Spotlight on Bernese GNSS Software

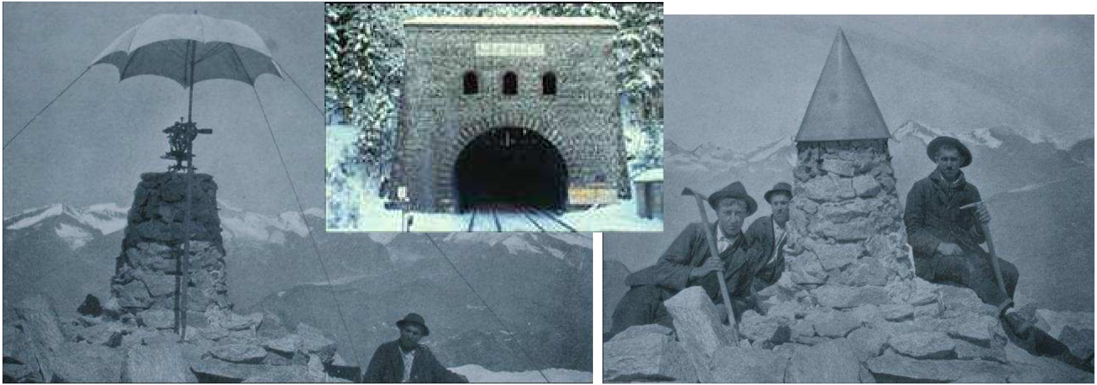
Prof. Dr. Rolf Dach
and the BSW-development team

Astronomical Institute, University of Bern, Switzerland

Reference Frames in Practice
Technical Seminar in the frame of XXVII FIG Congress
10.–11. September 2022, Warsaw, Poland

Building the Lötschberg tunnel in Switzerland

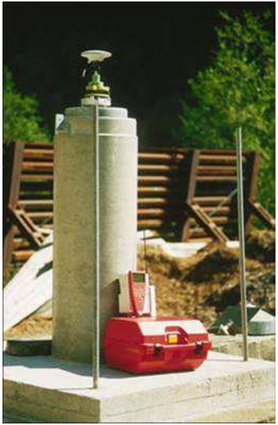
Before the construction of the tunnel starts a geodetic connection of the portals is needed.

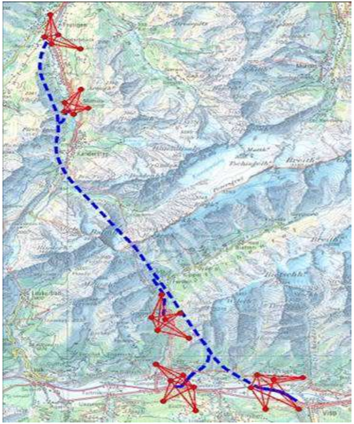


at that time: triangulations over several months have been necessary

Building the Lötschberg tunnel in Switzerland

Before the construction of the tunnel starts a geodetic connection of the portals is needed.

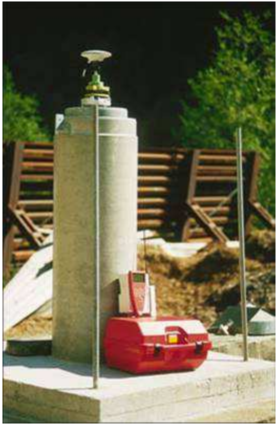


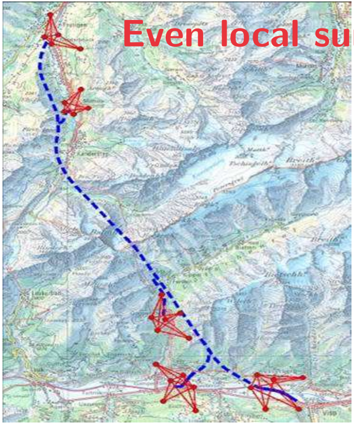


today: few hours of GNSS-measurements are sufficient

Building the Lötschberg tunnel in Switzerland

Before the construction of the tunnel starts a geodetic connection of the portals is needed.

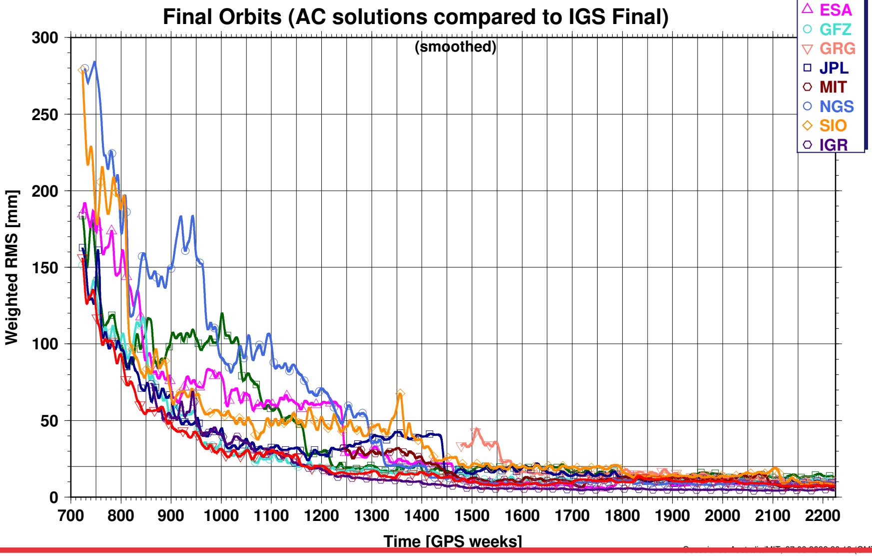




Even local surveying tasks are "global".

today: few hours of GNSS-measurements are sufficient

IGS Satellite Orbits



The CODE Analysis Center

- CODE, Center for Orbit Determination in Europe, is one of at present ten Analysis Centers of the IGS. CODE is formed as a joint venture of
  - the Astronomisches Institut, Universität Bern (AIUB),
  - the Bundesamt für Landestopographie (swisstopo),
  - the Bundesamt für Kartographie und Geodäsie (BKG), and
  - the Institut für Astronomische und Physikalische Geodäsie of TU München (IAPG, TUM).

AIUB

Bundesamt für Kartographie und Geodäsie

Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Technische Universität München

The CODE Analysis Center

- CODE is located at the AIUB in Bern.
- CODE started operating on 21 June 1992.

The CODE Analysis Center

- CODE is located at the AIUB in Bern.
- CODE started operating on 21 June 1992.
- Initially about 20, today more than 250 stations are processed daily. All results are generated using the *Bernese GNSS Software*.

The CODE Analysis Center

- CODE is located at the AIUB in Bern.
- CODE started operating on 21 June 1992.
- Initially about 20, today more than 250 stations are processed daily. All results are generated using the *Bernese GNSS Software*.
- CODE provides products for the repro, final, rapid, and ultra-rapid IGS products.
- CODE started with a rigorously combined GPS/GLONASS analysis in May 2003.
  Meanwhile also other IGS analysis centers join this a strategy for their final products: ESOC, GFZ, GRGS, NRCan, Wuhan.

The CODE Analysis Center

- CODE is located at the AIUB in Bern.
- CODE started operating on 21 June 1992.
- Initially about 20, today more than 250 stations are processed daily. All results are generated using the *Bernese GNSS Software*.
- CODE provides products for the repro, final, rapid, and ultra-rapid IGS products.
- CODE started with a rigorously combined GPS/GLONASS analysis in May 2003.
- Since 2012 CODE also contributes to the IGS MGEX project with a fully combined five-system solution: GPS+GLONASS+ Galileo+BeiDou+QZSS (currently 91 satellites).

The CODE Analysis Center

- CODE is located at the AIUB in Bern.
- CODE started operating on 21 June 1992.
- Initially about 20, today more than 250 stations are processed daily. All results are generated using the *Bernese GNSS Software*.
- CODE provides products for the repro, final, rapid, and ultra-rapid IGS products.
- CODE started with a rigorously combined GPS/GLONASS analysis in May 2003.
- Since September 2019, CODE – as the first AC – includes Galileo in the operational rapid and ultra-rapid solutions for the IGS, resulting in a GPS+GLONASS+Galileo solution.

Outline

The Bernese GNSS Software

The *Bernese GNSS Software* is

- a scientific software package
- for multi-GNSS data analysis
- with highest accuracy requirements
- in regional to global scale networks.

It is developed, maintained and used at the Astronomical Institute of the University of Bern since many years.

The *Bernese GNSS Software* is online at http://www.bernese.unibe.ch .

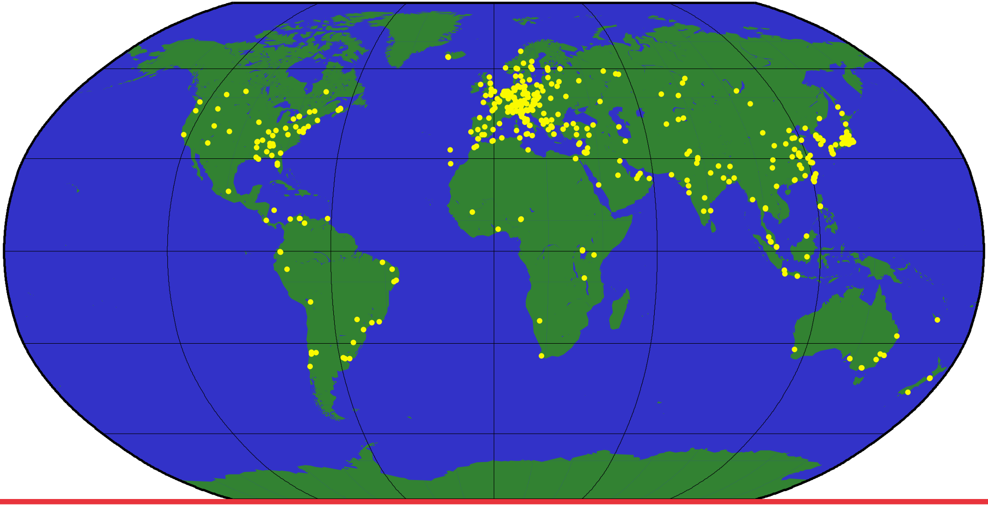
Bernese GNSS Software: typical applications

The Bernese GNSS Software is particularly well suited for:

- rapid processing of small-size surveys (static as well as kinematic stations — even LEOs)
- automatic processing of permanent networks (BPE: Bernese Processing Engine),
- combination of different receiver and antenna types, taking receiver biases and satellite antenna phase center variations into account,
- rigorosly combined processing of GPS, GLONASS, Galileo, BDS, and QZSS observations,
- ambiguity resolution on long baselines (2000 km and longer),
- precise point positioning (including ambigity resolution),
- generation of minimum constraint network solutions,
- ionosphere and troposphere monitoring,
- clock offset estimation and time transfer,
- orbit determination and estimation of Earth orientation parameters.
- ...

