Tutorial 4

Carrier ambiguity fixing

Contact: jaume.sanz@upc.edu
Web site: http://www.gage.upc.edu

Slides associated to
***gLAB* version 2.0.0**

Authorship statement

24 April 2014

Aim of this tutorial

- This tutorial is devoted to analyse and assess the ambiguity fixing and the differential positioning with carrier phase measurements (L1, L2). Two receivers UPC1 and UPC2 with a baseline of about 40 metres are considered.

- This study includes ambiguity fixing using the “cascade method” (i.e., fixing one at a time) and with the LAMBDA method.

  The effect of synchronization errors between the reference station and the user is and its effect on the navigation and ambiguity fixing is also analysed

- **All software tools** (including ***gLAB***) and associated files for the laboratory session are included in the CD-ROM or USB stick associated with this tutorial.

OVERVIEW

- **Introduction:** gLAB processing in command line.
- **Preliminary computations:** Data files.
- **Session A:** Fixing DD ambiguities <u>one at a time</u>: UPC1-UPC2.
- **Session B:** Assessing the fixed ambiguities in navigation: Differential positioning of UPC1-UPC2 receivers.
- **Session C:** Fixing DD ambiguities with <u>LAMBDA method</u>.

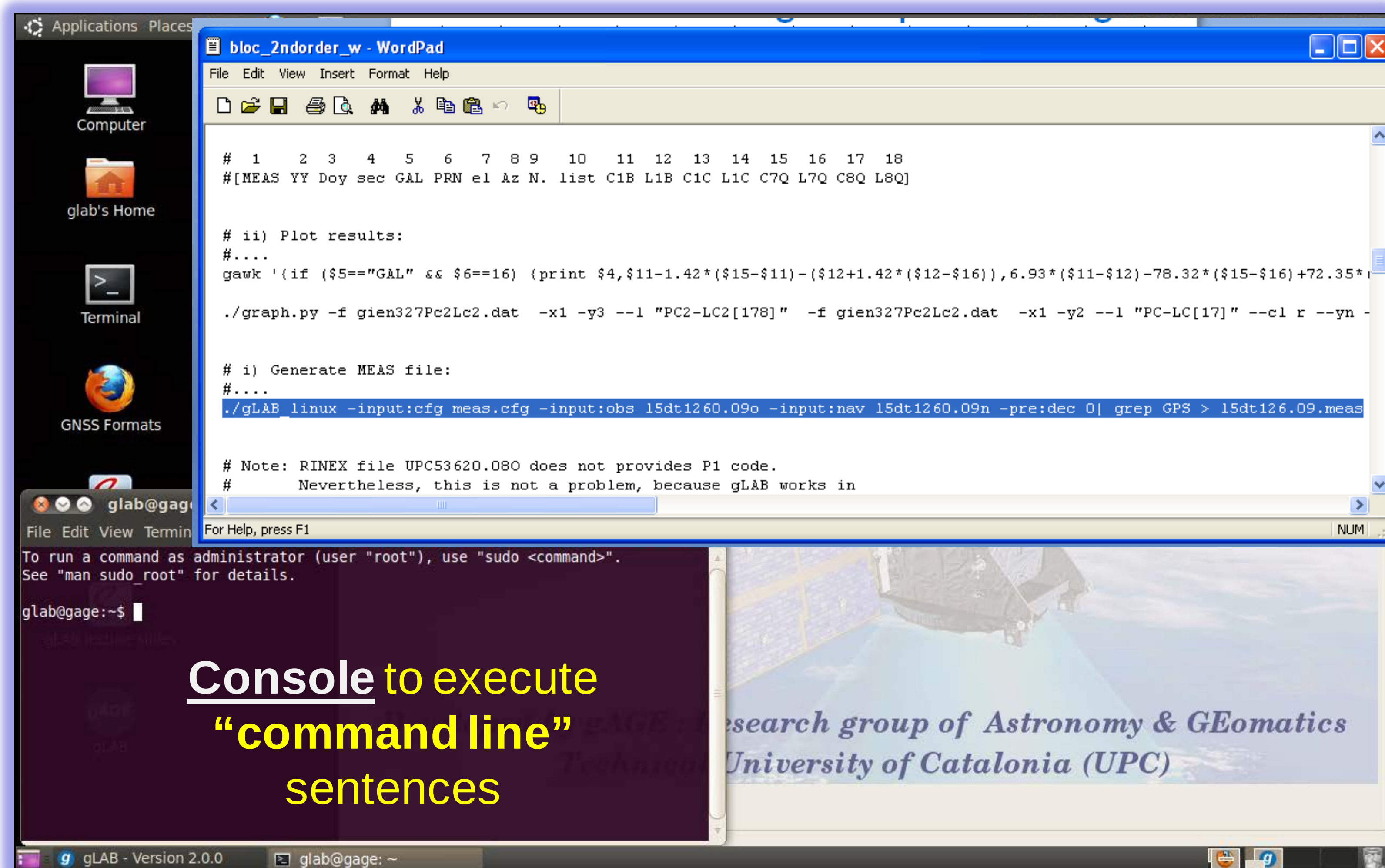
Note: UPC1-UPC2 receivers baseline: 37.95 metres.

OVERVIEW

- **Introduction: gLAB processing in command line.**
- **Preliminary computations:** Data files.
- **Session A:** Fixing DD ambiguities <u>one at a time</u>: UPC1-UPC2.
- **Session B:** Assessing the fixed ambiguities in navigation: Differential positioning of UPC1-UPC2 receivers.
- **Session C:** Fixing DD ambiguities with <u>LAMBDA method</u>.

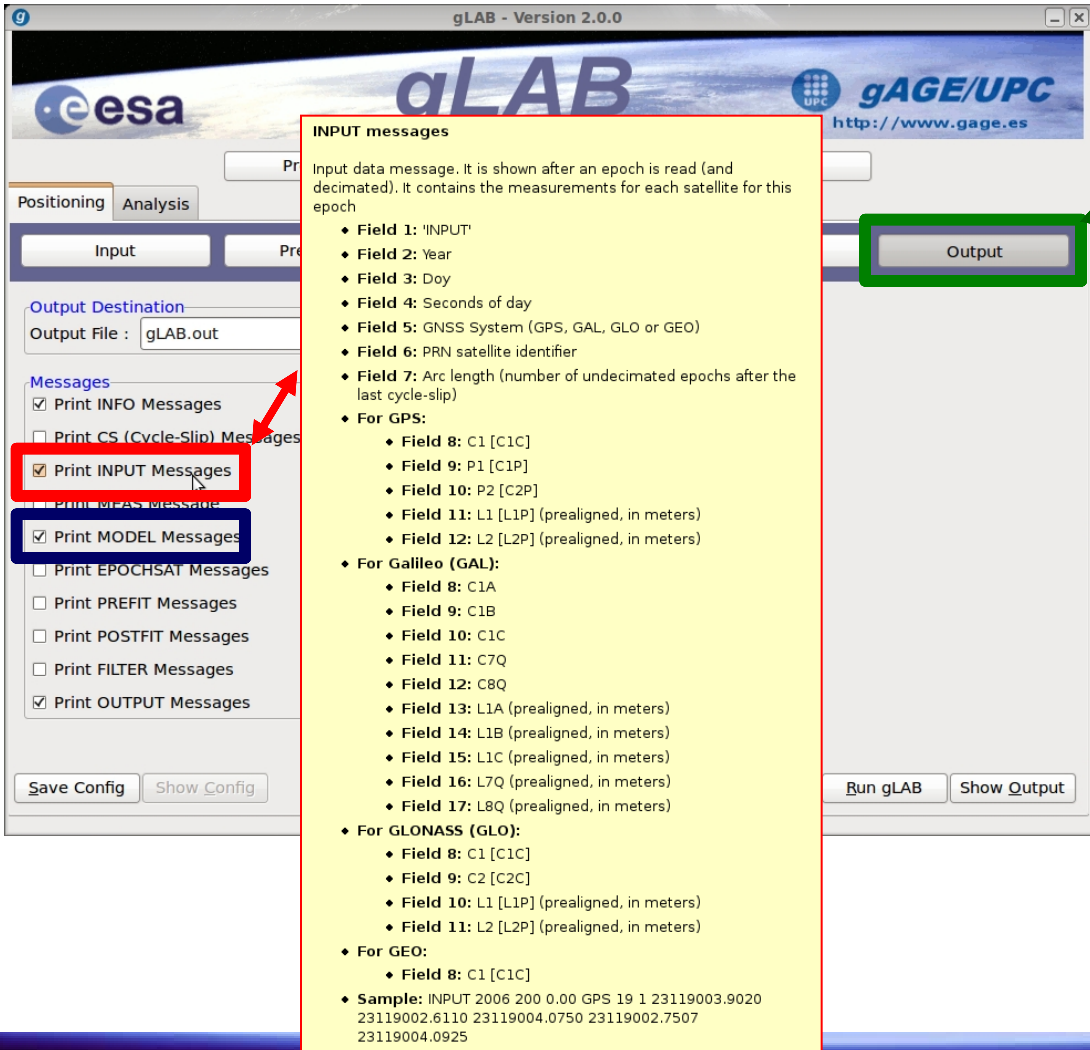
Note: UPC1-UPC2 receivers baseline: 37.95 metres.

gLAB processing in command line



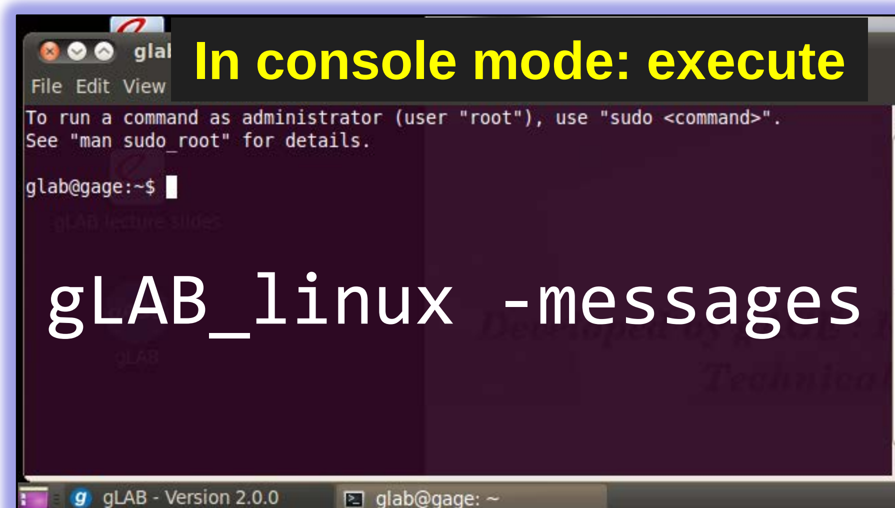
A "**notepad**" with the command line sentence is provided to facilitate the sentence writing: just **"copy" and " paste" from notepad to the working terminal.**

gLAB processing in command line



The different messages provided by ***gLAB*** and its content can be found in the [OUTPUT] section.

