passed as a command-line parameter, as in the previous example.
- The PRN identification of the satellite is also fixed in this example (18). The same remark as for the file name applies here.
- A new gLAB data structure is introduced (`TGNSSproducts`), provided by the `dataHandling` module. Some of the gLAB data structures are properly initialised on lines 17 to 19.
- Both `position[]` and `velocity[]` variables are three-dimensional arrays.

Line 20 is very important because it is responsible for reading the contents of the RINEX ephemeris file and storing them in the `TGNSSproducts` data structure. The gLAB function `readRinexNavFile()` is used for this, available at the `input` module.

This function returns an integer exit code to signal the success or failure of the operation. A '+1' exit code means that the reading was successful, while '0' means the file could be read but some errors were found (bad format, for instance) and '−1' signals that the file could not be opened. This characteristic is then used in lines 21 to 24, where the exit code is checked to be positive; if it does not pass the test, an error line is printed and the program is terminated.

Afterwards, code lines 25 and 26 initialise the `TTime` data structures called `currentEpoch` and `lastEpoch`. In order to achieve this, functions `getProductsFirstEpochBRDC()` and `getProductsLastEpochBRDC()` are used to fetch the first and last available epochs of data in the ephemeris file, respectively.

Lines 27 to 33 comprise a `while` loop designed to transverse all the available ephemeris data and compute the satellite's position at different epochs. The behaviour of this loop is controlled by the gLAB function `tdiff()` to check if the difference (in seconds) between `TTime` data structures `lastEpoch` and `currentEpoch` is greater than zero. If so, the gLAB function `getSatellitePVTBRDCraw()` will try to compute the Position, Velocity, Time (PVT) of satellite PRN18 at the provided `currentEpoch`.

Again, the resulting function exit code is used in lines 29 to 31 to check that the computed PVT is correct, and print the corresponding values for time and position. This check is very important, because a given RINEX navigation file may not have the ephemeris information needed to compute the PVT of a given satellite during the whole period of interest.

Finally, line 32 uses the `tadd()` function to increase `currentEpoch` by 900 seconds (15 minutes) and the loop repeats itself until `currentEpoch` is higher than `lastEpoch`. When the logic flow leaves the `while` loop, the program ends at line 34.

Compilation of this program is carried out in the same way as previous examples:

```
gcc -o printSatPosition printSatPosition.c -lglab -lm
```

And the program execution will also be very simple:

```
./printSatPosition
```

The output of this program will be shown on the screen. This may not be convenient if we are interested in plotting the results. Therefore, in UNIX/Linux systems it is possible to redirect the output to a new file by a simple modification in the execution line:

```
./printSatPosition > mySatData.txt
```

The contents of the `mySatData.txt` data file will be very similar to the following:

```
GPS 18 53935    18.000  16291872.348  17567370.495  11909671.366
GPS 18 53935   918.000  14881250.814  17123959.287  14175730.568
GPS 18 53935  1818.000  13224511.320  16667024.710  16203005.375
GPS 18 53935  2718.000  11342902.804  16233027.194  17957118.149
GPS 18 53935  3618.000   9265037.975  15854778.355  19408138.161
GPS 18 53935  4518.000   7026015.680  15560127.230  20531078.094
GPS 18 53935  5418.000   4666303.193  15370836.559  21306323.590
GPS 18 53935  6318.000   2230416.342  15301691.634  21719988.487
GPS 18 53935  7218.000   -234557.616  15359876.735  21764188.601
GPS 18 53935  8118.000  -2680541.239  15544645.497  21437228.414
```

As an additional exercise, we suggest trying to use the `graph.py` tool to plot the position results of satellite PRN18 with respect to time, extend this example to print the velocity also, and modify the program to accept both RINEX navigation file name and satellite PRN number as command-line parameters.

4. Complementary exercises: coordinates transformation

The following programs are some additional simple examples of using the `gLAB` libraries to perform coordinates transformations.

Complete the following steps:

(a) The program `Car2Esf.c` uses the `gLAB` libraries to perform the Cartesian to spherical coordinates transformation.

i. After compiling the source code `Car2Esf.c`,[6] transform the data file `cart.dat` to check the program is running.
Execute:

```
cat cart.dat | ./Car2Esf | less
```

ii. Edit the source code and identify the `gLAB` functions used by this program.

(b) The program `Car2Geo.c` uses the `gLAB` libraries to perform the Cartesian to ellipsoidal (geodetic) coordinates transformation.

[6] Execute `gcc -o Car2Esf Car2Esf.c -lglab -lm`, or `gcc -o Car2Esf Car2Esf.c -L. -lglab -lm` if the library `libglab.a` is in the working directory.

i. After compiling the source code `Car2Geo.c`, transform the data file `cart.dat` to check the program is running.
Execute:
```
cat cart.dat | ./Car2Geo | less
```

ii. Edit the source code and identify the `gLAB` functions used by this program.