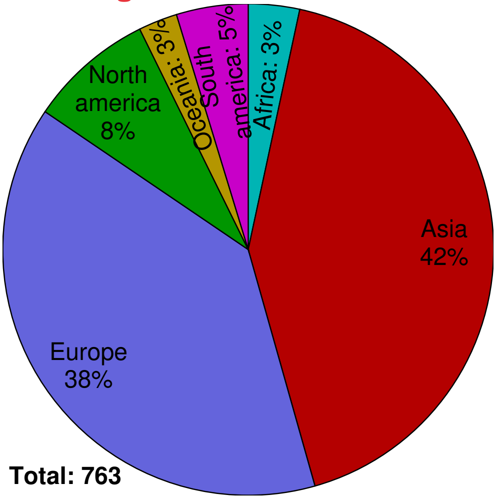
Bernese GNSS Software: users

Distribution of institutions using the Bernese GNSS Software

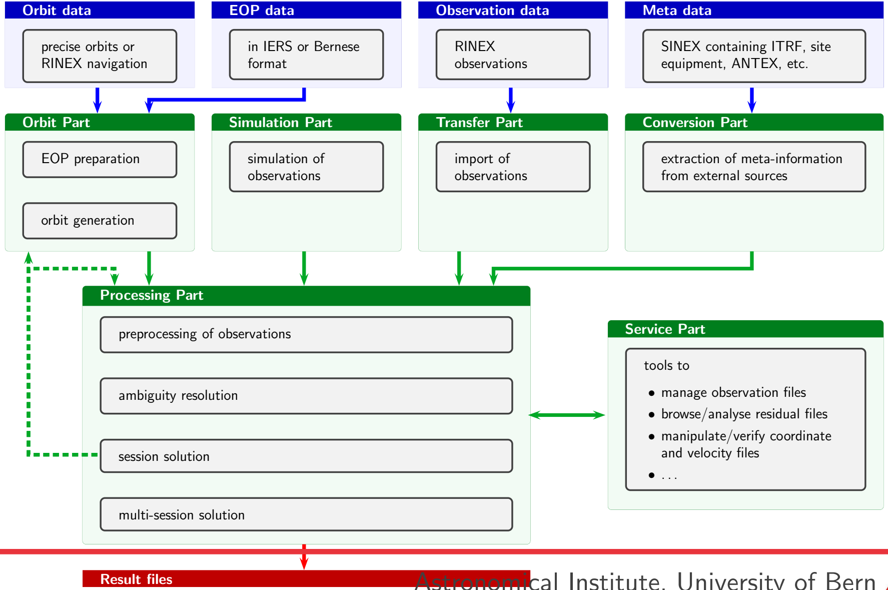


Bernese GNSS Software: users

Distribution of institutions using the Bernese GNSS Software



Bernese GNSS Software: Program Overview



Bernese GNSS Software: Program Overview

- Transfer Part:
  Programs for generating files in the Bernese format from RINEX. Furthermore, this part also contains a set of tools to cut/concatenate and to manipulate RINEX files.

- Conversion Part:
  Programs to extract external information necessary for the processing from international to Bernese specific formats
  (e.g., coordinates and velocities from ITRF in SINEX format, ANTEX, Bias SINEX).

- Orbit Part:
  Programs for generation of a source-independent orbit representation (standard orbits), to update orbits, generate orbits in precise orbit format, compare orbits, etc. The Earth orientation related tools are included in this part too.

Bernese GNSS Software: Program Overview

- Processing Part:
  Programs for receiver clock synchronization, code and phase pre-processing, ambiguity resolution, parameter estimation based on GNSS observations (pgm. GPSEST) and on the superposition of normal equations (pgm. ADDNEQ2).

- Simulation Part:
  Program to generate simulated GNSS observations (code and/or phase, one or two frequencies) based on statistical information (RMS for observations, biases, cycle slips).

- Service Part:
  A collection of useful tools to edit/browse/manipulate binary data files, compare coordinate sets, display residuals, etc. A set of programs to convert binary files to ASCII and vice versa belong to the service part, too.

Bernese GNSS Software: Directory structure

Main Directories:

- $C=BERN54
  Program area with source code, executables, and supporting files

- $U=GPSUSER54
  User area with user–specific settings for interactive processing and the BPE configurations running for this user

- $T=GPSTEMP54
  Temporary file area for the users BPE processing

Bernese GNSS Software: Directory structure

Main Directories:

- `$D=GPSDATA/DATAPOOL`
  Local database with all external files needed for GNSS data processing
- `$P=GPSDATA/CAMPAIGN54`
  Campaign area where the processing with the Bernese GNSS Software takes place
- `$S=GPSDATA/SAVEDISK`
  Product archive containing all GNSS derived products for further analysis

Bernese GNSS Software: Directory structure

Main Directories:

- `$D=GPSDATA/DATAPOOL`
  Local database with all external files needed for GNSS data processing
- `$P=GPSDATA/CAMPAIGN54`
  Campaign area where the processing with the Bernese GNSS Software takes place
- `$S=GPSDATA/SAVEDISK`
  Product archive containing all GNSS derived products for further analysis

Dataflow realized in the processing examples of Version 5.4:

DATAPOOL $\rightarrow$ **CAMPAIGN** $\rightarrow$ **SAVEDISK**

Questions before starting the processing

Defining what to do::

Questions before starting the processing

Defining what to do::

1. Which stations I need to process?
   Select also reference frame site (e.g., from the IGS network).

Questions before starting the processing

Defining what to do::

1. Which stations I need to process?
   Select also reference frame site (e.g., from the IGS network).

2. Which GNSS shall be considered?

3. Which external orbit/ERP/clock/bias prodcuts shall be used?

Questions before starting the processing

Defining what to do::

1. Which stations I need to process?
   Select also reference frame site (e.g., from the IGS network).

2. Which GNSS shall be considered?

3. Which external orbit/ERP/clock/bias prodcuts shall be used?

4. Decide on consistent IERS modelling, antenna model, and reference frame.

Questions before starting the processing

Defining what to do::

1. Which stations I need to process?
   Select also reference frame site (e.g., from the IGS network).
2. Which GNSS shall be considered?
3. Which external orbit/ERP/clock/bias prodcuts shall be used?
4. Decide on consistent IERS modelling, antenna model, and reference frame.

- Populate the DATAPOOL with the necessary files.

Bernese GNSS Software: Processing steps

Bernese GNSS Software: Processing steps

1. Data transfer: copy data into the campaign area

Bernese GNSS Software: Processing steps

1. Data transfer: copy data into the campaign area

2. Import observation data into Bernese format

Bernese GNSS Software: Processing steps

1. Data transfer: copy data into the campaign area

2. Import observation data into Bernese format

3. Prepare EOP and orbit information

Bernese GNSS Software: Processing steps

1. Data transfer: copy data into the campaign area
2. Import observation data into Bernese format
3. Prepare EOP and orbit information
4. Data preprocessing: cycle slip detection and correction; outlier rejection

Bernese GNSS Software: Processing steps

1. Data transfer: copy data into the campaign area
2. Import observation data into Bernese format
3. Prepare EOP and orbit information
4. Data preprocessing: cycle slip detection and correction; outlier rejection
5. Make a first network solution (real-valued ambiguities)

Bernese GNSS Software: Processing steps

1. Data transfer: copy data into the campaign area
2. Import observation data into Bernese format
3. Prepare EOP and orbit information
4. Data preprocessing: cycle slip detection and correction; outlier rejection
5. Make a first network solution (real-valued ambiguities)
6. Resolve ambiguities

Bernese GNSS Software: Processing steps

1. Data transfer: copy data into the campaign area
2. Import observation data into Bernese format
3. Prepare EOP and orbit information
4. Data preprocessing: cycle slip detection and correction; outlier rejection
5. Make a first network solution (real-valued ambiguities)
6. Resolve ambiguities
7. Create a normal equation containing all relevant parameters

Bernese GNSS Software: Processing steps

1. Data transfer: copy data into the campaign area
2. Import observation data into Bernese format
3. Prepare EOP and orbit information
4. Data preprocessing: cycle slip detection and correction; outlier rejection
5. Make a first network solution (real-valued ambiguities)
6. Resolve ambiguities
7. Create a normal equation containing all relevant parameters
8. NEQ-based single- or multi-session solution

Bernese Processing Engine

- The implementation of these steps for an automated processing is done in the frame of a BPE - Bernese Processing Engine.

Bernese Processing Engine

- The implementation of these steps for an automated processing is done in the frame of a BPE - Bernese Processing Engine.
- The BPE needs to know
  - what is to do: user scripts
  - the order of running the scripts (dependencies)
  - where a script can be started (CPU)

Bernese Processing Engine

- The implementation of these steps for an automated processing is done in the frame of a BPE - Bernese Processing Engine.
- The BPE needs to know
  - what is to do: user scripts
  - the order of running the scripts (dependencies)
  - where a script can be started (CPU)
- Process Control File (PCF) is the way how this information is implemented.

Bernese Processing Engine

- The implementation of these steps for an automated processing is done in the frame of a BPE - Bernese Processing Engine.
- The BPE needs to know
  - what is to do: user scripts
  - the order of running the scripts (dependencies)
  - where a script can be started (CPU)
- Process Control File (PCF) is the way how this information is implemented.
- An example of such a PCF looks like:

Bernese Processing Engine

```
PID  SCRIPT    OPT_DIR   PARAMETERS
#
# Copy required files
# -------------------
001  R2S_COP   R2S_GEN   CPU=ANY
011  RNX_COP   R2S_GEN   CPU=ANY; WAIT=001;  NEXTJOB= 101 999
#
# Prepare the pole and orbit information
# --------------------------------------
101  POLUPD    R2S_GEN   CPU=ANY; WAIT=001
112  ORBGEN    R2S_GEN   CPU=ANY; WAIT=101 111
#
# Preprocess, convert, and synchronize observation data
# -----------------------------------------------------
221  RXOBV3    R2S_GEN   CPU=ANY; WAIT=011
231  CODSPP    R2S_GEN   CPU=ANY; WAIT=112 221
#
# Form baselines and pre-process phase data (incl. residual screening)
# --------------------------------------------------------------------
302  SNGDIF    R2S_GEN   CPU=ANY; WAIT=231
311  MAUPRP    R2S_GEN   CPU=ANY; WAIT=302
321  GPSEDT    R2S_EDT   CPU=ANY; WAIT=311
341  ADDNEQ2   R2S_GEN   CPU=ANY; WAIT=331
...
```

Bernese Processing Engine

```
PID  SCRIPT    OPT_DIR   PARAMETERS
#
# Copy required files
# -------------------
001  R2S_COP   R2S_GEN   CPU=ANY
011  RNX_COP   R2S_GEN   CPU=ANY; WAIT=001;  NEXTJOB= 101 999
#
# Prepare the pole and orbit information
# --------------------------------------
101  POLUPD    R2S_GEN   CPU=ANY; WAIT=001
112  ORBGEN    R2S_GEN   CPU=ANY; WAIT=101 111
#
# Preprocess, convert, and synchronize observation data
# -----------------------------------------------------
221  RXOBV3    R2S_GEN   CPU=ANY; WAIT=011
231  CODSPP    R2S_GEN   CPU=ANY; WAIT=112 221
#
# Form baselines and pre-process phase data (incl. residual screening)
# --------------------------------------------------------------------
302  SNGDIF    R2S_GEN   CPU=ANY; WAIT=231
311  MAUPRP    R2S_GEN   CPU=ANY; WAIT=302
321  GPSEDT    R2S_EDT   CPU=ANY; WAIT=311
341  ADDNEQ2   R2S_GEN   CPU=ANY; WAIT=331
...
```

1

Bernese Processing Engine

```
PID  SCRIPT    OPT_DIR   PARAMETERS
#
# Copy required files
# -------------------
001  R2S_COP   R2S_GEN   CPU=ANY
011  RNX_COP   R2S_GEN   CPU=ANY; WAIT=001;  NEXTJOB= 101 999
#
# Prepare the pole and orbit information
# --------------------------------------
101  POLUPD    R2S_GEN   CPU=ANY; WAIT=001
112  ORBGEN    R2S_GEN   CPU=ANY; WAIT=101 111
#
# Preprocess, convert, and synchronize observation data
# -----------------------------------------------------
221  RXOBV3    R2S_GEN   CPU=ANY; WAIT=011
231  CODSPP    R2S_GEN   CPU=ANY; WAIT=112 221
#
# Form baselines and pre-process phase data (incl. residual screening)
# --------------------------------------------------------------------
302  SNGDIF    R2S_GEN   CPU=ANY; WAIT=231
311  MAUPRP    R2S_GEN   CPU=ANY; WAIT=302
321  GPSEDT    R2S_EDT   CPU=ANY; WAIT=311
341  ADDNEQ2   R2S_GEN   CPU=ANY; WAIT=331
...
```