By placing the mouse on a given message name, a tooltip appears describing the different fields.



In console mode: execute

gLAB_linux -messages

OVERVIEW

- **Introduction:** gLAB processing in command line.
- **Preliminary computations: Data files.**
- **Session A:** Fixing DD ambiguities one at a time: UPC1-UPC2.
- **Session B:** Assessing the fixed ambiguities in navigation: Differential positioning of UPC1-UPC2 receivers.
- **Session C:** Fixing DD ambiguities with LAMBDA method.

Note: UPC1-UPC2 receivers baseline: 37.95 metres.

Previous

Preliminary Computations

P. Preliminary computations

- This section is devoted to prepare the data files to be used in the exercises.

- These data files will include the code and carrier measurements and the model components: geometric range, nominal troposphere and ionosphere corrections, satellite elevation and azimuth from each receiver…

- This data processing will be done with ***gLAB*** for each individual receiver.

- This preliminary processing will provide the baseline data files to perform computations easily using basic tools (such as **awk** for data files handling, to compute Double Differences of measurements) or using octave (MATLAB) scripts for the LAMBDA method implementation.

- Detailed **guidelines** for **self learning students** are provided in this tutorial and in its associated ***notepad*** text file.

P. Preliminary computations

P1. Model Components computation

- The script "**ObsFile.scr**" generates a data file with the following content

```
  1   2   3   4   5  6  7  8  9   10   11  12   13
[sta sat DoY sec P1 L1 P2 L2 Rho Trop Ion elev azim]
```

- Run this script for all receivers:

```
ObsFile.scr UPC10770.11o brdc0770.11n
ObsFile.scr UPC20770.11o brdc0770.11n
```

- Merge all files into a single file:

```
cat ????.obs > ObsFile.dat
```

P. Preliminary computations

Selecting measurements: Time interval [18000:19900]

- To simplify computations, a time interval with always the same set of satellites in view and without cycle-slips is selected.
- Moreover an elevation mask of 10 degrees will be applied.

If the satellites change or cycle-slips appear during the data processing interval, care with the associated parameters handling must be taken in the navigation filter. Set up new parameters when new satellites appear and treat the ambiguities as constant between cycle-slips and white noise when a cycle-slip happens.

P. Preliminary computations

Selecting measurements: Time interval [18000:19900]

- Select the satellites in the time interval [18000:19900] with elevation over 10º

```
cat ObsFile.dat|gawk '{if ($4>=18000 && $4<=19900 && $12>10) print $0}' >
obs.dat
```

- Reference satellite (over the time interval [18000:19900])

Confirm that the satellite PRN06 is the satellite with the highest elevation
(this satellite will be used as the reference satellite)

obs.dat →

1	2	3	4	5	6	7	8	9	10	11	12	13
[sta	sat	DoY	sec	P1	L1	P2	L2	Rho	Trop	Ion	elev	azim]

P2. **Double differences** between receivers and satellites computation

The script "**DDobs.scr**" computes the double differences between receivers and satellites from file **obs.dat**.

```
  1    2    3    4    5   6   7   8   9    10    11   12    13
[sta sat DoY sec P1 L1 P2 L2 rho Trop Ion elev azim]
```

For instance, the following sentence:

```
DDobs.scr obs.dat UPC1 UPC2 06 03
```

generates the file

```
-------------------  DD_{sta1}_{sta2}_{sat1}_{sat2}.dat ----------------------------
   1    2    3    4    5   6    7    8    9    10    11    12     13    14  15  16  17
[sta1 sta2 sat1 sat2 DoY sec DDP1 DDL1 DDP2 DDL2 DDRho DDTrop DDIon El1 Az1 El2 Az2]
                                                                   <---- sta2 ---->
-----------------------------------------------------------------------------------
```

Where the elevation (EL) and azimuth (AZ) are taken from station #2.
and where (EL1, AZ1) are for satellite #1 and (EL1, AZ1) are for satellite #2.

P. Preliminary computations

Compute the double differences between receivers **UPC1 (reference)** and UPC2 and satellites **PRN06 (reference)** and [PRN 03, 07,16, 18, 19, 21, 22, 24]

```
DDobs.scr obs.dat UPC1 UPC2 06 03
DDobs.scr obs.dat UPC1 UPC2 06 07
DDobs.scr obs.dat UPC1 UPC2 06 16
DDobs.scr obs.dat UPC1 UPC2 06 18
DDobs.scr obs.dat UPC1 UPC2 06 19
DDobs.scr obs.dat UPC1 UPC2 06 21
DDobs.scr obs.dat UPC1 UPC2 06 22
DDobs.scr obs.dat UPC1 UPC2 06 24
```

Merge the files in a single file and sort by time:

```
cat DD_UPC1_UPC2_06_??.dat|sort -n -k +6 > DD_UPC1_UPC2_06_ALL.dat
```

OVERVIEW

- **Introduction:** gLAB processing in command line.
- **Preliminary computations:** Data files.
- **Session A: Fixing DD ambiguities one at a time: UPC1-UPC2.**
- **Session B:** Assessing the fixed ambiguities in navigation:
  Differential positioning of UPC1-UPC2 receivers.
- **Session C:** Fixing DD ambiguities with LAMBDA method.

Note: UPC1-UPC2 receivers baseline: 37.95 metres.

A. Fixing DD ambiguities one at a time: UPC1-UPC2

- This exercise is devoted to study the ambiguity fixing using the *cascade method*, that is, fixing the ambiguities one at a time.

- In the first part, we are going to assess this approach for a single frequency receiver, trying to fix DDN1 and DDN2 independently.

- In the second part, we are going to assess this approach for dual frequency receivers, fixing first the wide-lane ambiguity DDNw and afterwards DDN1 and DDN2.
- The results (i.e. the DDN1 and DDN2 ambiguities) will be assessed in the next “Session B” by computing the navigation solution using the carrier phases repaired with the fixed DDN1 and DDN2 ambig.

- Finally, in Session C, the LAMBDA method will be applied for comparison.

Resolving ambiguities one at a Time: single Freq.

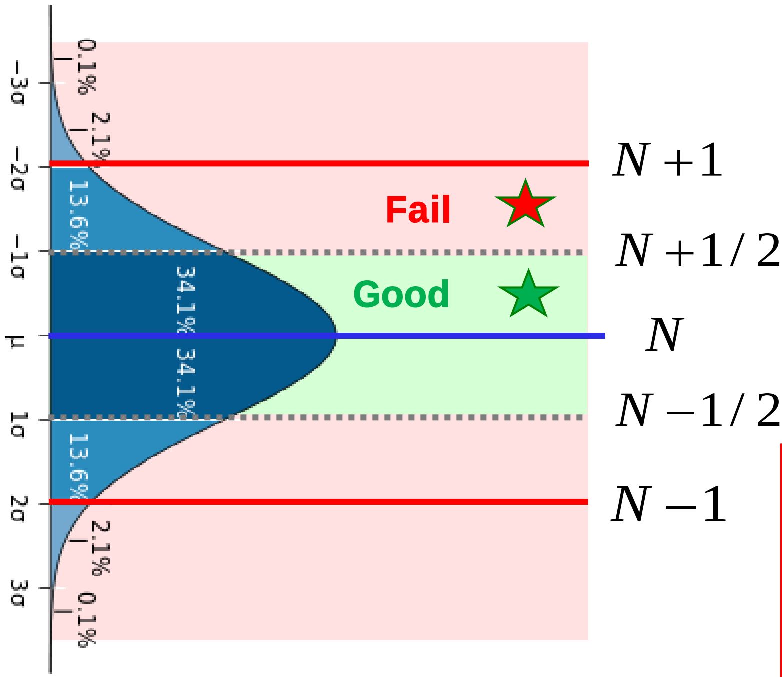
A simple trial would be (for instance using L1 and P1):

$$P_1^{jk} = \rho^{jk} + \nu_{P_1}^{jk}$$
$$L_1^{jk} = \rho^{jk} + \lambda_1 N_1^{jk} + \nu_{L_1}^{jk}$$

$$\rightarrow L_1^{jk} - P_1^{jk} = \lambda_1 N_1^{jk} + \nu_{P_1}^{jk} \rightarrow \hat{N}_1^{jk} = \left[\frac{L_1^{jk} - P_1^{jk}}{\lambda_1}\right]_{roundoff}$$

$$\lambda_1 \simeq 20\,\mathrm{cm}$$
$$\sigma_{P_1^{jk}} \approx 1\,\mathrm{m}$$
$$\sigma_{L_1^{jk}} \approx 1\,\mathrm{cm}$$

$$\sigma_{\hat{N}_1^{jk}} \simeq \frac{1}{\lambda_1}\sigma_{P_1^{jk}} \approx 5$$



To much error (5 wavelengths)!

Note that, assuming a Gaussian distribution of errors, $\sigma_{\hat{N}_1^{jk}} \simeq 1/2$ guarantee only the 68% of success

Similar results with *L2*, *P2* measurements

As the ambiguity is constant (between cycle-slips), we would try to reduce uncertainty by averaging the estimate on time, but we will need 100 epochs to reduce noise up to ½ (but measurement errors are highly correlated on time!)

A1 Trying to fix ambiguities in Single frequency

A1.1 Fixing N1 and N2 independently:

Estimate graphically values of DDN1 and DDN2 (i.e. try to identify the true ambiguity from the plot).