(c) The program `Trs2Crs.c` uses the `gLAB` libraries to perform a terrestrial (TRS) to celestial (CRS) coordinates 'approximate' transformation. This program performs a simple rotation around the Z-axis by an angle of sidereal time magnitude.

i. After compiling the source code `Trs2Crs.c`, transform the data file `cart.dat` to check the program is running.
Execute:
```
cat cart.dat | ./Trs2Crs | less
```

ii. Edit the source code and identify the `gLAB` functions used by this program.

5. Complementary exercise: RINEX-2.10 writer

The program `Txt2Rnx.c` generates an observation RINEX-2.10 file from a text file in a columnar format.

(a) After compiling the source code `Txt2Rnx.c`, generate a RINEX-2.10 file from the contents of file `casa2910.95o.txt` to check the program is running.
Execute:
```
./Txt2Rnx casa2910.95.txt
```

(b) Edit the source code and identify the `gLAB` functions used by this program.

6. Complementary exercise: RINEX-3.0 to RINEX-2.10 converter

The program `Rinex3to2` converts an observation RINEX-3.0 file to RINEX-2.10 format.

(a) After compiling the source code `Rinex3to2.c`, convert the RINEX-3.0 file `iex3326f.12o` to RINEX-2.10 file format to check the program is running.
Execute:
```
cat iex3326f.12o | ./Rinex3to2 > iex3326f.12o_rnx210
```

(b) Edit the source code and identify the `gLAB` functions used by this program.

B. CD-ROM Content and Software Installation

B.1. CD-ROM Content

The CD-ROM provided with this book is organised in the following directories:

PROG This directory contains the programs associated with the book:

- The directory `src` contains the following subdirectories:
  - Directory `gLAB_src` contains the `gLAB` tool suite source files.
  - Directories `CC_src` and `F77_src` contain other source files of C and FORTRAN programs, respectively, used in the laboratory sessions.
- The directories `SESxy` contain configuration files and links to the programs for each session.

  For instance, directory `SES31` holds programs and configuration files associated with session 3.1.

  See the programs list with a brief explanation in Appendix C.
- The `Makefile` to install the software (on Linux OS).

FILES This directory contains a set of subdirectories with the data files for the different sessions. As in the previous case, they are labelled as `SESxy`.

HTML This directory contains the `html` files associated with session 2.2 describing the different standard GNSS data formats.

Notepad files: A set of `ntpd_sesXY.scr` files is provided to help the user write sentences of code. These files contain all the sentences to execute in the laboratory exercises and can be *copied* and *pasted* to the working terminal.

For those exercises where `gLAB` is executed using the GUI, the equivalent command-line `gLAB` sentence and the associated `gLAB` configuration file (i.e. `gLAB.cfg`), if needed, are also provided in the subdirectory **PROG**/`SESxy`.

Note: The Notepad `ntpd_sesXY.scr` *files are UNIX (i.e. Linux) scripts that automatically execute the entire session when they are executed.*

B.2. Software Installation

- *Minimum hardware requirements*: The minimum hardware requirements to execute `gLAB` properly are as follows:
  - 256 MB of RAM.
  - A CPU of at least 1 GHz.
  - 200 MB of hard disk free space.
  - A screen resolution of at least 1024×768 is recommended to display the GUI entirely. In order to cope with potential users using smaller screens, scrollbars can be displayed in the preferences button.
- *Software requirements*: `gLAB` and all the programs associated with this book can be compiled on LINUX (Ubuntu) or Windows operating systems. Nevertheless, the LINUX environment is recommended due to its higher performance.

  The following programs must be installed (all of them are available in both operating systems):
  - `awk` (or `gawk`)
  - C compiler
  - FORTRAN compiler
  - Python version 2.6 or newer.

To install the programs and associated data files, complete the following steps:

1. Copy the contents of the CD-ROM directories to the home directory.

   To save hard disk space, a link between the directory `FILES` in the CD-ROM and the home directory can be made instead of copying the data files.
2. Follow the instructions in the `README.txt` file.

C. Software Programs Associated with this Book

A list of the programs associated with this book is provided as follows.

Main programs

gLAB
The GNSS-LAB tool suite (`gLAB`) is an advanced interactive, educational, multipurpose package for processing and analysing GNSS data with High Accuracy Positioning capability. It can be executed by a Graphical User Interface (GUI) or in console mode with the Data Processing Core (DPC).

The DPC of `gLAB` is written in C while the GUI is written in the Python language.

graph.py
The plotting tool for data analysis. It is written in the Python language and is the core of the `gLAB` DAT.