1

2

Bernese Processing Engine

```
PID  SCRIPT    OPT_DIR   PARAMETERS
#
# Copy required files
# -------------------
001  R2S_COP   R2S_GEN   CPU=ANY
011  RNX_COP   R2S_GEN   CPU=ANY; WAIT=001;  NEXTJOB= 101 999
#
# Prepare the pole and orbit information
# --------------------------------------
101  POLUPD    R2S_GEN   CPU=ANY; WAIT=001
112  ORBGEN    R2S_GEN   CPU=ANY; WAIT=101 111
#
# Preprocess, convert, and synchronize observation data
# -----------------------------------------------------
221  RXOBV3    R2S_GEN   CPU=ANY; WAIT=011
231  CODSPP    R2S_GEN   CPU=ANY; WAIT=112 221
#
# Form baselines and pre-process phase data (incl. residual screening)
# ---------------------------------------------------------------------
302  SNGDIF    R2S_GEN   CPU=ANY; WAIT=231
311  MAUPRP    R2S_GEN   CPU=ANY; WAIT=302
321  GPSEDT    R2S_EDT   CPU=ANY; WAIT=311
341  ADDNEQ2   R2S_GEN   CPU=ANY; WAIT=331
...
```

1

3

2

Bernese Processing Engine

```
PID  SCRIPT    OPT_DIR   PARAMETERS
#
# Copy required files
# -------------------
001  R2S_COP   R2S_GEN   CPU=ANY
011  RNX_COP   R2S_GEN   CPU=ANY; WAIT=001;  NEXTJOB= 101 999
#
# Prepare the pole and orbit information
# --------------------------------------
101  POLUPD    R2S_GEN   CPU=ANY; WAIT=001
112  ORBGEN    R2S_GEN   CPU=ANY; WAIT=101 111
#
# Preprocess, convert, and synchronize observation data
# -----------------------------------------------------
221  RXOBV3    R2S_GEN   CPU=ANY; WAIT=011
231  CODSPP    R2S_GEN   CPU=ANY; WAIT=112 221
#
# Form baselines and pre-process phase data (incl. residual screening)
# --------------------------------------------------------------------
302  SNGDIF    R2S_GEN   CPU=ANY; WAIT=231
311  MAUPRP    R2S_GEN   CPU=ANY; WAIT=302
321  GPSEDT    R2S_EDT   CPU=ANY; WAIT=311
341  ADDNEQ2   R2S_GEN   CPU=ANY; WAIT=331
...
```

1

3

2

4

Bernese Processing Engine

```
PID  SCRIPT    OPT_DIR   PARAMETERS
#
# Copy required files
# -------------------
001  R2S_COP   R2S_GEN   CPU=ANY
011  RNX_COP   R2S_GEN   CPU=ANY; WAIT=001;  NEXTJOB= 101 999
#
# Prepare the pole and orbit information
# --------------------------------------
101  POLUPD    R2S_GEN   CPU=ANY; WAIT=001
112  ORBGEN    R2S_GEN   CPU=ANY; WAIT=101 111
#
# Preprocess, convert, and synchronize observation data
# -----------------------------------------------------
221  RXOBV3    R2S_GEN   CPU=ANY; WAIT=011
231  CODSPP    R2S_GEN   CPU=ANY; WAIT=112 221
#
# Form baselines and pre-process phase data (incl. residual screening)
# --------------------------------------------------------------------
302  SNGDIF    R2S_GEN   CPU=ANY; WAIT=231
311  MAUPRP    R2S_GEN   CPU=ANY; WAIT=302
321  GPSEDT    R2S_EDT   CPU=ANY; WAIT=311
341  ADDNEQ2   R2S_GEN   CPU=ANY; WAIT=331
...
```

1

3

2

4

5

Bernese Processing Engine

```
PID  SCRIPT    OPT_DIR   PARAMETERS
...
#
# Resolve phase ambiguities
# -------------------------
411  GNSAMBAP  R2S_AMB   CPU=ANY; WAIT=401
412  GNSAMB_P  R2S_AMB   CPU=ANY; WAIT=411;  PARALLEL=411
#
# Compute ambiguity-fixed network solution, create final NEQ/SNX/TRO files
# ------------------------------------------------------------------------
501  GPSEST    R2S_FIN   CPU=ANY; WAIT=412;  PARAM2=V_FIN
511  ADDNEQ2   R2S_FIN   CPU=ANY; WAIT=501
513  HELMCHK   R2S_FIN   CPU=ANY; WAIT=511;  NEXTJOB= 511
514  COMPAR    R2S_FIN   CPU=ANY; WAIT=513
#
# Create summary file and delete files
# ------------------------------------
901  R2S_SUM   R2S_GEN   CPU=ANY; WAIT=513
902  R2S_SAV   R2S_GEN   CPU=ANY; WAIT=901
904  R2S_DEL   R2S_GEN   CPU=ANY; WAIT=902 903;   PARAM1=(10)
#
# End of BPE
# ----------
999  DUMMY     NO_OPT    CPU=ANY; WAIT=904
```

Bernese Processing Engine

```
PID  SCRIPT    OPT_DIR   PARAMETERS
...
#
# Resolve phase ambiguities
# -------------------------
411  GNSAMBAP  R2S_AMB   CPU=ANY; WAIT=401
412  GNSAMB_P  R2S_AMB   CPU=ANY; WAIT=411;  PARALLEL=411
#
# Compute ambiguity-fixed network solution, create final NEQ/SNX/TRO files
# ------------------------------------------------------------------------
501  GPSEST    R2S_FIN   CPU=ANY; WAIT=412;  PARAM2=V_FIN
511  ADDNEQ2   R2S_FIN   CPU=ANY; WAIT=501
513  HELMCHK   R2S_FIN   CPU=ANY; WAIT=511;  NEXTJOB= 511
514  COMPAR    R2S_FIN   CPU=ANY; WAIT=513
#
# Create summary file and delete files
# ------------------------------------
901  R2S_SUM   R2S_GEN   CPU=ANY; WAIT=513
902  R2S_SAV   R2S_GEN   CPU=ANY; WAIT=901
904  R2S_DEL   R2S_GEN   CPU=ANY; WAIT=902 903;  PARAM1=(10)
#
# End of BPE
# ----------
999  DUMMY     NO_OPT    CPU=ANY; WAIT=904
```

6

Bernese Processing Engine

```
PID  SCRIPT    OPT_DIR   PARAMETERS
...
#
# Resolve phase ambiguities
# -------------------------
411  GNSAMBAP  R2S_AMB   CPU=ANY; WAIT=401
412  GNSAMB_P  R2S_AMB   CPU=ANY; WAIT=411;  PARALLEL=411
#
# Compute ambiguity-fixed network solution, create final NEQ/SNX/TRO files
# ------------------------------------------------------------------------
501  GPSEST    R2S_FIN   CPU=ANY; WAIT=412;  PARAM2=V_FIN
511  ADDNEQ2   R2S_FIN   CPU=ANY; WAIT=501
513  HELMCHK   R2S_FIN   CPU=ANY; WAIT=511;  NEXTJOB= 511
514  COMPAR    R2S_FIN   CPU=ANY; WAIT=513
#
# Create summary file and delete files
# ------------------------------------
901  R2S_SUM   R2S_GEN   CPU=ANY; WAIT=513
902  R2S_SAV   R2S_GEN   CPU=ANY; WAIT=901
904  R2S_DEL   R2S_GEN   CPU=ANY; WAIT=902 903;  PARAM1=(10)
#
# End of BPE
# ----------
999  DUMMY     NO_OPT    CPU=ANY; WAIT=904
```

6

7

Bernese Processing Engine

```
PID  SCRIPT    OPT_DIR   PARAMETERS
...
#
# Resolve phase ambiguities
# -------------------------
411  GNSAMBAP  R2S_AMB   CPU=ANY; WAIT=401
412  GNSAMB_P  R2S_AMB   CPU=ANY; WAIT=411;  PARALLEL=411
#
# Compute ambiguity-fixed network solution, create final NEQ/SNX/TRO files
# -------------------------------------------------------------------------
501  GPSEST    R2S_FIN   CPU=ANY; WAIT=412;  PARAM2=V_FIN
511  ADDNEQ2   R2S_FIN   CPU=ANY; WAIT=501
513  HELMCHK   R2S_FIN   CPU=ANY; WAIT=511;  NEXTJOB= 511
514  COMPAR    R2S_FIN   CPU=ANY; WAIT=513
#
# Create summary file and delete files
# ------------------------------------
901  R2S_SUM   R2S_GEN   CPU=ANY; WAIT=513
902  R2S_SAV   R2S_GEN   CPU=ANY; WAIT=901
904  R2S_DEL   R2S_GEN   CPU=ANY; WAIT=902 903;  PARAM1=(10)
#
# End of BPE
# ----------
999  DUMMY     NO_OPT    CPU=ANY; WAIT=904
```

6

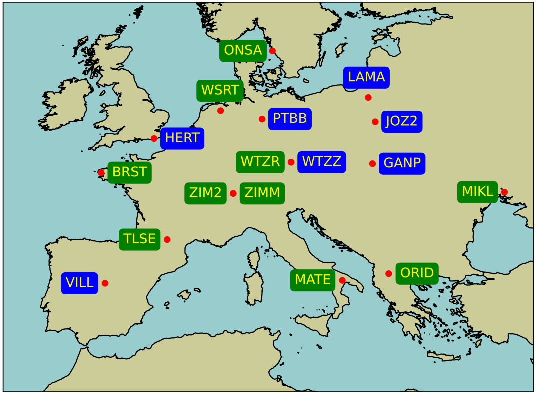
7

8

Example Data for Demonstration

Seventeen European stations of the IGS network and from the EPN:

BRST	Brest, FRA
GANP	Ganovce, SVK
HERT	Hailsham, GBR
JOZ2	Jozefoslaw, POL
LAMA	Olsztyn, POL
MATE	Matera, ITA
MIKL	Mykolaiv, UKR
ONSA	Onsala, SWE
ORID	Ohrid, MKD
PTBB	Braunschweig,DEU
TLSE	Toulouse, FRA
VILL	Villafranca, ESP
WSRT	Westerbork, NLD
WTZR, WTZZ	Kötzting, DEU
ZIM2, ZIMM	Zimmerwald, CHE



Stations used in example campaign (green stations with coordinates given in the IGS 20 reference frame)

Demonstration

Bernese GNSS Software: some facts

The software package consists of:

- a QT-based graphical user interface
- a set of processing programs (Fortran 2003)
- distribution contains the full source code
- it runs on PC/Windows, UNIX/LINUX, MAC

```
! Update the statistics over all files per system
DO iSys = 1,maxSys
  CALL statisSysAll%stat(iSys)%stack(statisSys%stat(iSys))
ENDDO

! Update the statistics over all files and satellites
CALL statisTotAll%stat(1)%stack(statisTot%stat(1))

! Print the statistics for this observation file
CALL obxprt(opt, iFil, obsHeadObx, statisSat, statisSys, stat

! Consider the condition regarding file selection
IF ( opt%what%isOptionWhatObservation() ) THEN
  IF ( opt%checkObs(statisTot%stat(1)) ) THEN
    CALL opt%lstFile%append(opt%fillst(1,iFil))
  ELSE
    CALL opt%delFile%append(opt%fillst(1,iFil))
    CALL opt%delFile%append(opt%fillst(2,iFil))
  ENDIF
ENDIF
```

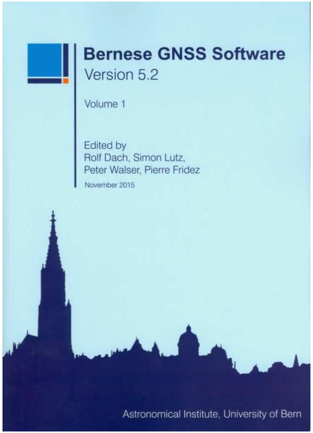
The software package counts today:

- nearly 90 processing programs and 1400 subroutines, functions, and modules about 600,000 lines of source code (including comment lines),
- the GUI/BPE-program with 18,000 lines of source code

Bernese GNSS Software: some facts

Intensive user support includes

- online-help system provides explanations on the options,
- a 850 pages user manual (downloadable as PDF for free),
- a series of README-files on various topics
- FAQ-section on the webpage,
- e-mail support to help with potential problems,
- regular updates for bugfixes and improvements,
- a one week introductory course in Bern.