Hint:

From file **DD_UPC1_UPC2_06_ALL.dat**, generate the file **DDN1N2.dat** with the following content:

```
  1     2    3    4         5              6
[PRN  sec  DDN1  DDN2  nint(DDN1)  nint(DDN2)]
```

where:

$$DDN1=[DDL1 - DDP1]/\lambda_1; \; DDN2=[DDL2 - DDP2]/\lambda_2$$

Note:
"nint" means near-integer

Be careful: "**nint**" in **awk**:
nint(x) must be generated as:
int(x+0.5*sign(x))

A1 Trying to fix ambiguities in Single frequency

```
-----------------------------  DD_UPC1_UPC2_06_ALL.dat -------------------------
   1    2   3   4   5   6    7    8    9    10    11    12    13    14  15  16  17
[UPC1 UPC2 06 PRN DoY sec DDP1 DDL1 DDP2 DDL2 DDRho DDTrop DDIon El1 Az1 El2 Az2   ]
                                                                  <---- UPC2 ---->
------------------------------------------------------------------------------------
```

a) From file **DD_UPC1_UPC2_06_ALL.dat**, generate the file **DDN1N2.dat** with the following content:

```
  1     2   3    4          5           6
[PRN  sec DDN1 DDN2   nint(DDN1) nint(DDN2)]
```

Execute, for instance:

```
gawk 'BEGIN{c=299792458;f0=10.23e+6;l1=c/(154*f0);l2=c/(120*f0)}
     {A1=($8-$7)/l1; A2=($10-$9)/l2;if (A1!=0){signA1=A1/sqrt(A1*A1)}else{signA1=0};
                                    if (A2!=0) {signA2=A2/sqrt(A2*A2)}else{signA2=0};
 print $4,$6,A1,int(A1+0.5*signA1),A2,int(A2+0.5*signA2)}' DD_UPC1_UPC2_06_ALL.dat >
                                                                           DDN1N2.dat
```

A1 Trying to fix ambiguities in Single frequency

b) Plot DDN1 and DDN2 for the different satellites and discuss if the ambiguity DDN1 and DDN2 can be fixed:

```
  1    2    3    4        5           6
[PRN sec DDN1 DDN2  nint(DDN1) nint(DDN2)]
```

b1) DDN1 plot:

```
graph.py -f DDN1N2.dat -x2 -y3 -c '($1==16)' -s. -f DDN1N2.dat -x2 -y4 -c '($1==16)'
 -sx --cl r --yn -4 --yx 10 --xl "time (s)" --yl "cycles L1" -t "DDN1 ambiguity: PRN16"
```

b2) DDN2 plot:

```
graph.py -f DDN1N2.dat -x2 -y5 -c '($1==16)' -s. -f DDN1N2.dat -x2 -y6 -c '($1==16)'
 -sx --cl r --yn -8 --yx 6 --xl "time (s)" --yl "cycles L2" -t "DDN2 ambiguity: PRN16"
```

A1 Trying to fix ambiguities in Single frequency

b1) DDN1 plot:

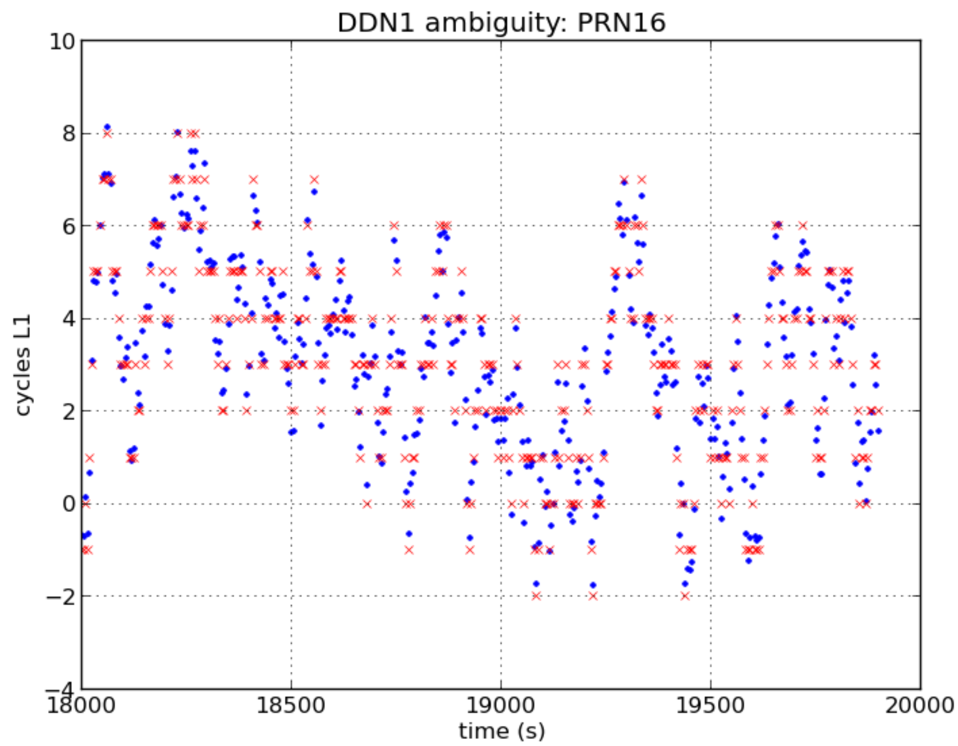
```
graph.py -f DDN1N2.dat -x2 -y3 -c '($1==16)' -s. -f DDN1N2.dat -x2 -y4 -c '($1==16)'
 -sx --cl r --yn -4 --yx 10 --xl "time (s)" --yl "cycles L1" -t "DDN1 ambiguity: PRN16"
```

Questions:

1.- Explain what is the meaning of this plot.

2.- Is it possible to identify the integer ambiguity?

3.- How reliability can be improved?



A1 Trying to fix ambiguities in Single frequency

b2) DDN2 plot:

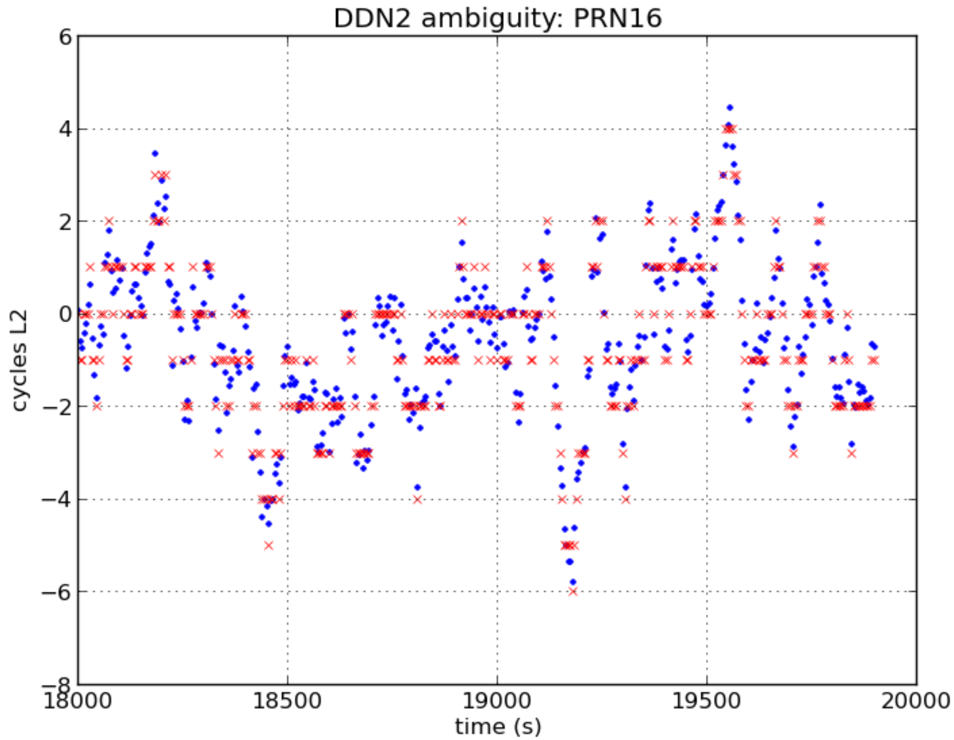
```
graph.py -f DDN1N2.dat -x2 -y5 -c '($1==16)' -s. -f DDN1N2.dat -x2 -y6 -c '($1==16)'
  -sx --cl r --yn -8 --yx 6 --xl "time (s)" --yl "cycles L2" -t "DDN2 ambiguity: PRN16"
```

Questions:

1.- Explain what is the meaning of this plot.

2.- Is it possible to identify the integer ambiguity?

3.- How reliability can be improved?



A1 Trying to fix ambiguities in Single frequency

```
------------------------------  DD_UPC1_UPC2_06_ALL.dat -------------------------
   1     2    3    4    5    6     7     8     9    10     11    12     13     14   15   16   17
[UPC1 UPC2 06 PRN DoY sec DDP1 DDL1 DDP2 DDL2 DDRho DDTrop DDIon El1 Az1 El2 Az2   ]
-------------------------------------------------------------------------------------
```

c) Make plots to analyze the DDP1 and DDP2 code noise.