Additional programs in C

Car2Esf.c
This program is an example of using `gLAB` as a GNSS library to perform a Cartesian to spherical coordinates transformation. It is provided as the complementary exercise 4a of session A.2.

Car2Geo.c
This program is an example of using `gLAB` as a GNSS library to perform a Cartesian to ellipsoidal (geodetic) coordinates transformation. It is provided as the complementary exercise 4b of session A.2.

FilterLS.c
This program builds the navigation equations system and applies the LS solver to compute the receiver's position. The output provides the user coordinates in Cartesian and geodetic systems, the position error regarding to reference coordinates, as well as the postfit residuals, according to the contents of Table A.6. It is given in session A.1 as an example of an elemental GNSS routine.

handleTime.c
This program is to illustrate the use of `gLAB` libraries for handling time. It is the driving example of exercise 1 in session A.2.

Model.c
This program computes the measurement modelling for SPP (see Chapter 5, Volume I). The output provides different model components according to the contents of Table A.5. It is given in session A.1 as an example of an elemental GNSS routine.

NavRinex2txt.c
This program reads a RINEX-2.1 navigation file and outputs the data in a columnar format, according to the contents of Table A.2. An additional file (`IonoParameters`) with the Klobuchar coefficients is also generated. It is given in session A.1 as an example of an elemental GNSS routine.

ObsRinex2txt.c
This program reads a RINEX-2.1 observation file and outputs the data in a columnar format, according to the contents of Table A.1. An additional file (`station.pos`) with the station coordinates is also generated. It is given in session A.1 as an example of an elemental GNSS routine.

PreProcess.c
This program performs data pre-checking and cycle-slip detection in the `ObsRinex2txt` output. It also performs carrier phase pre-alignment with code measurements. Single- and double-frequency cycle-slip detectors (see section 4.3, Volume I) and single frequency smoothing (see section 4.2.3, Volume I) are included. The contents of the output file are given in Table A.3. It is given in session A.1 as an example of an elemental GNSS routine.

printSatPosition.c
This program illustrates the use of `gLAB` libraries to read a broadcast RINEX navigation file and to compute the position of a given satellite during a full day. It is the driving example of exercise 3 in session A.2.

readRinexObs.c
This program illustrates the use of `gLAB` libraries to read a RINEX-2.10 observation file. It is the driving example of exercise 2 in session A.2.

Rinex3to2.c
This program illustrates the use of `gLAB` libraries to write a RINEX format file converter from RINEX-3.0 to RINEX-2.10 (observation files). It is provided as the complementary exercise 6 of session A.2.

Sat2Orb.c
This program computes the ECEF satellite coordinates and velocities at signal transmission time (section 3.3.1, Volume I) together with the satellite clock offsets and TGDs (sections 5.2 and 5.3, Volume I). The main input data are the broadcast navigation message and the code pseudorange measurements. Transmission time is computed using the pseudorange-based method (section 5.1.1, Volume I). It is given in session A.1 as an example of an elemental GNSS routine.

Trs2Crs.c
This program is to illustrate the use of `gLAB` libraries to perform a terrestrial (TRS) to celestial (CRS) coordinates 'approximate' transformation. It implements a simple approach based on rotation around the Z-axis by an angle of sidereal time magnitude. It is provided as the complementary exercise 4c of session A.2.

Txt2Rnx.c
This program illustrates the use of `gLAB` libraries to write a RINEX-2.10 file format from a text file in a columnar format. It is provided as the complementary exercise 5 of session A.2.

Additional programs in FORTRAN

Programs:

car2esf.f
Cartesian to spherical coordinates transformation.

car2geo.f
Cartesian to ellipsoidal (geodetic) coordinates transformation. This program implements the expressions of section B.1.2, Volume I.

geo2car.f
Ellipsoidal (geodetic) to Cartesian coordinates transformation. This program implements the expressions of section B.1.1, Volume I.

GLOeph2sp3.f
Implements the Glonass ICD for the satellite coordinate computation from the broadcast message (see section 3.3.2, Volume I). The orbit integrator is the Fourth-order Runge–Kutta (RK4) method. The force model includes the J2 term and solar–lunar accelerations. The program can compute the solar–lunar accelerations by itself or use the Glonass broadcast values. It can also be used to integrate GPS, Galileo or Beidou orbits from given initial position and velocity conditions.

Main subroutines included:
`sub_cal2cal.f`, `cal2doy.f`, `sub_nwsw2cal.f`, `sub_GLONASSephRK.f`, `sub_sid.f`, `sub_orb_deriv.f`, `sub_Moon_pos_GLO.f`, `sub_Sun_pos_GLO.f`, `sub_nsteffensen.f`

GPSbrd2rvorb.f
Reads the RINEX ephemeris file and writes satellite coordinate and osculating orbital elements.