Bernese GNSS Software: Processing examples

The distribution of the software package contains ready-to-use examples:

- PPP – PRECISE POINT POSITIONING
  - standard PPP for coordinate, troposphere, and receiver clock determination
  - as single- or multi-GNSS solutions (GPS, GLONASS, Galileo, BeiDou, QZSS)
  - ambiguity resolution, if the consistent bias products are available
  - several extended processing examples can be enabled: geocenter estimation, pseudo-kinematic, high-rate troposphere

Bernese GNSS Software: Processing examples

The distribution of the software package contains ready-to-use examples:

- RNX2SNX: RINEX-to-SINEX
  - standard double difference network solution
  - primary products are coordinates and troposphere corrections
  - as single- or multi-GNSS solutions (GPS, GLONASS, Galileo, BeiDou, QZSS)
  - extended ambiguity resolution scheme
  - datum definition with verification based on minimum constraint solution
- CLKDET: CLOCK DETERMINATION
  - standard zero difference network solution
  - primary products are receiver and satellite clock corrections (also, w.r.t. an existing coordinate and troposphere solution)
  - as single- or multi-GNSS solutions (GPS, GLONASS, Galileo, BeiDou, QZSS)

Bernese GNSS Software: Processing examples

The distribution of the software package contains ready-to-use examples:

- IONDET: IONOSPHERE MODEL DETERMINATION for LEOs
  - ionosphere model determination from regional or global networks for dual-frequency
- LEOPOD: PRECISE ORBIT DETERMINATION for LEOs
  - Precise Orbit Determination for a Low Earth Orbiting Satellites based on on-board GPS-measurements (e.g., for GRACE)
- SLRVAL: SLR ORBIT VALIDATION
  - Validation of an existing GNSS or LEO orbit using SLR measurements

Bernese GNSS Software: Processing examples

Each example BPEs is accompanied by an extensive README file:

- explaining the main purpose,
- providing a detailed description on the realization of the purpose,
- showing where to find the key quality indicators for the results and giving some ideas about potential sources of problems,
- listing of the BPE example configuration,
- listing the necessary input and result files.

Processing examples: coordinate computation

The processing examples distributed with the *Bernese GNSS Software* offer three ways to compute coordinates:

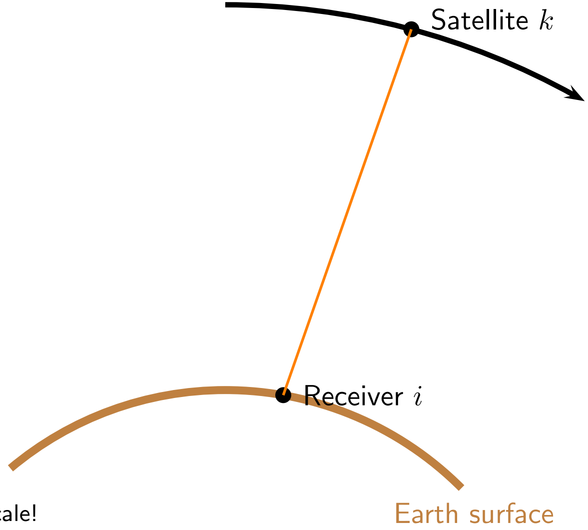
1. PPP: Precise Point Positioning
   processing of single stations, very efficient in case of parallelization

2. RNX2SNX: double-difference network solution
   efficient because clock parameters are not explicitly setup, but
   needs bookkeeping to consider correlations due to differencing

3. CLKDET: zero-difference network solution
   network solution means, to solved for satellite and receiver clock corrections
   at least a normal equation with all satellite clock parameters need to be inverted.

Are there differences between the three strategies or are they equivalent?

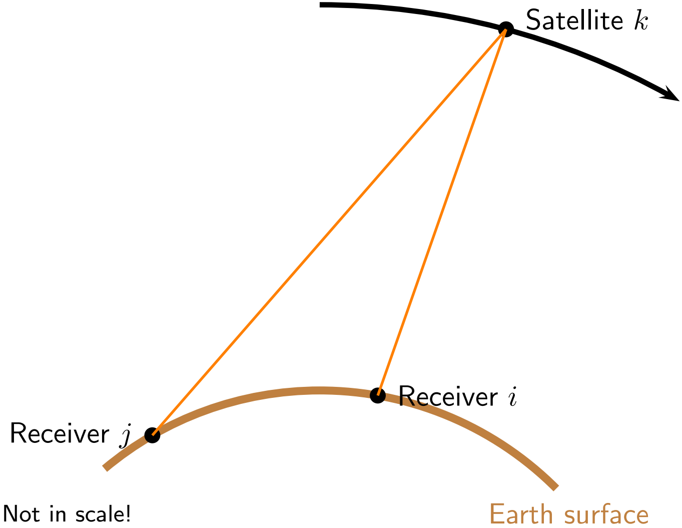
Principle of GNSS Network Solution



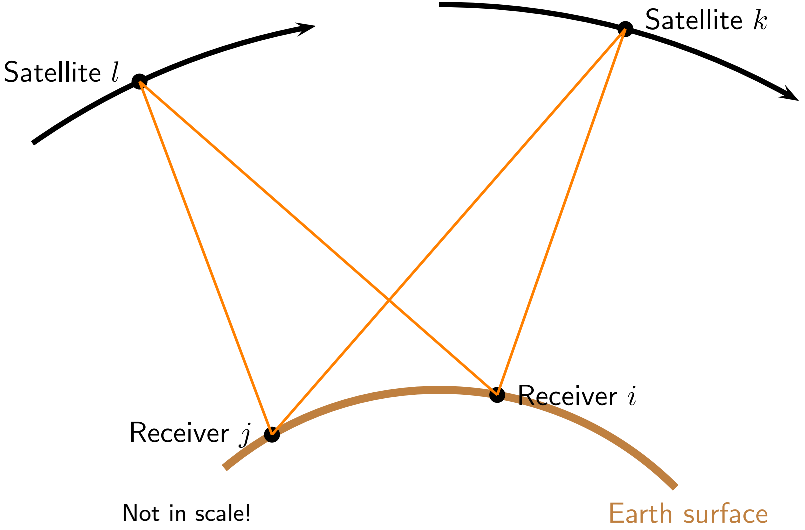
Not in scale!

R. Dach and the BSW-development team: Spotlight on Bernese GNSS Software
FIG, Technical Seminar, 10.–11. Sept. 2022, Warsaw, Poland

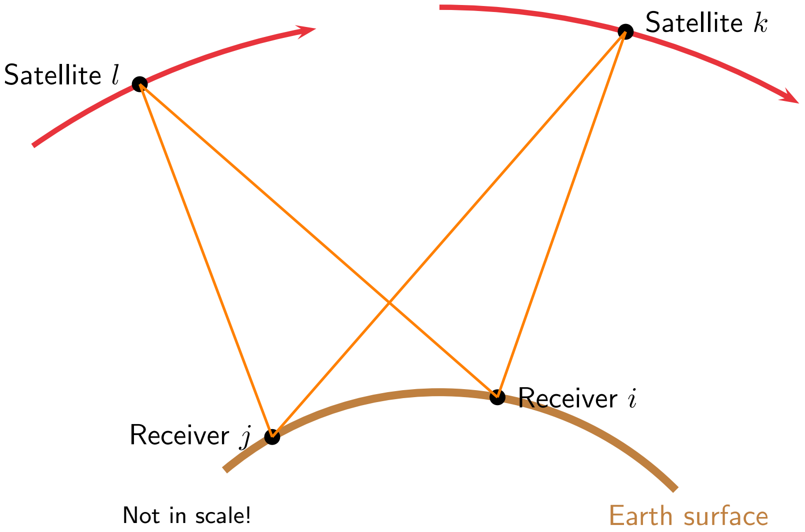
Principle of GNSS Network Solution



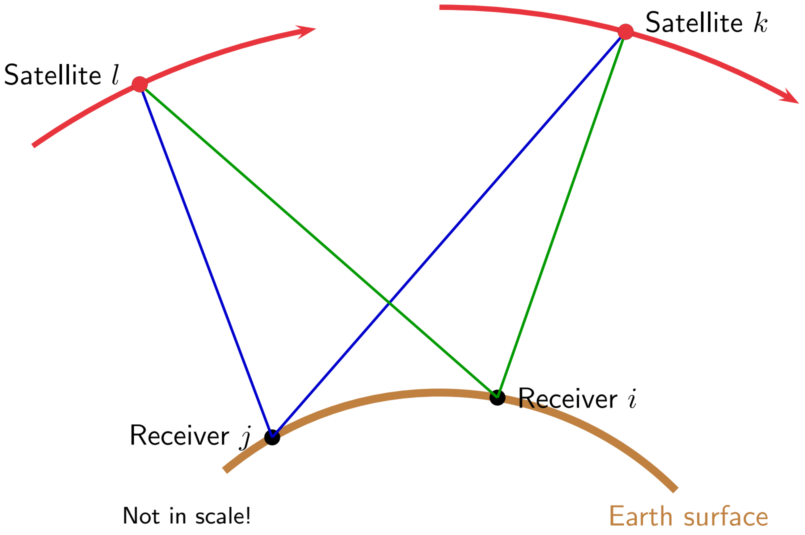
Principle of GNSS Network Solution



Principle of GNSS Network Solution

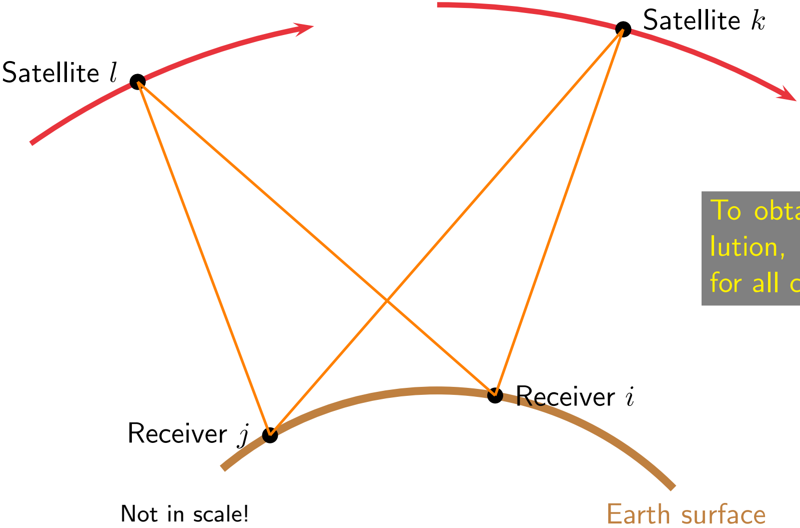


Principle of GNSS Network Solution



Not in scale!

Principle of GNSS Network Solution



To obtain a network solution, we have to solve for all clock parameters.