Hint:

From file **DD_UPC1_UPC2_06_ALL.dat**, generate the file **P1P2noise.dat** with the following content:

```
   1     2     3           4
[PRN  sec  DDP1-DDRho  DDP2-DDRho]
```

Execute, for instance:

```
gawk '{print $4,$6,$7-$11,$9-$11}' DD_UPC1_UPC2_06_ALL.dat > P1P2noise.dat
```

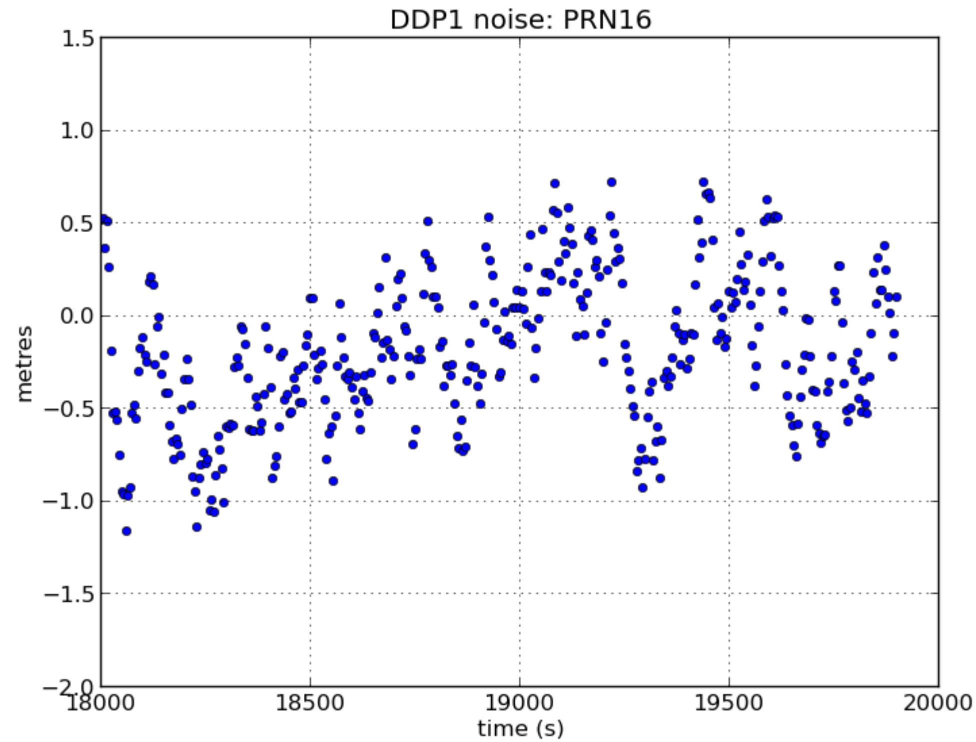
A1 Trying to fix ambiguities in Single frequency

c1) Depict the DDP1 code noise:

```
graph.py -f P1P2noise.dat -x2 -y3 -c '($1==16)' -so --yn -2 --yx 1.5 --xl "time (s)"
                                              --yl "metres" -t "DDP1 noise: PRN16"
```

Questions:

Discuss why the ambiguities cannot be fixed by rounding-off the expression DDN1=[DDL1 -DDP1]/λ_1



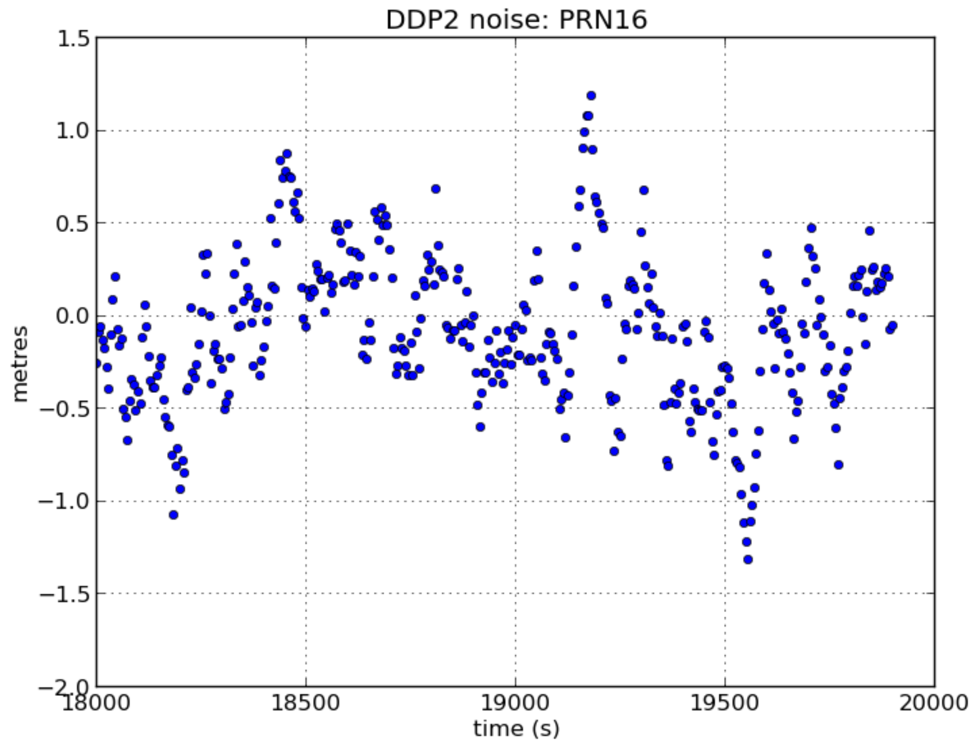
A1 Trying to fix ambiguities in Single frequency

c2) Depict the DDP2 code noise:

```
graph.py -f P1P2noise.dat -x2 -y4 -c '($1==16)' -so --yn -2 --yx 1.5 --xl "time (s)"
                                        --yl "metres" -t "DDP2 noise: PRN16"
```

Questions:

Discuss why the ambiguities cannot be fixed by rounding-off the expression $DDN2=[DDL2-DDP2]/\lambda_2$



A1 Trying to fix ambiguities in Single frequency

```
-----------------------------  DD_UPC1_UPC2_06_ALL.dat -------------------------
  1    2    3   4   5   6    7   8    9   10    11   12    13    14  15  16  17
[UPC1 UPC2 06 PRN DoY sec DDP1 DDL1 DDP2 DDL2 DDRho DDTrop DDIon El1 Az1 El2 Az2   ]
------------------------------------------------------------------------------------
```

d) Make plots to analyze the DDL1 and DDL2 carrier noise.

Hint:

From file **DD_UPC1_UPC2_06_ALL.dat**, generate the file **L1L2noise.dat** with the following content:

```
  1    2    3      4           5
[PRN sec DDL1-DDRho DDL2-DDRho]
```

Execute, for instance:

```
gawk '{print $4,$6,$8-$11,$10-$11}' DD_UPC1_UPC2_06_ALL.dat > L1L2noise.dat
```

A1 Trying to fix ambiguities in Single frequency

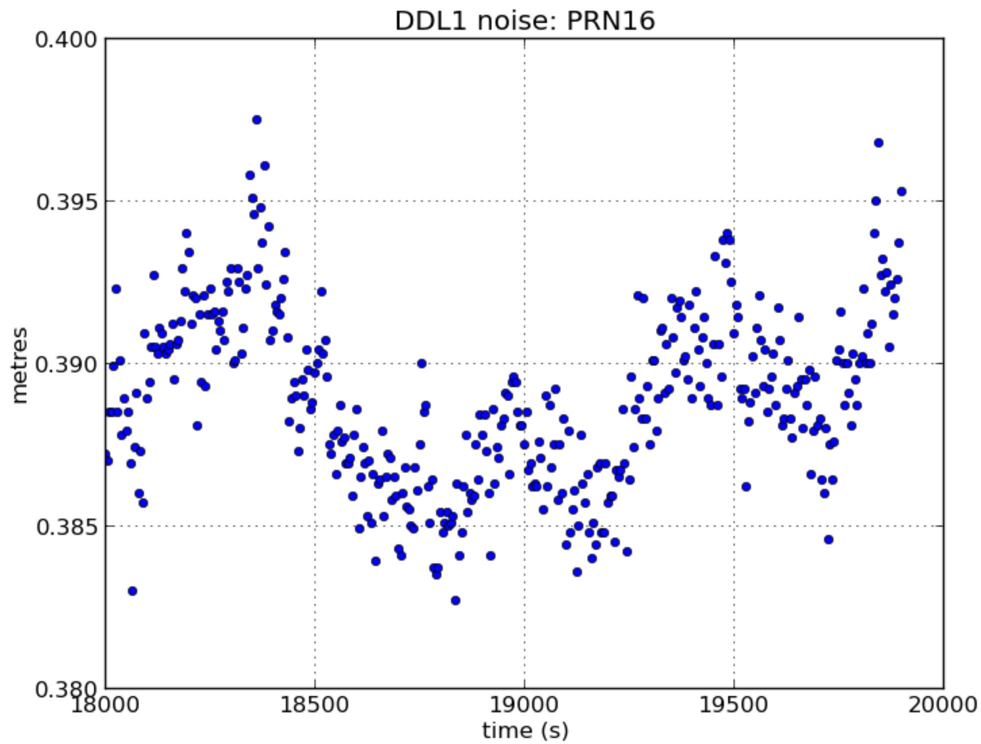
d1) Depict the DDL1 code noise:

```
graph.py -f L1L2noise.dat -x2 -y3 -c '($1==16)' -so --yn 0.38 --yx 0.40 --xl "time (s)"
                                                   --yl "metres" -t "DDL1 noise: PRN16"
```

Questions:

Discuss the plot.
What is the level of noise?