Main subroutines included:
`sub_cal2dgpsws.f`, `sub_orbit.f`

GPSxyz.f
Very basic program associated with exercise 5(d)ii in session 5.2. It computes the satellite coordinates at a given time from the GPS broadcast message.

External subroutines:
`sub_orbit.f`

iono.f
Very basic program associated with exercise 5h in session 5.2. It computes the Klobuchar ionospheric correction at a given time from the GPS broadcast message.

External subroutines:
`sub_klob.f`

mapp_niell.f
Implements the mapping of Niell for the tropospheric model. See section 5.4.2.2.1, Volume I.

Main subroutines included:
`sub_nwsw2cal.f`

nwsw2cal.f
Computes the calendar date and Julian day from the GPS week and seconds of week (see section A.1.4, Volume I).

Main subroutines included:
`sub_nwsw2cal.f`

osc2rv.f
Satellite position and velocity computation from osculating orbital elements. This program implements the expressions of section C.2, Volume I.

out2sp3.f
Format converter from the `gLAB OUTPUT` message with the receiver coordinates to SP3 file format.

Main subroutines included:
`sub_nwsw2cal.f`, `sub_doy2cal.f`

rv2osc.f
Osculating orbital element computation from a given satellite position and velocity. This program implements the expressions of section C.1, Volume I.

trnfsp3n.f
Transforms SP3 ephemeris from the current (ITRF) frame to a new (e.g. NAD83) datum. This program is taken from: *http://igscb.jpl.nasa.gov/igscb/resource/tutorial/APPENDIX_IGS_ITRF.txt*

tropo.f
Very basic program associated with exercise 5g in session 5.2. It computes the tropospheric correction at a given time using the nominal (UNB3) tropospheric model.

External subroutines:
`sub_trpUNB3.f`

trs2crs.f
Terrestrial (TRS) to celestial (CRS) coordinates 'approximate' transformation. This program performs a simple rotation around the Z-axis by an angle of sidereal time magnitude.

Main subroutines included:
`sub_sid.f`

txt2sp3.f
Format converter from position and velocity in columnar text file format to SP3 format.

Main subroutines included:
`sub_doy2cal.f`, `sub_nwsw2cal.f`

Subroutines:

sub_klob.f
Implements the UNB3 model (see section 5.4.1.2.1, Volume I) to compute the Klobuchar ionospheric correction.

Main subroutines included:
`sub_car2geo.f`

sub_orbit.f
Implements the computation of GPS coordinates from the broadcast navigation message, according to the GPS ICD. See section 3.3.1, Volume I.

Main subroutines included:
`sub_nsteffensen.f`

sub_trpUNB3.f
Implements the UNB3 model (see section 5.4.2.1, Volume I) to compute the tropospheric correction.

Main subroutines included:
`sub_car2geo.f`

Other subroutines included in the programs

sub_cal2cal.f
Computes the Day of Year (DOY), GPS week, second of GPS week, Julian date and modified Julian date from calendar date (YY:MM:DD) and time (hh:mm:ss). This programs is based on the expressions of sections 3.1.1 and A.1.4 in Volume I.

sub_cal2doy.f
Computes the Day of Year from the calendar (YY:MM:DD) date. This program is based on the expressions of section 3.1.1, Volume I.

sub_cal2dgpsw.f
Computes the Day of Year (DOY), GPS week, second of GPS week and sidereal time from the calendar (YY:MM:DD) date and time (hh:mm:ss). It implements the expressions of sections 3.1.1 and A.1.4 in Volume I.

sub_car2geo.f
Cartesian to ellipsoidal (geodetic) coordinates transformation. This subroutine implements the expressions of section B.1.2, Volume I.

sub_doy2cal.f
Computes the calendar date (YY:MM:DD:hh:mm:sec), GPS week and seconds of GPS week from year, Day of Year and seconds of day. This program in based on the expressions of sections 3.1.1 and A.1.4 in Volume I.

sub_GLONASSephRK.f
This subroutine implements the fourth-order Runge–Kutta (RK4) algorithm of section 3.3.2.1 in Volume I and computes a table with the Glonass satellite coordinates from the broadcast navigation message.

Main subroutines included:
`sub_orb_deriv.f`

sub_Moon_pos_GLO.f
It implements the expressions of [GLONASS ICD, 1998] (some typographical errors in this document have been fixed). Computes Earth–Moon unit vector and distance in km for a given epoch (in Julian days).

sub_nsteffensen.f
Implements the Steffensen method to find the root of a nonlinear equation. It is a variant of the Newton–Raphson method with faster convergence.

sub_nwsw2cal.f
Computes the calendar date and Julian day from GPS week and seconds of week. It implements the expressions of sections 3.1.1 and A.1.4 in Volume I.

sub_orb_deriv.f
This is a subroutine of `sub_GLONASSephRK.f` implementing the function `X'=F[t,X(t)]` (see section 3.3.2.1 in Volume I).

sub_sid.f
Computes the sidereal time in Greenwich from the year and DoY (see sections 3.1.1 and A.2.5.2 in Volume I).

sub_Sun_pos_GLO.f
It implements the expressions of [GLONASS ICD, 1998] (some typographical errors in this document have been fixed). Computes Earth–Sun unit vector and distance in km for a given epoch (in Julian days).