Principle of GNSS Network Solution

We have the GNSS observations of several stations to some satellites:

$$L_i^k = \left|\vec{x^k} - \vec{x_i}\right| + T_i^k + c\delta_i - c\delta^k + \lambda N_i^k \qquad L_i^l = \left|\vec{x^l} - \vec{x_i}\right| + T_i^l + c\delta_i - c\delta^l + \lambda N_i^l \qquad \ldots$$

$$L_j^k = \left|\vec{x^k} - \vec{x_j}\right| + T_j^k + c\delta_j - c\delta^k + \lambda N_j^k \qquad L_j^l = \left|\vec{x^l} - \vec{x_j}\right| + T_j^l + c\delta_j - c\delta^l + \lambda N_j^l \qquad \ldots$$

Principle of GNSS Network Solution

If you are not interested in the clock parameters ...

$$L_i^k = \left|\vec{x^k} - \vec{x_i}\right| + T_i^k + c\delta_i - c\delta^k + \lambda N_i^k \qquad L_i^l = \left|\vec{x^l} - \vec{x_i}\right| + T_i^l + c\delta_i - c\delta^l + \lambda N_i^l \qquad \ldots$$

$$L_j^k = \left|\vec{x^k} - \vec{x_j}\right| + T_j^k + c\delta_j - c\delta^k + \lambda N_j^k \qquad L_j^l = \left|\vec{x^l} - \vec{x_j}\right| + T_j^l + c\delta_j - c\delta^l + \lambda N_j^l \qquad \ldots$$

Principle of GNSS Network Solution

If you are not interested in the clock parameters ...

$$L_i^k = \left|\vec{x^k} - \vec{x_i}\right| + T_i^k + c\delta_i - c\delta^k + \lambda N_i^k \qquad L_i^l = \left|\vec{x^l} - \vec{x_i}\right| + T_i^l + c\delta_i - c\delta^l + \lambda N_i^l \qquad \ldots$$

$$L_j^k = \left|\vec{x^k} - \vec{x_j}\right| + T_j^k + c\delta_j - c\delta^k + \lambda N_j^k \qquad L_j^l = \left|\vec{x^l} - \vec{x_j}\right| + T_j^l + c\delta_j - c\delta^l + \lambda N_j^l \qquad \ldots$$

... we may form differences between observations to cancel out the clock parameters:

$$L_i^k - L_j^k = \left|\vec{x^k} - \vec{x_i}\right| - \left|\vec{x^k} - \vec{x_j}\right| + T_i^k - T_j^k + c\left(\delta_i - \delta^k - \delta_j + \delta^k\right) + \lambda\left(N_i^k - N_j^k\right)$$
$$L_i^l - L_j^l = \left|\vec{x^l} - \vec{x_i}\right| - \left|\vec{x^l} - \vec{x_j}\right| + T_i^l - T_j^l + c\left(\delta_i - \delta^k - \delta_j + \delta^k\right) + \lambda\left(N_i^l - N_j^l\right)$$

Principle of GNSS Network Solution

If you are not interested in the clock parameters ...

$$L_i^k = \left|\vec{x^k} - \vec{x_i}\right| + T_i^k + c\delta_i - c\delta^k + \lambda N_i^k \qquad L_i^l = \left|\vec{x^l} - \vec{x_i}\right| + T_i^l + c\delta_i - c\delta^l + \lambda N_i^l \qquad \ldots$$

$$L_j^k = \left|\vec{x^k} - \vec{x_j}\right| + T_j^k + c\delta_j - c\delta^k + \lambda N_j^k \qquad L_j^l = \left|\vec{x^l} - \vec{x_j}\right| + T_j^l + c\delta_j - c\delta^l + \lambda N_j^l \qquad \ldots$$

... we may form differences between observations to cancel out the clock parameters:

$$L_i^k - L_j^k = \left|\vec{x^k} - \vec{x_i}\right| - \left|\vec{x^k} - \vec{x_j}\right| + T_i^k - T_j^k + c\left(\delta_i - \delta^k - \delta_j + \delta^k\right) + \lambda\left(N_i^k - N_j^k\right)$$
$$L_i^l - L_j^l = \left|\vec{x^l} - \vec{x_i}\right| - \left|\vec{x^l} - \vec{x_j}\right| + T_i^l - T_j^l + c\left(\delta_i - \delta^k - \delta_j + \delta^k\right) + \lambda\left(N_i^l - N_j^l\right)$$

Principle of GNSS Network Solution

If you are not interested in the clock parameters . . .

$$L_i^k = \left|\vec{x^k} - \vec{x_i}\right| + T_i^k + c\delta_i - c\delta^k + \lambda N_i^k \qquad L_i^l = \left|\vec{x^l} - \vec{x_i}\right| + T_i^l + c\delta_i - c\delta^l + \lambda N_i^l \qquad \ldots$$

$$L_j^k = \left|\vec{x^k} - \vec{x_j}\right| + T_j^k + c\delta_j - c\delta^k + \lambda N_j^k \qquad L_j^l = \left|\vec{x^l} - \vec{x_j}\right| + T_j^l + c\delta_j - c\delta^l + \lambda N_j^l \qquad \ldots$$

. . . we may form differences between observations to cancel out the clock parameters:

$$L_i^k - L_j^k = \left|\vec{x^k} - \vec{x_i}\right| - \left|\vec{x^k} - \vec{x_j}\right| + T_i^k - T_j^k + c\left(\delta_i \cancel{- \delta^k} - \delta_j \cancel{+ \delta^k}\right) + \lambda\left(N_i^k - N_j^k\right)$$

$$L_i^l - L_j^l = \left|\vec{x^l} - \vec{x_i}\right| - \left|\vec{x^l} - \vec{x_j}\right| + T_i^l - T_j^l + c\left(\delta_i \cancel{- \delta^k} - \delta_j \cancel{+ \delta^k}\right) + \lambda\left(N_i^l - N_j^l\right)$$

Principle of GNSS Network Solution

If you are not interested in the clock parameters ...

$$L_i^k = \left|\vec{x^k} - \vec{x_i}\right| + T_i^k + c\delta_i - c\delta^k + \lambda N_i^k \qquad L_i^l = \left|\vec{x^l} - \vec{x_i}\right| + T_i^l + c\delta_i - c\delta^l + \lambda N_i^l \qquad \ldots$$

$$L_j^k = \left|\vec{x^k} - \vec{x_j}\right| + T_j^k + c\delta_j - c\delta^k + \lambda N_j^k \qquad L_j^l = \left|\vec{x^l} - \vec{x_j}\right| + T_j^l + c\delta_j - c\delta^l + \lambda N_j^l \qquad \ldots$$

... we may form differences between observations to cancel out the clock parameters:

$$L_i^k - L_j^k = \left|\vec{x^k} - \vec{x_i}\right| - \left|\vec{x^k} - \vec{x_j}\right| + T_i^k - T_j^k + c\,(\delta_i - \delta_j) + \lambda\big(N_i^k - N_j^k\big)$$
$$L_i^l - L_j^l = \left|\vec{x^l} - \vec{x_i}\right| - \left|\vec{x^l} - \vec{x_j}\right| + T_i^l - T_j^l + c\,(\delta_i - \delta_j) + \lambda\big(N_i^l - N_j^l\big)$$

Principle of GNSS Network Solution

If you are not interested in the clock parameters . . .

$$L_i^k = \left|\vec{x^k} - \vec{x_i}\right| + T_i^k + c\delta_i - c\delta^k + \lambda N_i^k \qquad L_i^l = \left|\vec{x^l} - \vec{x_i}\right| + T_i^l + c\delta_i - c\delta^l + \lambda N_i^l \qquad \ldots$$

$$L_j^k = \left|\vec{x^k} - \vec{x_j}\right| + T_j^k + c\delta_j - c\delta^k + \lambda N_j^k \qquad L_j^l = \left|\vec{x^l} - \vec{x_j}\right| + T_j^l + c\delta_j - c\delta^l + \lambda N_j^l \qquad \ldots$$

. . . we may form differences between observations to cancel out the clock parameters:

$$L_i^k - L_j^k = \left|\vec{x^k} - \vec{x_i}\right| - \left|\vec{x^k} - \vec{x_j}\right| + T_i^k - T_j^k + c\left(\delta_i - \delta_j\right) + \lambda\left(N_i^k - N_j^k\right)$$

$$L_i^l - L_j^l = \left|\vec{x^l} - \vec{x_i}\right| - \left|\vec{x^l} - \vec{x_j}\right| + T_i^l - T_j^l + c\left(\delta_i - \delta_j\right) + \lambda\left(N_i^l - N_j^l\right)$$

Principle of GNSS Network Solution

If you are not interested in the clock parameters ...

$$L_i^k = \left|\vec{x^k} - \vec{x_i}\right| + T_i^k + c\delta_i - c\delta^k + \lambda N_i^k \qquad L_i^l = \left|\vec{x^l} - \vec{x_i}\right| + T_i^l + c\delta_i - c\delta^l + \lambda N_i^l \qquad \ldots$$

$$L_j^k = \left|\vec{x^k} - \vec{x_j}\right| + T_j^k + c\delta_j - c\delta^k + \lambda N_j^k \qquad L_j^l = \left|\vec{x^l} - \vec{x_j}\right| + T_j^l + c\delta_j - c\delta^l + \lambda N_j^l \qquad \ldots$$

... we may form differences between observations to cancel out the clock parameters:

$$L_{ij}^k = \left|\vec{x^k} - \vec{x_i}\right| - \left|\vec{x^k} - \vec{x_j}\right| + T_{ij}^k + c\,(\delta_i - \delta_j) + \lambda N_{ij}^k$$
$$L_{ij}^l = \left|\vec{x^l} - \vec{x_i}\right| - \left|\vec{x^l} - \vec{x_j}\right| + T_{ij}^l + c\,(\delta_i - \delta_j) + \lambda N_{ij}^l$$

Principle of GNSS Network Solution

If you are not interested in the clock parameters ...

$$L_i^k = \left|\vec{x^k} - \vec{x_i}\right| + T_i^k + c\delta_i - c\delta^k + \lambda N_i^k \qquad L_i^l = \left|\vec{x^l} - \vec{x_i}\right| + T_i^l + c\delta_i - c\delta^l + \lambda N_i^l \qquad \ldots$$

$$L_j^k = \left|\vec{x^k} - \vec{x_j}\right| + T_j^k + c\delta_j - c\delta^k + \lambda N_j^k \qquad L_j^l = \left|\vec{x^l} - \vec{x_j}\right| + T_j^l + c\delta_j - c\delta^l + \lambda N_j^l \qquad \ldots$$

... we may form differences between observations to cancel out the clock parameters:

$$L_{ij}^k = \left|\vec{x^k} - \vec{x_i}\right| - \left|\vec{x^k} - \vec{x_j}\right| + T_{ij}^k + c\left(\delta_i - \delta_j\right) + \lambda N_{ij}^k$$
$$L_{ij}^l = \left|\vec{x^l} - \vec{x_i}\right| - \left|\vec{x^l} - \vec{x_j}\right| + T_{ij}^l + c\left(\delta_i - \delta_j\right) + \lambda N_{ij}^l$$

Principle of GNSS Network Solution

If you are not interested in the clock parameters ...

$$L_i^k = \left|\vec{x^k} - \vec{x_i}\right| + T_i^k + c\delta_i - c\delta^k + \lambda N_i^k \qquad L_i^l = \left|\vec{x^l} - \vec{x_i}\right| + T_i^l + c\delta_i - c\delta^l + \lambda N_i^l \qquad \ldots$$

$$L_j^k = \left|\vec{x^k} - \vec{x_j}\right| + T_j^k + c\delta_j - c\delta^k + \lambda N_j^k \qquad L_j^l = \left|\vec{x^l} - \vec{x_j}\right| + T_j^l + c\delta_j - c\delta^l + \lambda N_j^l \qquad \ldots$$

... we may form differences between observations to cancel out the clock parameters:

$$L_{ij}^k = \left|\vec{x^k} - \vec{x_i}\right| - \left|\vec{x^k} - \vec{x_j}\right| + T_{ij}^k + c\,(\delta_i - \delta_j) + \lambda N_{ij}^k$$

$$L_{ij}^l = \left|\vec{x^l} - \vec{x_i}\right| - \left|\vec{x^l} - \vec{x_j}\right| + T_{ij}^l + c\,(\delta_i - \delta_j) + \lambda N_{ij}^l$$

$$L_{ij}^{kl} = \left|\vec{x^k} - \vec{x_i}\right| - \left|\vec{x^k} - \vec{x_j}\right| - \left|\vec{x^l} - \vec{x_i}\right| + \left|\vec{x^l} - \vec{x_j}\right| + T_{ij}^{kl} + \lambda N_{ij}^{kl}$$

Conclusions:

- A consequent creation of (artificial) double-difference observations is equivalent to pre-eliminating the clock parameters on normal equation level.

Principle of GNSS Network Solution

Conclusions:

- A consequent creation of (artificial) double-difference observations is equivalent to pre-eliminating the clock parameters on normal equation level.