Compare the noise with the wavelength λ_1*=19.0cm*



A1 Trying to fix ambiguities in Single frequency

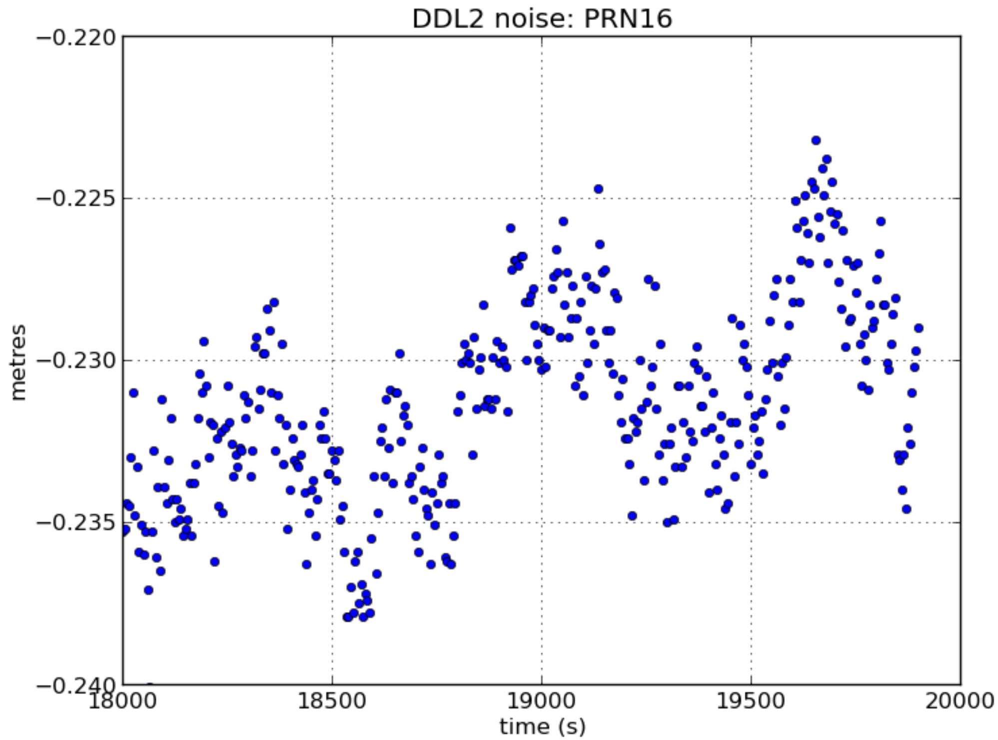
d2) Depict the DDL2 code noise:

```
graph.py –f L1L2noise.dat -x2 -y4 -c'($1==16)' -so --yn -0.24 --yx -0.22 --xl "time (s)"
                                            --yl "metres" -t "DDL2 noise: PRN16"
```

Questions:

Discuss the plot.
What is the level of noise?

Compare the noise with the wavelength λ_2=24.4cm



Resolving ambiguities one at a Time: Dual Freq.

Dual frequency measurements: wide-laning with the Melbourne-Wübbena combination

$$P_1^{jk} = \rho^{jk} + \nu_{P_1}^{jk}$$
$$P_2^{jk} = \rho^{jk} + \nu_{P_2}^{jk}$$
$$L_1^{jk} = \rho^{jk} + \lambda_1 N_1^{jk} + \nu_{L_1}^{jk}$$
$$L_2^{jk} = \rho^{jk} + \lambda_2 N_2^{jk} + \nu_{L_2}^{jk}$$

$$P_N^{jk} = \frac{f_1 P_1^{jk} + f_2 P_2^{jk}}{f_1 + f_2} = \rho^{jk} + \nu_{P_N}^{jk}$$
$$L_W^{jk} = \frac{f_1 L_1^{jk} - f_2 L_2^{jk}}{f_1 - f_2} = \rho^{jk} + \lambda_W N_W^{jk} + \nu_{L_W}^{jk}$$

$$N_W = N_1 - N_2$$
$$\lambda_W = \frac{c}{f_1 - f_2} \simeq 86.2\,\mathrm{cm}$$

$$\sigma_{P_N^{jk}} \approx \sigma_{P_1^{jk}} / \sqrt{2} \approx 71\,\mathrm{cm}$$
$$\sigma_{L_W^{jk}} \approx 6\,\sigma_{L_1^{jk}} \approx 6\,\mathrm{cm}$$

$$\downarrow$$

$$L_W^{jk} - P_N^{jk} = \lambda_W N_W^{jk} + \nu_{P_N}^{jk} \rightarrow \hat{N}_W^{jk} = \left[\frac{L_W^{jk} - P_N^{jk}}{\lambda_W}\right]_{roundoff}$$

$$\sigma_{\hat{N}_W^{jk}} \approx \frac{1}{\lambda_W}\sigma_{P_N^{jk}} \simeq \frac{71cm}{86.2cm} \simeq 0.8$$

Now, with uncorrelated measurements from 10 epochs will reduce noise up to about ¼.

Fixing N_1 (after fixing N_W)

$$L_1^{jk} - L_2^{jk} = \lambda_1 N_1^{jk} - \lambda_2 N_2^{jk} + \nu_{L_1-L_2}^{jk}$$
$$= (\lambda_1 - \lambda_2) N_1^{jk} + \lambda_2 N_W^{jk} + \nu_{L_1-L_2}^{jk}$$

$$\lambda_1 = 19.0\,\mathrm{cm}$$
$$\lambda_2 = 24.4\,\mathrm{cm}$$
$$\lambda_2 - \lambda_1 = 5.4\,\mathrm{cm}$$
$$\sigma_{L_1^{jk}} \approx 1\,\mathrm{cm}$$

$$\hat{N}_1^{jk} = \left[\frac{L_1^{jk} - L_2^{jk} - \lambda_2 \hat{N}_W^{jk}}{\lambda_1 - \lambda_2}\right]_{roundoff}$$

$$\hat{N}_2^{jk} = \hat{N}_1^{jk} - \hat{N}_W^{jk}$$

$$\sigma_{\hat{N}_1^{jk}} \approx \frac{1}{\lambda_1 - \lambda_2}\sqrt{2}\,\sigma_{L_1^{jk}} \simeq \frac{1.4cm}{5.4cm} \simeq 1/4$$

A2.1 Fixing Wide-lane ambiguity (Nw):

Estimate graphically values of DDNw (i.e. try to identify the true ambiguity from the plot).

Hint:

From file **DD_UPC1_UPC2_06_ALL.dat**, generate the file **DDNw.dat** with the following content:

1	2	3	4
[PRN	sec	DDNw	nint(DDNw)]

where: DDN1=[DDL$_W$ -DDP$_N$]/λ_W

Note:

$$L_W = \frac{\beta L_1 - L_2}{\beta - 1} \quad ; \quad P_N = \frac{\beta P_1 + P_2}{\beta + 1} \quad ; \quad \beta = \frac{f_1}{f_2} = \frac{154}{120} \quad ; \quad \lambda_W = \frac{c}{f_1 - f_2} = 86.2cm$$

A2 Dual Frequency Ambiguity Fixing

```
----------------------------- DD_UPC1_UPC2_06_ALL.dat -------------------------
   1     2    3    4    5    6     7     8     9    10     11    12     13     14   15   16   17
[UPC1 UPC2 06 PRN DoY sec DDP1 DDL1 DDP2 DDL2 DDRho DDTrop DDIon El1 Az1 El2 Az2   ]
----------------------------------------------------------------------------------
```

a) From file **DD_UPC1_UPC2_06_ALL.dat**, generate the file **DDNw.dat** with the following content:

```
  1     2     3         4
[PRN  sec   DDNw    nint(DDNw)]
```

where: $DDN1=[DDL_W - DDP_N]/\lambda_W$

Execute, for instance:

```
cat DD_UPC1_UPC2_06_ALL.dat| gawk 'BEGIN{s12=154/120}
 {mw=(s12*$8-$10)/(s12-1)-(s12*$7+$9)/(s12+1);if(mw!=0){sign=mw/sqrt(mw*mw)}
else{sign=0};printf "%02i %i %14.4f %i \n", $4,$6, mw/0.862,int(mw/0.862+0.5*sign)}'>
                                                                              DDNw.dat
```

A2 Dual Frequency Ambiguity Fixing

b) Plot DDNw for the different satellites and discuss if the ambiguity DDNw can be fixed:

```
   1    2     3        4
[PRN  sec   DDNw   nint(DDNw)]
```

- Example PRN03 plot:

```
graph.py -f DDNw.dat -x2 -y3 -c '($1==03)' -s. -f DDNw.dat -x2 -y4 -c '($1==03)'
 -sx --cl r  --yn -1 --yx 7 --xl "time (s)" --yl "cycles Lw" -t "DDNw ambiguity: PRN03"
```

- Example PRN07 plot:

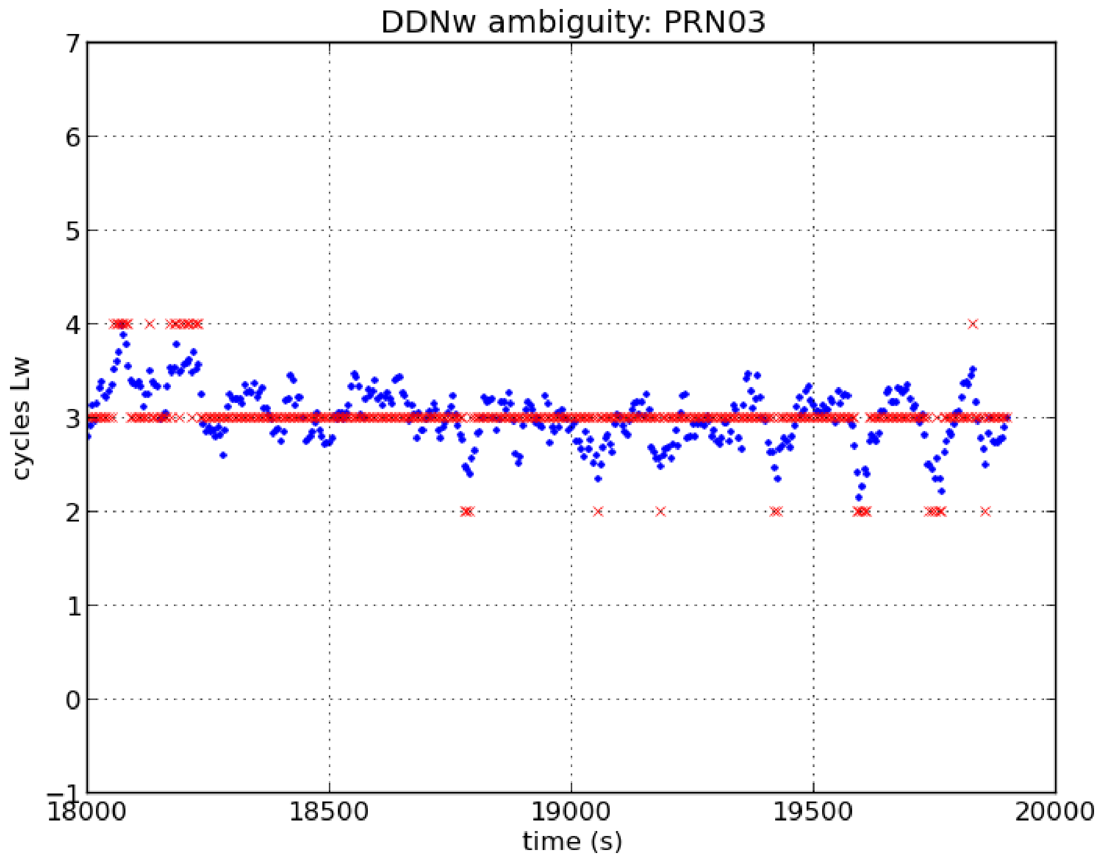
```
graph.py -f DDNw.dat -x2 -y3 -c '($1==07)' -s. -f DDNw.dat -x2 -y4 -c '($1==07)'
 -sx --cl r  --yn -4 --yx 4 --xl "time (s)" --yl "cycles Lw" -t "DDNw ambiguity: PRN07"
```

A2 Dual Frequency Ambiguity Fixing

```
graph.py -f DDNw.dat -x2 -y3 -c '($1==03)' -s. -f DDNw.dat -x2 -y4 -c '($1==03)'
 -sx --cl r  --yn -1 --yx 7 --xl "time (s)" --yl "cycles Lw" -t "DDNw ambiguity: PRN03"
```

PRN03 plot:

➔ DDNw= 3 ?