D. Sources of Data Files Used in the Exercises

A list of the main data servers used to gather the data files for the exercises is provided as follows:

RINEX measurement files

GPS, Glonass, Galileo:

ftp://cddis.gsfc.nasa.gov/pub/gps/data/daily
ftp://cddis.gsfc.nasa.gov/pub/gps/data/l5test/rinex2/daily
ftp://www.ngs.noaa.gov/cors/rinex
ftp://data-out.unavco.org/pub/rinex/obs
ftp://igs.ensg.ign.fr/pub/igs/data/rinex3
http://igs.org/mgex

Examples:

ftp://cddis.gsfc.nasa.gov/pub/gps/data/daily/2003/301/03d/
asc13010.03d.Z
ftp://cddis.gsfc.nasa.gov/pub/gps/data/l5test/rinex2/
daily/2009/141/l5dt1410.09o.Z
ftp://www.ngs.noaa.gov/cors/rinex/2001/291/mhcb/
mhcb2910.01d.Z
ftp://data-out.unavco.org/pub/rinex/obs/2003/301/
amc23010.03d.Z

LEO satellite: GRACE-A

ftp://podaac.jpl.nasa.gov/GeodeticsGravity/grace/

Examples:

ftp://podaac.jpl.nasa.gov/GeodeticsGravity/grace/L1B/
JPL/RL01/2007/grace_1B_2007-03-21_01.tar.gz
ftp://podaac.jpl.nasa.gov/GeodeticsGravity/grace/L1B/
JPL/RL01/sw/GraceReadSW_L1_2008-03-20.tar.gz

RINEX navigation files

GPS:

ftp://cddis.gsfc.nasa.gov/pub/gps/data/daily

Example:

ftp://cddis.gsfc.nasa.gov/pub/gps/data/daily/2010/025/10n/brdc0250.10n.Z

Glonass:

ftp://cddis.gsfc.nasa.gov/pub/glonass/data/daily

Example:

ftp://cddis.gsfc.nasa.gov/pub/glonass/data/daily/2009/232/09g/brdc2320.09g.Z

Precise orbits and clocks [SP3 and CLK files]

ftp://cddis.gsfc.nasa.gov/pub/gps/products

Examples:

GPS Orbits and clocks [SP3 files]:

ftp://cddis.gsfc.nasa.gov/pub/gps/products/1568/igs15681.sp3.Z
ftp://cddis.gsfc.nasa.gov/pub/gps/products/1568/igr15681.sp3.Z
ftp://cddis.gsfc.nasa.gov/pub/gps/products/1568/igu15681_00.sp3.Z
ftp://cddis.gsfc.nasa.gov/pub/gps/products/1568/igu15681_12.sp3.Z
ftp://cddis.gsfc.nasa.gov/pub/gps/products/1568/emr15681.sp3.Z
ftp://cddis.gsfc.nasa.gov/pub/gps/products/1568/cod15681.eph.Z

GPS Clocks [CLK files]:

ftp://cddis.gsfc.nasa.gov/pub/gps/products/1568/igs15681.clk.Z
ftp://cddis.gsfc.nasa.gov/pub/gps/products/1568/igr15681.clk.Z
ftp://cddis.gsfc.nasa.gov/pub/gps/products/1568/cod15681.clk.Z
ftp://cddis.gsfc.nasa.gov/pub/gps/products/1568/igs15681.clk_30s.Z
ftp://cddis.gsfc.nasa.gov/pub/gps/products/1568/cod15681.clk_05s.Z

Other servers for GPS orbits and clocks [SP3 files]:

ftp://ftp.nga.mil/pub2/gps

Examples:

ftp://ftp.nga.mil/pub2/gps/apcpe/2010apc/apc15681.Z
ftp://ftp.nga.mil/pub2/gps/pedata/2010pe/nga15681.Z

Glonass Orbits and clocks:

ftp://cddis.gsfc.nasa.gov/pub/glonass/products

Examples:

ftp://cddis.gsfc.nasa.gov/pub/glonass/products/1545/
iac15454.sp3.Z
ftp://cddis.gsfc.nasa.gov/pub/glonass/products/1545/
igl15454.sp3.Z
ftp://cddis.gsfc.nasa.gov/pub/glonass/products/1545/
mcc15454.sp3.Z