- When using the same original observations, we obtain the same estimates for the geometry-related parameters on zero, single or double difference level (given that all existing correlations are considered).

Principle of GNSS Network Solution

Conclusions:

- A consequent creation of (artificial) double-difference observations is equivalent to pre-eliminating the clock parameters on normal equation level.
- When using the same original observations, we obtain the same estimates for the geometry-related parameters on zero, single or double difference level (given that all existing correlations are considered).
- The ambiguity resolution is directly possible only on double-difference level (otherwise some bias parameters are needed).

Principle of GNSS Network Solution

Conclusions:

- A consequent creation of (artificial) double-difference observations is equivalent to pre-eliminating the clock parameters on normal equation level.

- When using the same original observations, we obtain the same estimates for the geometry-related parameters on zero, single or double difference level (given that all existing correlations are considered).

- The ambiguity resolution is directly possible only on double-difference level (otherwise some bias parameters are needed).

- Effects that cancel out when differencing the observations are absorbed by the satellite clock parameters in the zero-difference approach.

Processing examples: coordinate computation

The processing examples distributed with the *Bernese GNSS Software* offer three ways to compute coordinates:

1. PPP: Precise Point Positioning
   processing of single stations, very efficient in case of parallelization

2. RNX2SNX: double-difference network solution
   efficient because clock parameters are not explicitly setup, but
   needs bookkeeping to consider correlations due to differencing

3. CLKDET: zero-difference network solution
   network solution means, to solved for satellite and receiver clock corrections
   at least a normal equation with all satellite clock parameters need to be inverted.

Are there differences between the three strategies or are they equivalent?

Processing examples: coordinate computation

The processing examples distributed with the *Bernese GNSS Software* offer three ways to compute coordinates:

1. PPP: Precise Point Positioning
   processing of single stations, very efficient in case of parallelization

2. RNX2SNX: double-difference network solution
   efficient because clock parameters are not explicitly setup, but
   needs bookkeeping to consider correlations due to differencing

3. CLKDET: zero-difference network solution
   network solution means, to solved for satellite and receiver clock corrections
   at least a normal equation with all satellite clock parameters need to be inverted.

Zero- and double-difference solutions are equivalent.

Processing examples: coordinate computation

The processing examples distributed with the *Bernese GNSS Software* offer three ways to compute coordinates:

1. PPP: Precise Point Positioning
   processing of single stations, very efficient in case of parallelization

2. RNX2SNX: double-difference network solution
   efficient because clock parameters are not explicitly setup, but
   needs bookkeeping to consider correlations due to differencing

3. CLKDET: zero-difference network solution
   network solution means, to solved for satellite and receiver clock corrections
   at least a normal equation with all satellite clock parameters need to be inverted.

What is the consequence of introducing the GNSS orbits?

Datum Definition in a Network Solution

The *Bernese GNSS Software* supports:

- Free network solution
  no constraints on station coordinates
  Datum information is only introduced by fixed satellite orbits.

- Minimum constraint solution
  no-net translation, no-net rotation, no-net scale
  w.r.t. reference network

- Coordinates constrained:
  Constraining of station coordinate parameters

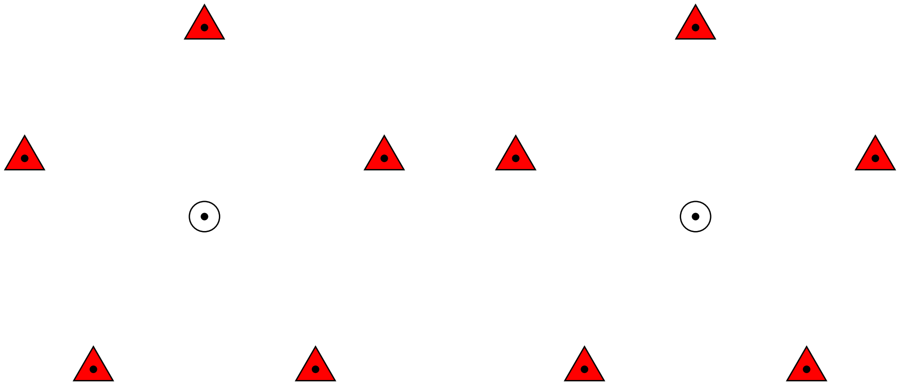
- Coordinates fixed:
  Deleting coordinate parameters from the NEQ
  Not recommended if NEQ-files are stored.

Demonstration

Principle of datum definition

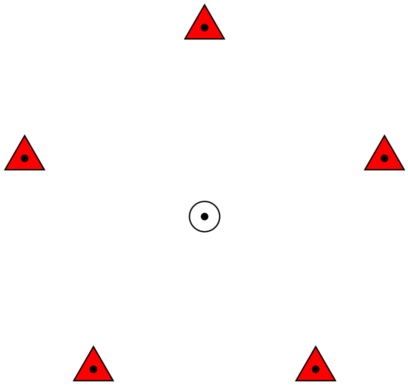
Solution A

Solution B



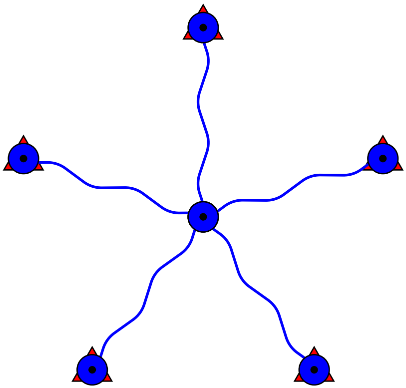
Principle of datum definition

Solution A



Solution B

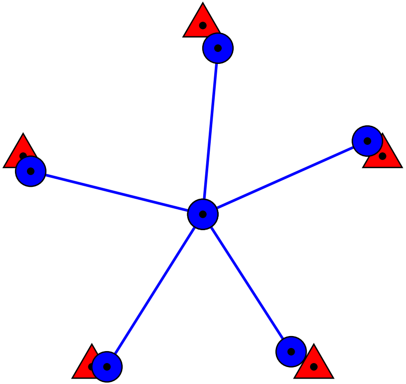
Fixed reference coordinates



Principle of datum definition

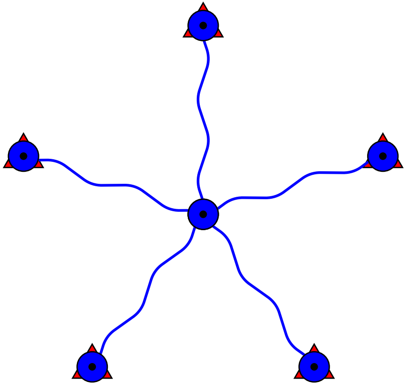
Solution A

Minimum constraint solution



Solution B

Fixed reference coordinates



Datum definition in the Bernese GNSS Software

Minimum constraint solution:

- Constraint on translation/rotation/scale of the network w.r.t. reference sites
- No distortion of the network geometry
- All coordinates are improved
- Well suited to identify problems with reference sites
  List of reference sites may automatically be verified by HELMR1-program.

Datum definition in the Bernese GNSS Software

Minimum constraint solution:

- Constraint on translation/rotation/scale of the network w.r.t. reference sites
- No distortion of the network geometry
- All coordinates are improved
- Well suited to identify problems with reference sites
  List of reference sites may automatically be verified by HELMR1-program.

Usually (regional solutions):

- Orientation of the network is given if the orbits were introduced as fixed $\Rightarrow$ no-net-rotation conditions are not needed/reasonable
- Only "Center of network" condition (translations), i.e., the center of selected reference sites remains unchanged

Datum definition in the Bernese GNSS Software

Minimum constraint solution:

- Constraint on translation/rotation/scale of the network w.r.t. reference sites
- No distortion of the network geometry
- All coordinates are improved
- Well suited to identify problems with reference sites
  List of reference sites may automatically be verified by HELMR1-program.

Coordinates introduced:

- "Fixing" coordinates of reference stations is useful if they are expected to be more accurate than the current GNSS solution.
- In that scenario it is even more essential to check the consistency first!

Processing examples: coordinate computation

The processing examples distributed with the *Bernese GNSS Software* offer three ways to compute coordinates:

1. PPP: Precise Point Positioning
   processing of single stations, very efficient in case of parallelization

2. RNX2SNX: double-difference network solution
   efficient because clock parameters are not explicitly setup, but
   needs bookkeeping to consider correlations due to differencing

3. CLKDET: zero-difference network solution
   network solution means, to solved for satellite and receiver clock corrections
   at least a normal equation with all satellite clock parameters need to be inverted.

What is the consequence of introducing the GNSS orbits?

Processing examples: coordinate computation

The processing examples distributed with the *Bernese GNSS Software* offer three ways to compute coordinates:

1. PPP: Precise Point Positioning
   processing of single stations, very efficient in case of parallelization

2. RNX2SNX: double-difference network solution
   efficient because clock parameters are not explicitly setup, but
   needs bookkeeping to consider correlations due to differencing

3. CLKDET: zero-difference network solution
   network solution means, to solved for satellite and receiver clock corrections
   at least a normal equation with all satellite clock parameters need to be inverted.

A consistent datum definition using verified sites only is indispersible.

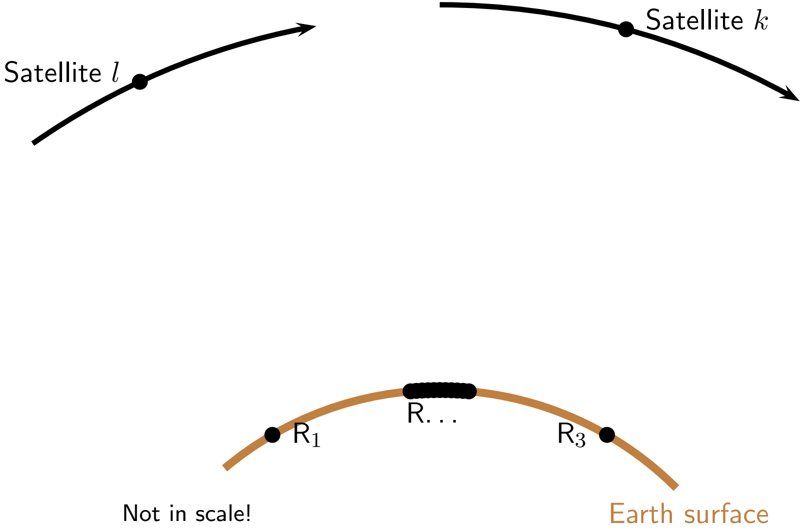
Processing examples: coordinate computation

The processing examples distributed with the *Bernese GNSS Software* offer three ways to compute coordinates:

1. PPP: Precise Point Positioning
   processing of single stations, very efficient in case of parallelization
2. RNX2SNX: double-difference network solution
   efficient because clock parameters are not explicitly setup, but
   needs bookkeeping to consider correlations due to differencing
3. CLKDET: zero-difference network solution
   network solution means, to solved for satellite and receiver clock corrections
   at least a normal equation with all satellite clock parameters need to be inverted.

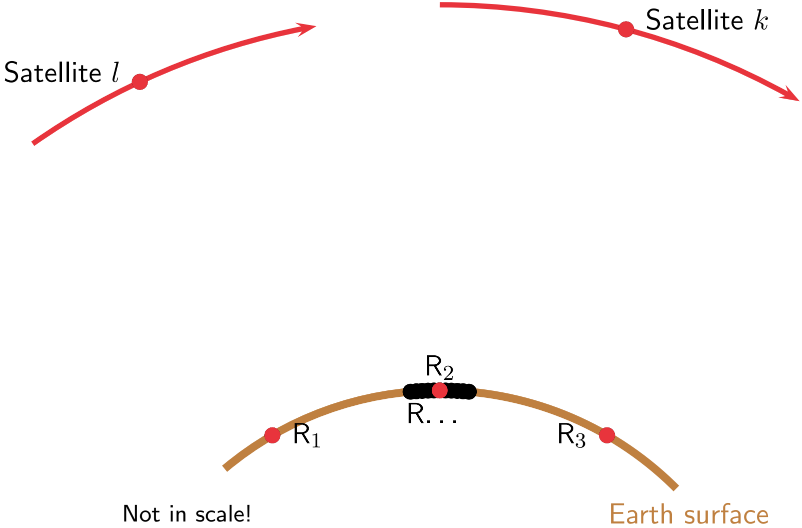
What about the datum definition in case of PPP?