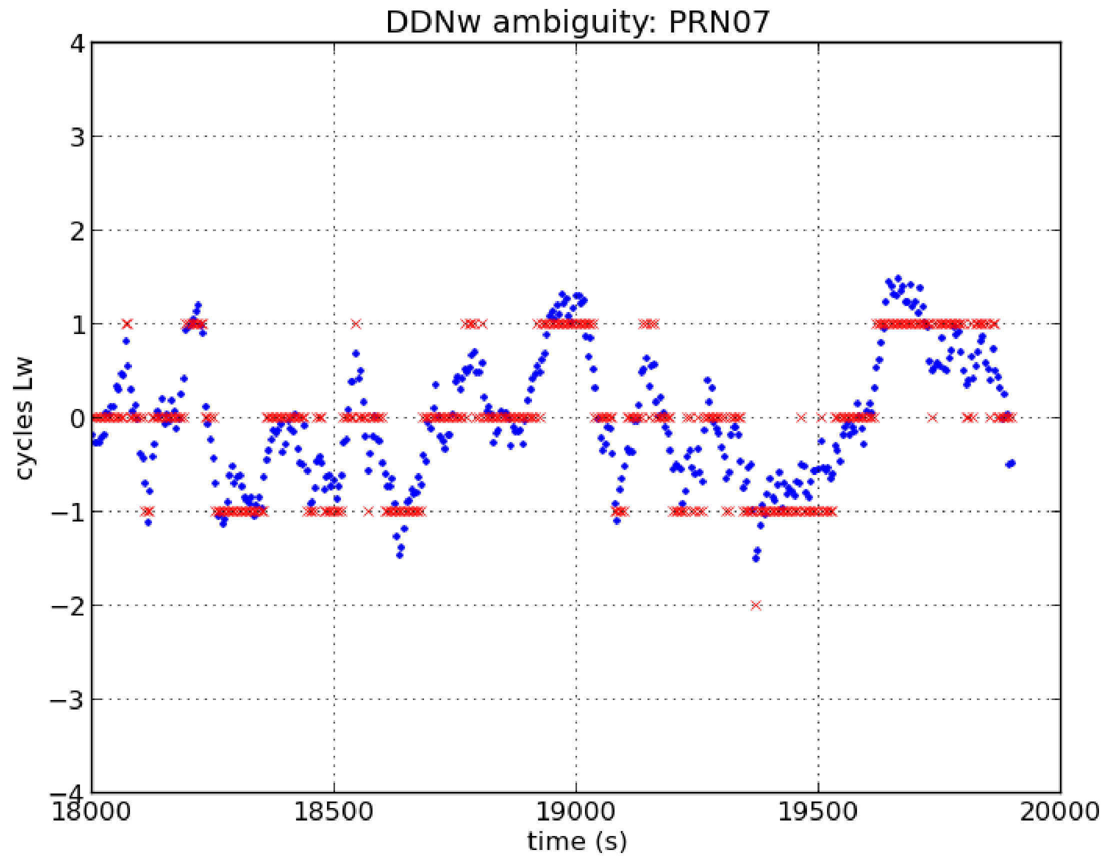


A2 Dual Frequency Ambiguity Fixing

```
graph.py -f DDNw.dat -x2 -y3 -c '($1==07)' -s. -f DDNw.dat -x2 -y4 -c '($1==07)'
  -sx --cl r  --yn -4 --yx 4 --xl "time (s)" --yl "cycles Lw" -t "DDNw ambiguity: PRN07"
```

PRN07 plot:

➔ DDNw= 0 ?



A2 Dual Frequency Ambiguity Fixing

The remaining plots:

```
graph.py -f DDNw.dat -x2 -y3 -c '($1==16)' -s. -f DDNw.dat -x2 -y4 -c '($1==16)'
  -sx --cl r  --yn -1 --yx 7 --xl "time (s)" --yl "cycles Lw" -t "DDNw ambiguity: PRN16"
```

```
graph.py -f DDNw.dat -x2 -y3 -c '($1==18)' -s. -f DDNw.dat -x2 -y4 -c '($1==18)'
  -sx --cl r  --yn -7 --yx 1 --xl "time (s)" --yl "cycles Lw" -t "DDNw ambiguity: PRN18"
```

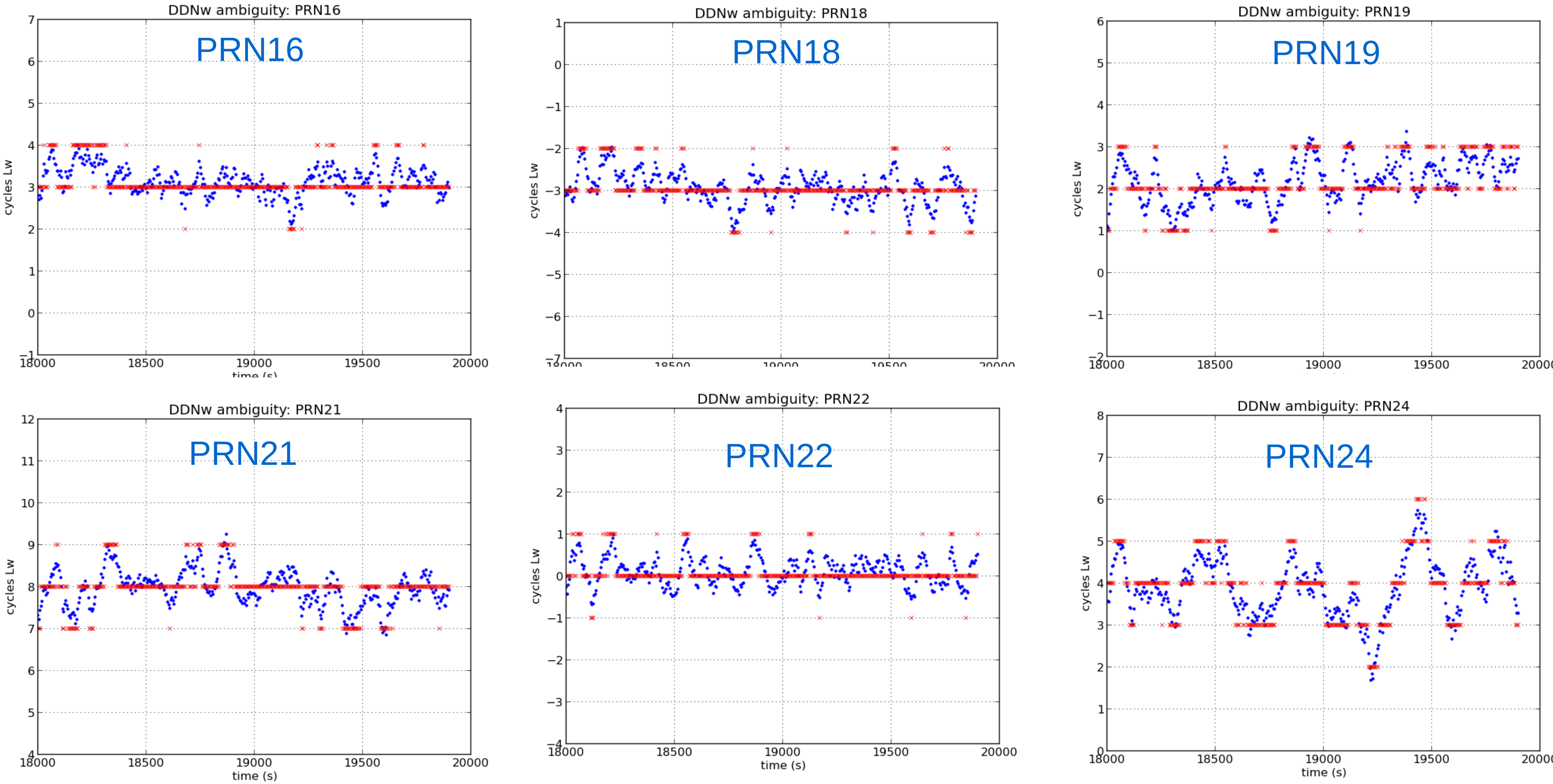
```
graph.py -f DDNw.dat -x2 -y3 -c '($1==19)' -s. -f DDNw.dat -x2 -y4 -c '($1==19)'
  -sx --cl r  --yn -2 --yx 6 --xl "time (s)" --yl "cycles Lw" -t "DDNw ambiguity: PRN19"
```

```
graph.py -f DDNw.dat -x2 -y3 -c '($1==21)' -s. -f DDNw.dat -x2 -y4 -c '($1==21)'
  -sx --cl r  --yn 4 --yx 12 --xl "time (s)" --yl "cycles Lw" -t "DDNw ambiguity: PRN21"
```

```
graph.py -f DDNw.dat -x2 -y3 -c '($1==22)' -s. -f DDNw.dat -x2 -y4 -c '($1==22)'
  -sx --cl r  --yn -4 --yx 4 --xl "time (s)" --yl "cycles Lw" -t "DDNw ambiguity: PRN22"
```

```
graph.py -f DDNw.dat -x2 -y3 -c '($1==24)' -s. -f DDNw.dat -x2 -y4 -c '($1==24)'
   -sx --cl r  --yn 0 --yx 8 --xl "time (s)" --yl "cycles Lw" -t "DDNw ambiguity: PRN24"
```

A2 Dual Frequency Ambiguity Fixing



A.2.2 Smoothing the DDNw:

Smooth the DDNw with a 300 seconds sliding window, in order to improve the ambiguity fixing.