ANTEX files

ftp://igscb.jpl.nasa.gov/igscb/station/general/pcv_archive

Example:

ftp://igscb.jpl.nasa.gov/igscb/station/general/pcv_archive/
igs05_1525.atx

Troposphere (ZTD files)

ftp://cddis.gsfc.nasa.gov/pub/gps/products/troposphere

Example:

ftp://cddis.gsfc.nasa.gov/pub/gps/products/troposphere/
new/2009/181/brus1810.09zpd.gz

Ionosphere (IONEX files)

ftp://cddis.gsfc.nasa.gov/gps/products/ionex

Examples:

ftp://cddis.gsfc.nasa.gov/gps/products/ionex/2009/181/
igsg1810.09i.Z
ftp://cddis.gsfc.nasa.gov/gps/products/ionex/2009/181/
igrg1810.09i.Z
ftp://cddis.gsfc.nasa.gov/gps/products/ionex/2009/181/
upcg1810.09i.Z

P1C1 and P1P2 DCB files

ftp://ftp.unibe.ch/aiub/CODE

Examples:

ftp://ftp.unibe.ch/aiub/CODE/2010/P1C11010.DCB.Z
ftp://ftp.unibe.ch/aiub/CODE/2010/P1P21010.DCB.Z

SINEX files

ftp://cddis.gsfc.nasa.gov/pub/gps/products

Examples:

ftp://cddis.gsfc.nasa.gov/pub/gps/products/1556/
igs09P1556.snx.Z

ITRF files

http://itrf.ensg.ign.fr/trans_para.php

Examples:

ftp://igscb.jpl.nasa.gov/pub/resource/tutorial/
APPENDIX_IGS_ITRF.txt
http://itrf.ensg.ign.fr/doc_ITRF/
Transfo-ITRF2008_ITRFs.txt

Other servers: GIPSY_PRODUCTS

ftp://sideshow.jpl.nasa.gov/pub/gipsy_products/
gipsy_params/GPS_Receiver_Types.gz
ftp://sideshow.jpl.nasa.gov/pub/gipsy_products/IERSB/
PRN_GPS.gz
ftp://sideshow.jpl.nasa.gov/pub/jpligsac/1062/
jpl10621.sp3.Z
ftp://sideshow.jpl.nasa.gov/pub/gipsy_products/2000/
orbits/2000-05-15.eci.Z

List of Acronyms

AGW	Atmospheric Gravity Waves
ANTEX	ANTenna EXchange format
APC	Antenna Phase Centre
ARP	Antenna Reference Point
ASCII	American Standard Code for Information Interchange
A/S	Anti-Spoofing
C/A	Coarse/Acquisition
CDDIS	Crustal Dynamics Data Information System
CODE	Centre for Orbit Determination in Europe
CRF	Celestial Reference Frame
CRS	Conventional Celestial Reference System
CS	Cycle Slip
DAT	Data Analysis Tool
DCB	Differential Code Bias
DLL	Dynamic Link Library
DLR	Deutsches Zentrum für Luft- und Raumfahrt
DOP	Dilution Of Precision
DoY	Day of Year
DPC	Data Processing Core
ECEF	Earth-Centred, Earth-Fixed
ECI	Earth-Centred Inertial
EMR	Energy Mines and Resources
ENU	East North Up
ESA	European Space Agency
gAGE	Research group of Astronomy and Geomatics

GAL	GALileo Satellite Identifier
GDOP	Geometric Dilution Of Precision
GEO	GEOstationary Satellite Identifier
GIPSY	GPS Inferred Positioning SYstem
gLAB	GNSS-LAB tool suite
GLO	Glonass Satellite Identifier
Glonass	GLObal NAvigation Satellite System
GNSS	Global Navigation Satellite System
GNU	GNU's Not Unix
GPS	Global Positioning System
GRACE	Gravity Recovery and Climate Experiment
GRAPHIC	Group and Phase Ionospheric Calibration
GFree	Geometry Free
GUI	Graphical User Interface
HDOP	Horizontal Dilution Of Precision
HTML	HyperText Markup Language
IAC	Information Analytical Centre
ICD	Interface Control Document
IFree	Ionosphere Free
IGL	IGS Glonass orbit products
IGRF	International Geomagnetic Reference Field
IGS	International GNSS Service
IGST	IGS Time
IODE	Issue Of Data Ephemerides
IODN	Issue Of Data Navigation
IONEX	IONosphere map EXchange format
IRI	International Reference Ionosphere
ITRF	International Terrestrial Reference Frame
JPL	Jet Propulsion Laboratory
LEO	Low Earth Orbit
LS	Least Squares