PPP – Precise Point Positioning

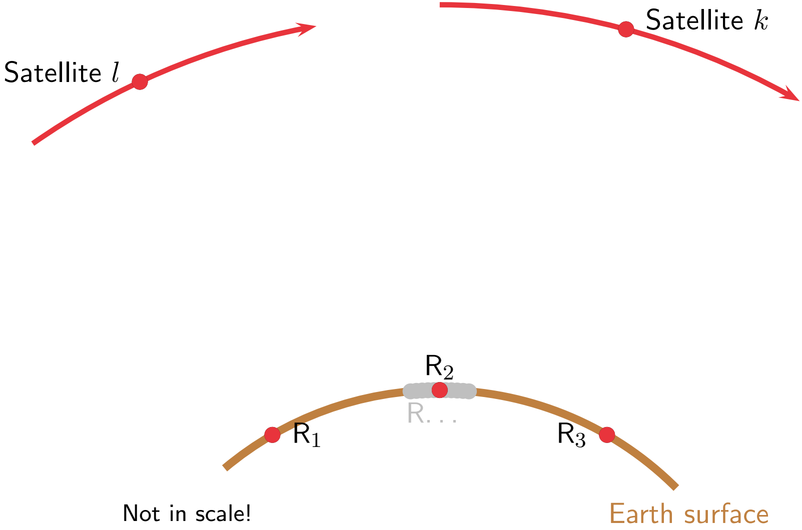


Not in scale!

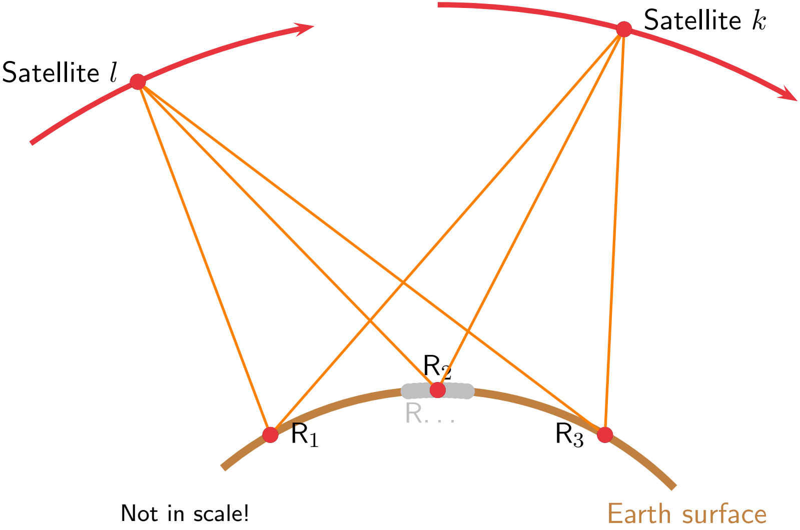
PPP – Precise Point Positioning



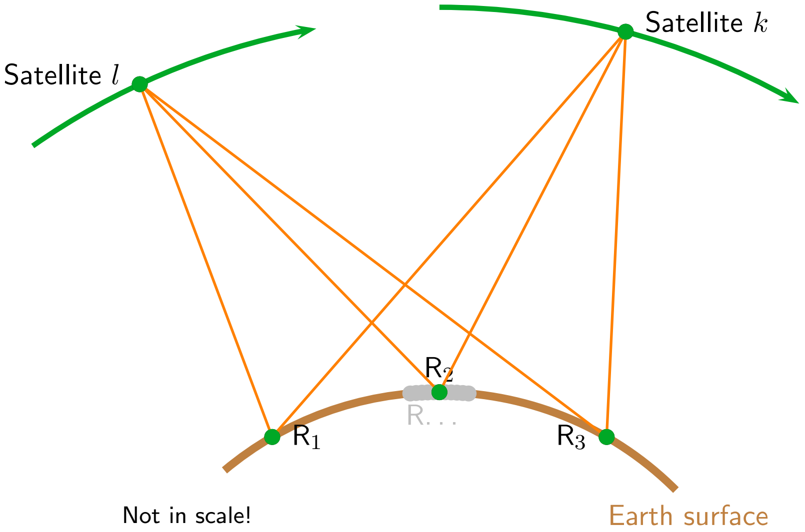
PPP – Precise Point Positioning



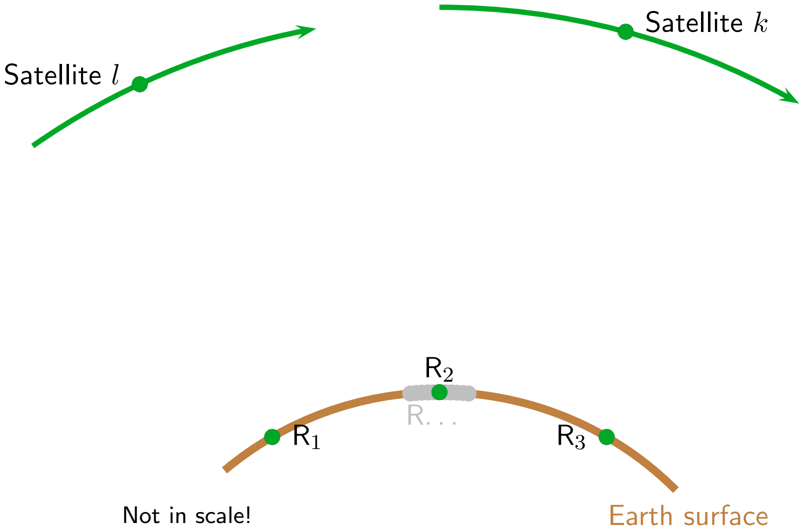
PPP – Precise Point Positioning



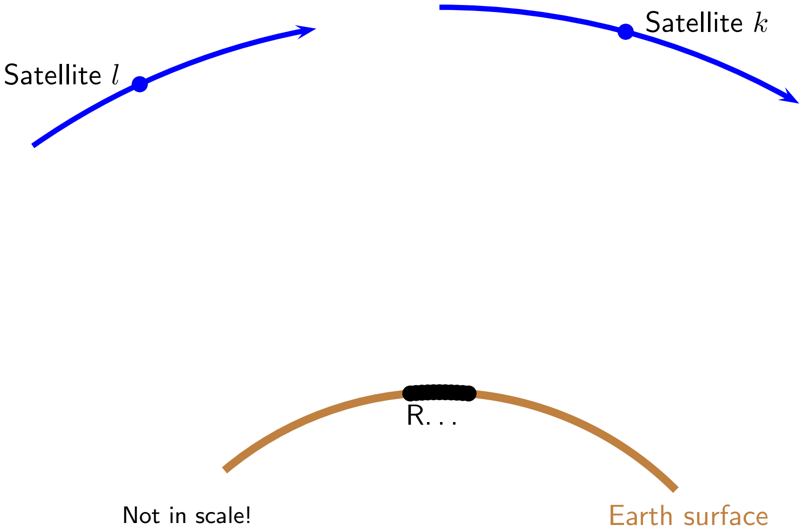
PPP – Precise Point Positioning



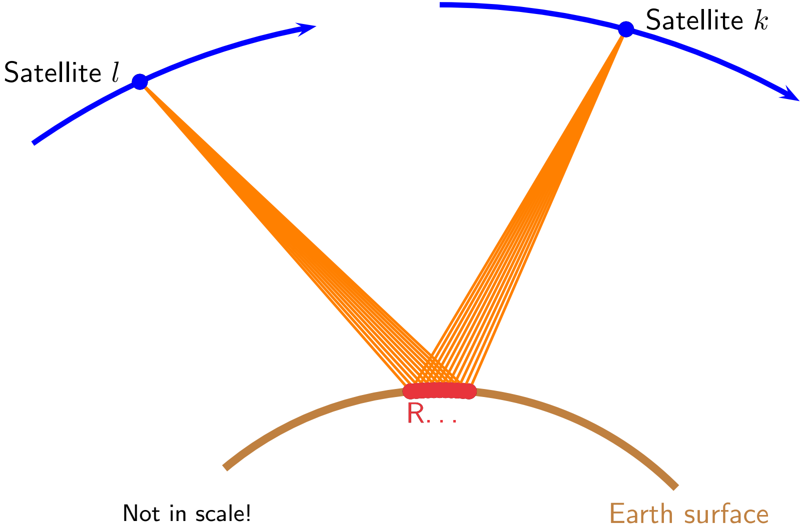
PPP – Precise Point Positioning



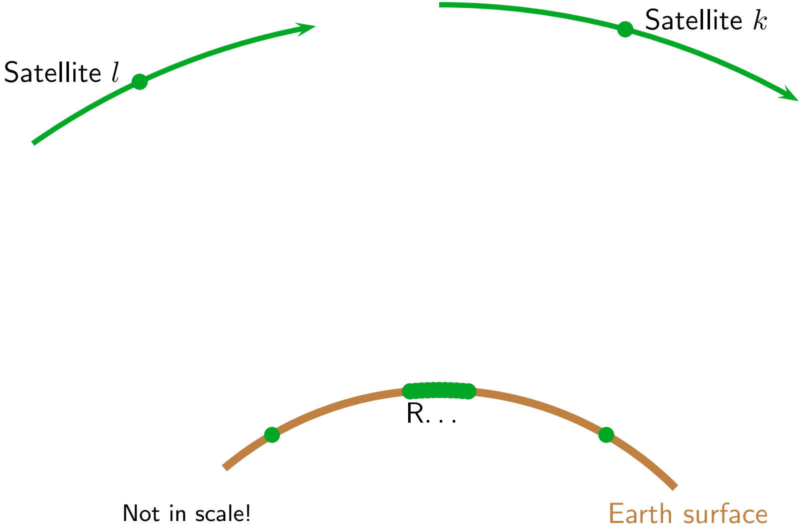
PPP – Precise Point Positioning



PPP – Precise Point Positioning



PPP – Precise Point Positioning



PPP – Precise Point Positioning

$$L_1^k = \left| \vec{x^k} - \vec{x_1} \right| + T_1^k + c\delta_1 - c\delta^k + \lambda N_1^k \qquad L_1^l = \left| \vec{x^l} - \vec{x_1} \right| + T_1^l + c\delta_1 - c\delta^l + \lambda N_1^l \qquad \ldots$$

PPP – Precise Point Positioning

$$L_1^k = \left|\vec{x^k} - \vec{x_1}\right| + T_1^k + c\delta_1 - c\delta^k + \lambda N_1^k \qquad L_1^l = \left|\vec{x^l} - \vec{x_1}\right| + T_1^l + c\delta_1 - c\delta^l + \lambda N_1^l \qquad \ldots$$

$$L_2^k = \left|\vec{x^k} - \vec{x_2}\right| + T_2^k + c\delta_2 - c\delta^k + \lambda N_2^k \qquad L_2^l = \left|\vec{x^l} - \vec{x_2}\right| + T_2^l + c\delta_2 - c\delta^l + \lambda N_2^l \qquad \ldots$$

PPP – Precise Point Positioning

$$L_1^k = \left|\vec{x^k} - \vec{x_1}\right| + T_1^k + c\delta_1 - c\delta^k + \lambda N_1^k \qquad L_1^l = \left|\vec{x^l} - \vec{x_1}\right| + T_1^l + c\delta_1 - c\delta^l + \lambda N_1^l \qquad \ldots$$

$$L_2^k = \left|\vec{x^k} - \vec{x_2}\right| + T_2^k + c\delta_2 - c\delta^k + \lambda N_2^k \qquad L_2^l = \left|\vec{x^l} - \vec{x_2}\right| + T_2^l + c\delta_2 - c\delta^l + \lambda N_2^l \qquad \ldots$$

$$L_3^k = \left|\vec{x^k} - \vec{x_3}\right| + T_3^k + c\delta_3 - c\delta^k + \lambda N_3^k \qquad L_3^l = \left|\vec{x^l} - \vec{x_3}\right| + T_3^l + c\delta_3 - c\delta^l + \lambda N_3^l \qquad \ldots$$

PPP – Precise Point Positioning

GNSS observation equations for a large number of stations

$$L_1^k = \left|\vec{x^k} - \vec{x_1}\right| + T_1^k + c\delta_1 - c\delta^k + \lambda N_1^k \qquad L_1^l = \left|\vec{x^l} - \vec{x_1}\right| + T_1^l + c\delta_1 - c\delta^l + \lambda N_1^l \qquad \ldots$$

$$L_2^k = \left|\vec{x^k} - \vec{x_2}\right| + T_2^k + c\delta_2 - c\delta^k + \lambda N_2^k \qquad L_2^l = \left|\vec{x^l} - \vec{x_2}\right| + T_2^l + c\delta_2 - c\delta^l + \lambda N_2^l \qquad \ldots$$

$$L_3^k = \left|\vec{x^k} - \vec{x_3}\right| + T_3^k + c\delta_3 - c\delta^k + \lambda N_3^k \qquad L_3^l = \left|\vec{x^l} - \vec{x_3}\right| + T_3^l + c\delta_3 - c\delta^l + \lambda N_3^l \qquad \ldots$$

$$\vdots$$

$$L_n^k = \left|\vec{x^k} - \vec{x_n}\right| + T_n^k + c\delta_n - c\delta^k + \lambda N_n^k \qquad L_n^l = \left|\vec{x^l} - \vec{x_n}\right| + T_n^l + c\delta_n - c\delta^l + \lambda N_n^l \qquad \ldots$$

PPP – Precise Point Positioning

GNSS observation equations for a large number of stations

$$L_1^k = \left|\vec{x^k} - \vec{x_1}\right| + T_1^k + c\delta_1 - c\delta^k + \lambda N_1^k \qquad L_1^l = \left|\vec{x^l} - \vec{x_1}\right| + T_1^l + c\delta_1 - c\delta^l + \lambda N_1^l \qquad \ldots$$

$$L_2^k = \left|\vec{x^k} - \vec{x_2}\right| + T_2^k + c\delta_2 - c\delta^k + \lambda N_2^k \qquad L_2^l = \left|\vec{x^l} - \vec{x_2}\right| + T_2^l + c\delta_2 - c\delta^l + \lambda N_2^l \qquad \ldots$$

$$L_3^k = \left|\vec{x^k} - \vec{x_3}\right| + T_3^k + c\delta_3 - c\delta^k + \lambda N_3^k \qquad L_3^l = \left|\vec{x^l} - \vec{x_3}\right| + T_3^l + c\delta_3 - c\delta^l + \lambda N_3^l \qquad \ldots$$

$$\vdots$$

$$L_n^k = \left|\vec{x^k} - \vec{x_n}\right| + T_n^k + c\delta_n - c\delta^k + \lambda N_n^k \qquad L_n^l = \left|\vec{x^l} - \vec{x_n}\right| + T_n^l + c\delta_n - c\delta^l + \lambda N_n^l \qquad \ldots$$

PPP – Precise Point Positioning

GNSS observation equations for a large number of stations

$$L_1^k = \left|\vec{x^k} - \vec{x_1}\right| + T_1^k + c\delta_1 - c\delta^k + \lambda N_1^k \qquad L_1^l = \left|\vec{x^l} - \vec{x_1}\right| + T_1^l + c\delta_1 - c\delta^l + \lambda N_1^l \qquad \ldots$$

$$L_2^k = \left|\vec{x^k} - \vec{x_2}\right| + T_2^k + c\delta_2 - c\delta^k + \lambda N_2^k \qquad L_2^l = \left|\vec{x^l} - \vec{x_2}\right| + T_2^l + c\delta_2 - c\delta^l + \lambda N_2^l \qquad \ldots$$

$$L_3^k = \left|\vec{x^k} - \vec{x_3}\right| + T_3^k + c\delta_3 - c\delta^k + \lambda N_3^k \qquad L_3^l = \left|\vec{x^l} - \vec{x_3}\right| + T_3^l + c\delta_3 - c\delta^l + \lambda N_3^l \qquad \ldots$$

$$\vdots$$

$$L_n^k = \left|\vec{x^k} - \vec{x_n}\right| + T_n^k + c\delta_n - c\delta^k + \lambda N_n^k \qquad L_n^l = \left|\vec{x^l} - \vec{x_n}\right| + T_n^l + c\delta_n - c\delta^l + \lambda N_n^l \qquad \ldots$$

PPP – Precise Point Positioning

GNSS observation equations for a large number of stations

$$L_1^k = \left|\vec{x^k} - \vec{x_1}\right| + T_1^k + c\delta_1 - c\delta^k + \lambda N_1^k \qquad L_1^l = \left|\vec{x^l} - \vec{x_1}\right| + T_1^l + c\delta_1 - c\delta^l + \lambda N_1^l \qquad \ldots$$

$$L_2^k = \left|\vec{x^k} - \vec{x_2}\right| + T_2^k + c\delta_2 - c\delta^k + \lambda N_2^k \qquad L_2^l = \left|\vec{x^l} - \vec{x_2}\right| + T_2^l + c\delta_2 - c\delta^l + \lambda N_2^l \qquad \ldots$$

$$L_3^k = \left|\vec{x^k} - \vec{x_3}\right| + T_3^k + c\delta_3 - c\delta^k + \lambda N_3^k \qquad L_3^l = \left|\vec{x^l} - \vec{x_3}\right| + T_3^l + c\delta_3 - c\delta^l + \lambda N_3^l \qquad \ldots$$

$$\vdots$$

$$L_n^k = \left|\vec{x^k} - \vec{x_n}\right| + T_n^k + c\delta_n - c\delta^k + \lambda N_n^k \qquad L_n^l = \left|\vec{x^l} - \vec{x_n}\right| + T_n^l + c\delta_n - c\delta^l + \lambda N_n^l \qquad \ldots$$

PPP – Precise Point Positioning

GNSS observation equations for a large number of stations

$$L_1^k = \left|\vec{x^k} - \vec{x_1}\right| + T_1^k + c\delta_1 - c\delta^k + \lambda N_1^k \qquad L_1^l = \left|\vec{x^l} - \vec{x_1}\right| + T_1^l + c\delta_1 - c\delta^l + \lambda N_1^l \qquad \ldots$$

$$L_2^k = \left|\vec{x^k} - \vec{x_2}\right| + T_2^k + c\delta_2 - c\delta^k + \lambda N_2^k \qquad L_2^l = \left|\vec{x^l} - \vec{x_2}\right| + T_2^l + c\delta_2 - c\delta^l + \lambda N_2^l \qquad \ldots$$

$$L_3^k = \left|\vec{x^k} - \vec{x_3}\right| + T_3^k + c\delta_3 - c\delta^k + \lambda N_3^k \qquad L_3^l = \left|\vec{x^l} - \vec{x_3}\right| + T_3^l + c\delta_3 - c\delta^l + \lambda N_3^l \qquad \ldots$$

$$\vdots$$

$$L_n^k = \left|\vec{x^k} - \vec{x_n}\right| + T_n^k + c\delta_n - c\delta^k + \lambda N_n^k \qquad L_n^l = \left|\vec{x^l} - \vec{x_n}\right| + T_n^l + c\delta_n - c\delta^l + \lambda N_n^l \qquad \ldots$$

PPP – Precise Point Positioning

Conclusions:

- Even if PPP stands for "Precise *Point* Positioning", the observation equations of an individual point are in any case a part of a global network solution with a zero-weight for station-independent parameters.

PPP – Precise Point Positioning

Conclusions:

- Even if PPP stands for "Precise *Point* Positioning", the observation equations of an individual point are in any case a part of a global network solution with a zero-weight for station-independent parameters.

- That's why the observation equations for a PPP processing have to be fully consistent to the related network solution.
  Any inconsistency will degrade your PPP results.

Astronomical Institute, University of Bern AIUB

PPP – Precise Point Positioning

Conclusions:

- Even if PPP stands for "Precise *Point* Positioning", the observation equations of an individual point are in any case a part of a global network solution with a zero-weight for station-independent parameters.

- That's why the observation equations for a PPP processing have to be fully consistent to the related network solution.
  Any inconsistency will degrade your PPP results.

- PPP-based station coordinates join the datum realization of the original network solution.

Demonstration

Bernese GNSS Software: Selected parameters (I)

- Station Coordinates
  Rectangular coordinates X, Y, Z in the ITRF (at present the ITRF 2014 is used).
  The results are also expressed in the (user defined) geodetic datum (λ, β, h)

- Station Velocities
  In program ADDNEQ2 station velocities may be set up if a long time series of NEQ systems containing the same stations is available.

Bernese GNSS Software: Selected parameters (I)

- Station Coordinates
  Rectangular coordinates X, Y, Z in the ITRF (at present the ITRF 2014 is used).
  The results are also expressed in the (user defined) geodetic datum (λ, β, h)

- Station Velocities
  In program ADDNEQ2 station velocities may be set up if a long time series of NEQ systems containing the same stations is available.

- Epoch specific station coordinates
  A set of station coordinates is assigned to each epoch (kinematic surveys).

List of Parameters (II)

- Scaling factors
  Scaling factors for up to three crustal deformation models provided in global grid files can be estimated to validate the model and/or to investigate the impact of the model on GNSS-derived parameters.

List of Parameters (II)

- Scaling factors
  Scaling factors for up to three crustal deformation models provided in global grid files can be estimated to validate the model and/or to investigate the impact of the model on GNSS-derived parameters.

- GNSS–specific Parameters
  Biases for station coordinates, vertical and horizontal troposphere between the observations from different GNSS may be indicate deficiencies in the GNSS-specific receiver antenna calibration

List of Parameters (II)

- Scaling factors
  Scaling factors for up to three crustal deformation models provided in global grid files can be estimated to validate the model and/or to investigate the impact of the model on GNSS-derived parameters.

- GNSS–specific Parameters
  Biases for station coordinates, vertical and horizontal troposphere between the observations from differrent GNSS may be indicate deficiencies in the GNSS-specific receiver antenna calibration

- HELMERT-parameters
  Transformation parameters (translation, rotation, scale) between the coordinate parameters from different normal equations can be estimated.

Summary

Why the *Bernese GNSS Software* might also be intersting for your applications?

Summary

Why the *Bernese GNSS Software* might also be intersting for your applications?

- You have three ways to obtain coordinates for your network using the ready-to-use examples:
  - PPP-Precise point positioning
  - Zero- and double-different network solution

Summary

Why the *Bernese GNSS Software* might also be intersting for your applications?

- You have three ways to obtain coordinates for your network using the ready-to-use examples:
  - PPP-Precise point positioning
  - Zero- and double-different network solution

- Highly automated processing is predestined
  - when you are responsible for a permanent GNSS-network
  - when you have to repeat similar GNSS-tasks for various projects.

Summary

Why the *Bernese GNSS Software* might also be intersting for your applications?

- You have three ways to obtain coordinates for your network using the ready-to-use examples:
  - PPP-Precise point positioning
  - Zero- and double-different network solution

- Highly automated processing is predestined
  - when you are responsible for a permanent GNSS-network
  - when you have to repeat similar GNSS-tasks for various projects.

- You have all aspects of the GNSS analysis under your control.

Summary

Why the *Bernese GNSS Software* might also be intersting for your applications?

- You have three ways to obtain coordinates for your network using the ready-to-use examples:
  - PPP-Precise point positioning
  - Zero- and double-different network solution
- Highly automated processing is predestined
  - when you are responsible for a permanent GNSS-network
  - when you have to repeat similar GNSS-tasks for various projects.
- You have all aspects of the GNSS analysis under your control.
- Many other groups have discovered the software already for their tasks.

THANK YOU

for your attention

Publications of the satellite geodesy research group:

`http://www.bernese.unibe.ch/publist`