LSTIDs Large-Scale TIDs

MC Satellite Mass Centre

MCC Mission Control Centre

MJD Modified Julian Day

MM Monument Marker

MSDOS Microsoft Disk Operating System

MSTID Medium-Scale Travelling Ionospheric Disturbance

MSTIDs Medium-Scale Travelling Ionospheric Disturbances

MW Melbourne–Wübbena

NANU Notice Advisory to NAVSTAR Users

NASA National Aeronautics and Space Administration

NEU North East Up

NGA National Geospatial-Intelligence Agency

NSE Navigation System Error

OS Operating System

PCO Phase Centre Offset

PCV Phase Centre Variation

PDOP Precision Dilution Of Precision

PNG Portable Network Graphics

PPP Precise Point Positioning

PRN Pseudo-Random Noise

PVT Position, Velocity, Time

PZ-90 Parametry Zemli 1990 (Parameters of Earth 1990)

RINEX Receiver INdependent EXchange format

RK4 Fourth-order Runge–Kutta

RMS Root Mean Square

RO Radio Occultation

S/A Selective Availability

SBAS Satellite-Based Augmentation System

SED Storm Enhancement Density

SINEX Solution (Software/technique) INdependent EXchange format

SIS	Signal In Space
SISRE	Signal In Space Range Error
SOHO	SOlar Helioscopic Observatory
SP3	Standard Product #3
SPP	Standard Point Positioning
SPS	Standard Positioning Service
SSI	Signal Strength Indicator
STEC	Slant Total Electron Content
STROP	Slant TROPospheric delay
SU	Soviet Union
SV	Space Vehicle
SVN	Space Vehicle Number
TDOP	Time Dilution Of Precision
TEC	Total Electron Content
TECU	Total Electron Content Unit
TGD	Total Group Delay
TID	Travelling Ionospheric Disturbance
TRF	Terrestrial Reference Frame
TRS	Conventional Terrestrial Reference System
TUM	Technische Universität München
UPC	Technical University of Catalonia
USNO	United States Naval Observatory
UTC	Coordinated Universal Time
VDOP	Vertical Dilution Of Precision
WGS-84	World Geodetic System 84
ZTD	Zenith Tropospheric Delay

Bibliography

[Borre et al., 2006] Borre, K., Akos, D., Bertelsen, N., Rinder, P. and Jensen, S. H., 2006. A Software-Defined Gps And Galileo Receiver: A Single-Frequency Approach. Birkhauser Verlag, Boston, MA.

[GLONASS ICD, 1998] GLONASS ICD, 1998. Technical report, Moscow. v.4.0.

[Goldstein, 2010] Goldstein, D., 2010. Global Positioning Systems Wing: Request for Feedback on GPS IIR-20 (SVN-49) Mitigation Options.

[Hernández-Pajares, 2004] Hernández-Pajares, M., 2004. IGS Ionosphere WG Status Report: Performance of IGS Ionosphere TEC Maps – Position Paper. In: IGS Workshop and Symposium 2004 "Celebrating a Decade of the International GPS Service", Bern, Switzerland.

[HJS, 2006] Hernández-Pajares, M. and Juan, M. and Sanz, J., 2006. Medium-Scale Traveling Ionospheric Disturbances Affecting GPS Measurements: Spatial and Temporal Analysis. *Journal of Geophysical Research* **111**, pp. 1–13.

[HJS, 2007] Hernández-Pajares, M. and Juan, M. and Sanz, J., 2007. Second-Order Ionospheric Term in GPS: Implementation and Impact on Geodetic Estimates. *Journal of Geophysical Research* **112**, pp. 1–16.

[HJS et al., 2005] Hernández-Pajares, M. and Juan, M. and Sanz, J., Farnworth, R. and Soley, S., 2005. EGNOS Test Bed Ionospheric Corrections under the October and November 2003 Storms. *IEEE Transactions on Geoscience and Remote Sensing* **43**(10), pp. 2283–2293.

[Kenneth et al., 2008] Kenneth, L., Ray, J. and Beard, R., 2008. Characterization of Periodic Variations in the GPS Satellite Clocks. *GPS Solutions* **12**(3), pp. 211–225.

[Marechal et al., 2007] Marechal, M., Suard, N. and Chinchilla, D., 2007. EGNOS Protection on Two GPS Clock Events. *Proceedings of the 20th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS 2007)*. pp. 919–926.

[Montenbruck and Eberhard, 2005] Montenbruck, O. and Eberhard, G., 2005. Satellite Orbits. Models, methods, applications. Springer, Germany.

[Ramos-Bosch et al., 2010] Ramos-Bosch, P., Rovira-Garcia, A., Hernández-Pajares, M., Juan, J. and Sanz, J., 2010. gLAB Software User Manual, EDUNAV-SUM-gAGE/UPC, Issue 1.6. gAGE/UPC.

[Sanz et al., 2010] Sanz, J., Juan, J. and Hernández-Pajares, M., 2010. *GNSS Data Processing laboratory exercises*. 4th ESA GNSS Summer School., Slettestrand (Aalborg), Denmark. *http://www.gage.upc.edu/tutorials*.

[Sanz et al., 2012a] Sanz, J., Juan, J. and Hernández-Pajares, M., 2012a. *Analysis of propagation effects from GNSS observables based on laboratory exercises*. Course on Propagation effects, channel models and related error sources on GNSS, ESA/ESAC., Madrid, Spain. *http://www.gage.upc.edu/tutorials*.

[Sanz et al., 2012b] Sanz, J., Rovira-Garcia, A., Hernández-Pajares, M., Juan, M., Ventura-Traveset, J., Lopez-Echazarreta, C. and Hein, G., 2012b. The ESA/UPC GNSS-Lab Tool (gLAB): An Advanced Educational and Professional Package for GNSS Data Processing and Analysis. In: *Proceedings of Toulouse Space Show 2012, 4th International Conference on Space Applications*, *Toulouse, France*.

[Schaer, S. and Steingenberger, P., 2006] Schaer, S. and Steingenberger, P., 2006. Determination and Use of GPS Differential Code Bias Values. *http://nng.esoc.esa.de/ws2006/REPR2.pdf*.

[Seeber, 1993] Seeber, G., 1993. Satellite Geodesy: Foundations, Methods, and Applications. Walter de Gruyter & Co., Berlin, Germany.

[Springer and Dilssner, 2009] Springer, T. and Dilssner, F., 2009. SVN49 and Other GPS Anomalies. *Inside GNSS* **4**(4), pp. 32–36.

[Strang and Borre, 1997] Strang, G. and Borre, K., 1997. Linear Algebra, Geodesy, and GPS. Wellesley-Cambridge Press, Wellesley MA.

[Thoelert et al., 2009] Thoelert, S., Erker, S., Montenbruck, O., Hauschild, A. and Meurer, M., 2009. GPS SVN49 – L1 Anomaly Analysis Based on Measurements with a High Gain Antenna. In: *Proceedings of the 4th European Workshop on GNSS Signals and Signal Processing, 10–11 December 2009,*, *Oberpfaffenhofen, Germany*.

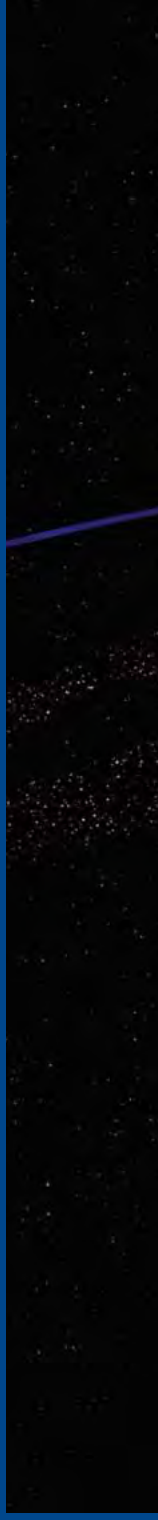
[Tsyganenko, 2003] Tsyganenko, N., 2003. A Set of FORTRAN Subroutines for Computations of the Geomagnetic Field in the Earth's Magnetosphere (Geopack). University Space Research Association, Columbia, MD, USA.

[Warren and Raquet, 2003] Warren, D. and Raquet, J., 2003. Broadcast vs. Precise GPS Ephemerides: A Historical Perspective. *GPS Solutions* **7**, pp. 151–156.

Index

About the authors

Jaume Sanz Subirana, PhD, José Miguel Juan Zornoza, PhD, and Manuel Hernández-Pajares, PhD, are Professors at the Technical University of Catalonia (UPC). They created the Research Group of Astronomy and Geomatics (gAGE) at the UPC in 1987. Their current research interests are in the areas of GNSS data processing algorithms, GNSS ionospheric sounding, satellite and ground-based augmentation systems (SBAS and GBAS), and high-accuracy GNSS navigation. This last is one of the main research topics of the gAGE/UPC group, where new algorithms for wide-area high-accuracy navigation are being developed and tested, leading to the Wide Area RTK (WARTK) and the Fast-Precise Point Positioning (Fast-PPP) techniques. The authors share several international patents on these techniques, founded by ESA. As a result of the gAGE/UPC research group's activities in GNSS, in 2009 they founded the spin-off company gAGE-NAV S.L.



ESA Member States

Austria
Belgium
Czech Republic
Denmark
Finland
France
Germany
Greece
Ireland
Italy
Luxembourg
Netherlands
Norway
Poland
Portugal
Romania
Spain
Sweden
Switzerland
United Kingdom

An ESA Communications Production