Hint: From file **DD_UPC1_UPC2_06_ALL.dat**, generate the file **DDNws.dat** with the content:

```
  1    2   3     4         5          6
[PRN sec DDNw  DDNws nint(DDNw) nint(DDNws)]
```

Execute, for instance (to smooth the DDNw):

```
cat DDNw.dat|gawk '{dt=300;for (j=0;j<dt;j++) {t=j+dt/2+int(($2-j)/dt)*dt;ind=$1" "t;
                          n[ind]++;v[ind]=$3;m[ind]=m[ind]+$3}}
       END{for (i in n) {if (n[i]!=0) {;if(n[i]>1) {val=v[i];mean=m[i]/n[i];
                                     print i,val,mean}}}}' |  sort -n -k+1 > DDNws.tmp
```

Estimate again the ambiguity from the raw DDNw and smoothed DDNws:

```
cat DDNws.tmp | gawk '{sign3=$3/sqrt($3*$3);sign4=$4/sqrt($4*$4);
                  print $1,$2,$3,$4,int($3+0.5*sign3),int($4+0.5*sign4)}' > DDNws.dat
```

A2 Dual Frequency Ambiguity Fixing

b) Plot DDNw and DDNws for the different satellites and discuss if the ambiguity DDNw can be fixed:

```
   1     2    3      4          5            6
[PRN sec DDNw  DDNws nint(DDNw) nint(DDNws)]
```

- Example: PRN03 plot:

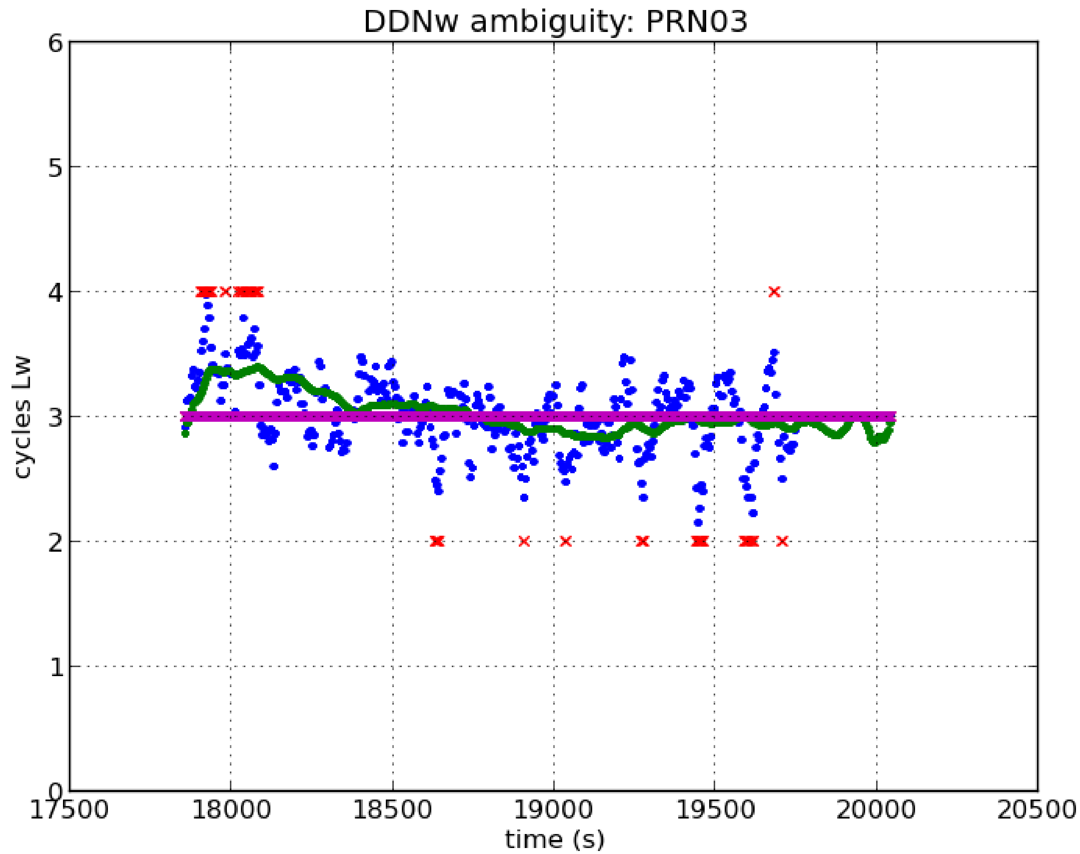
```
graph.py -f DDNws.dat -x2 -y3 -c '($1==03)' -s.
         -f DDNws.dat -x2 -y5 -c '($1==03)' -sx --cl r
         -f DDNws.dat -x2 -y4 -c '($1==03)' -s. --cl g
         -f DDNws.dat -x2 -y6 -c '($1==03)' -sx --cl m
         --yn 0 --yx 6 --xl "time (s)" --yl "cycles Lw" -t "DDNw ambig. PRN03"
```

A2 Dual Frequency Ambiguity Fixing

```
graph.py -f DDNws.dat -x2 -y3 -c '($1==03)' -s.
         -f DDNws.dat -x2 -y5 -c '($1==03)' -sx --cl r
         -f DDNws.dat -x2 -y4 -c '($1==03)' -s. --cl g
         -f DDNws.dat -x2 -y6 -c '($1==03)' -sx --cl m
         --yn 0 --yx 6 --xl "time (s)" --yl "cycles Lw" -t "DDNw ambig. PRN03"
```

PRN03 plot:

➔ DDNw=3

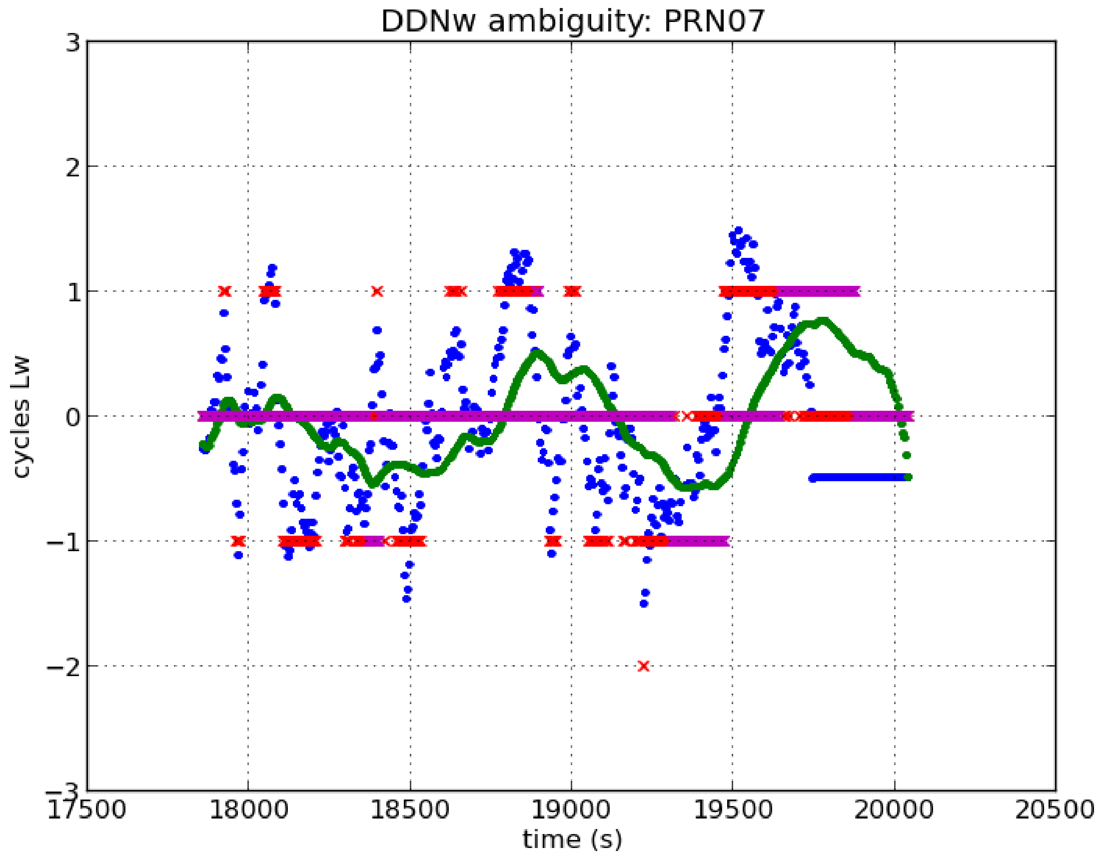


A2 Dual Frequency Ambiguity Fixing

```
graph.py -f DDNws.dat -x2 -y3 -c '($1==07)' -s.
         -f DDNws.dat -x2 -y5 -c '($1==07)' -sx --cl r
         -f DDNws.dat -x2 -y4 -c '($1==07)' -s. --cl g
         -f DDNws.dat -x2 -y6 -c '($1==07)' -sx --cl m
         --yn -3 --yx 3 --xl "time (s)" --yl "cycles Lw" -t "DDNw ambig. PRN07"
```

PRN07 plot:

➔ DDNw=0

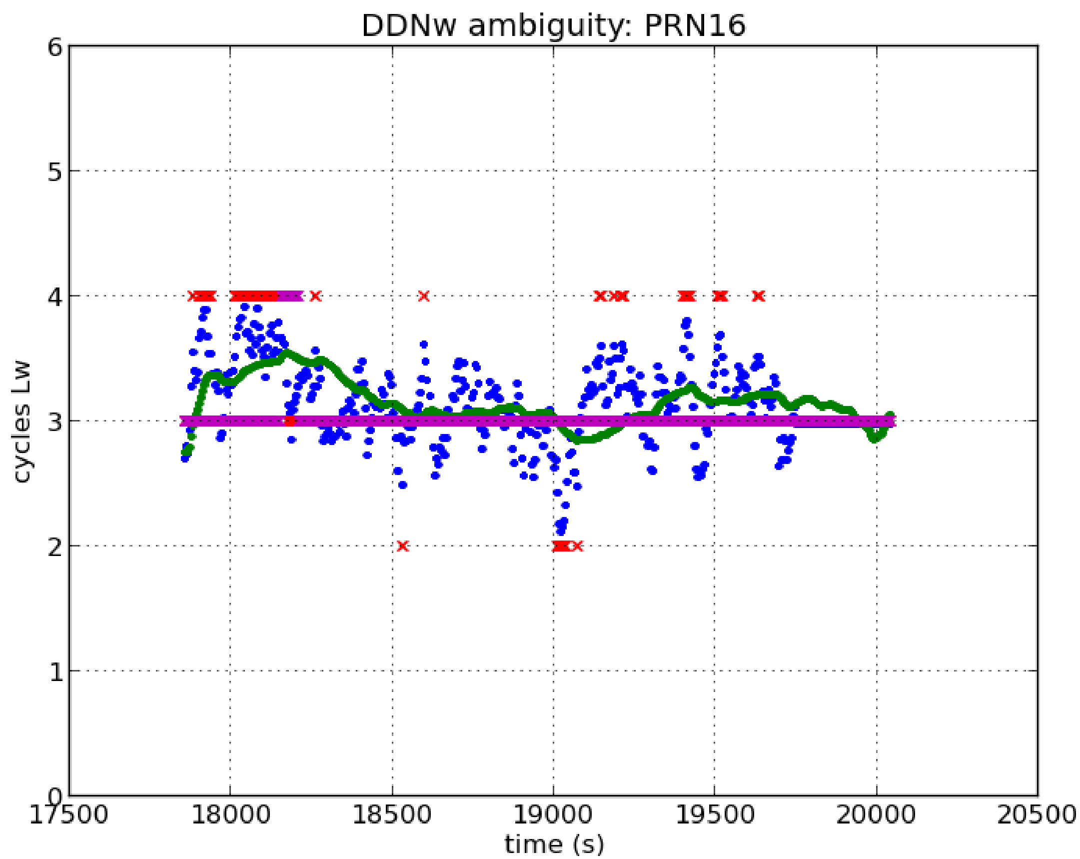


A2 Dual Frequency Ambiguity Fixing

```
graph.py -f DDNws.dat -x2 -y3 -c '($1==16)' -s.
         -f DDNws.dat -x2 -y5 -c '($1==16)' -sx --cl r
         -f DDNws.dat -x2 -y4 -c '($1==16)' -s. --cl g
         -f DDNws.dat -x2 -y6 -c '($1==16)' -sx --cl m
         --yn 0 --yx 6 --xl "time (s)" --yl "cycles Lw" -t "DDNw ambig. PRN16"
```

PRN16 plot:

➔ DDNw=3

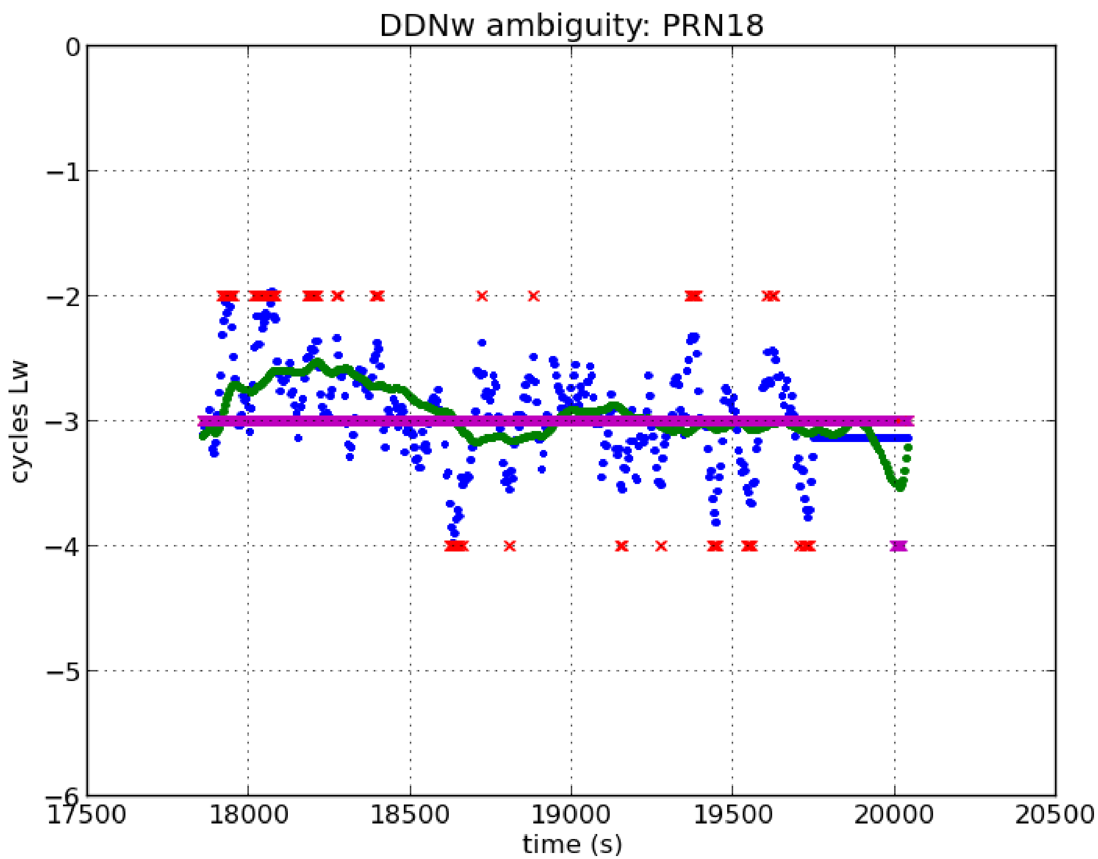


A2 Dual Frequency Ambiguity Fixing

```
graph.py -f DDNws.dat -x2 -y3 -c '($1==18)' -s.
         -f DDNws.dat -x2 -y5 -c '($1==18)' -sx --cl r
         -f DDNws.dat -x2 -y4 -c '($1==18)' -s. --cl g
         -f DDNws.dat -x2 -y6 -c '($1==18)' -sx --cl m
         --yn -6 --yx 0 --xl "time (s)" --yl "cycles Lw" -t "DDNw ambig. PRN18"
```

PRN18 plot:

➔ DDNw=-3

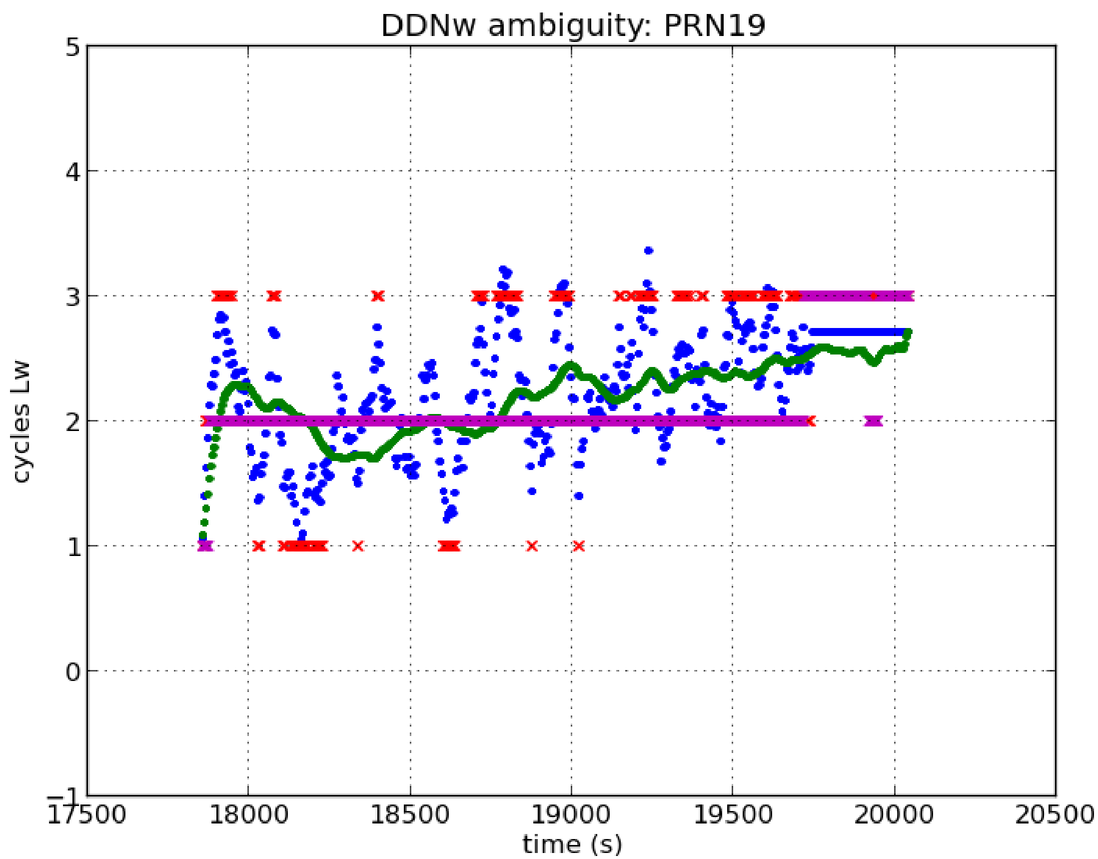


A2 Dual Frequency Ambiguity Fixing

```
graph.py -f DDNws.dat -x2 -y3 -c '($1==19)' -s.
         -f DDNws.dat -x2 -y5 -c '($1==19)' -sx --cl r
         -f DDNws.dat -x2 -y4 -c '($1==19)' -s. --cl g
         -f DDNws.dat -x2 -y6 -c '($1==19)' -sx --cl m
         --yn -1 --yx 5 --xl "time (s)" --yl "cycles Lw" -t "DDNw ambig. PRN19"
```

PRN19 plot:

→ DDNw=2

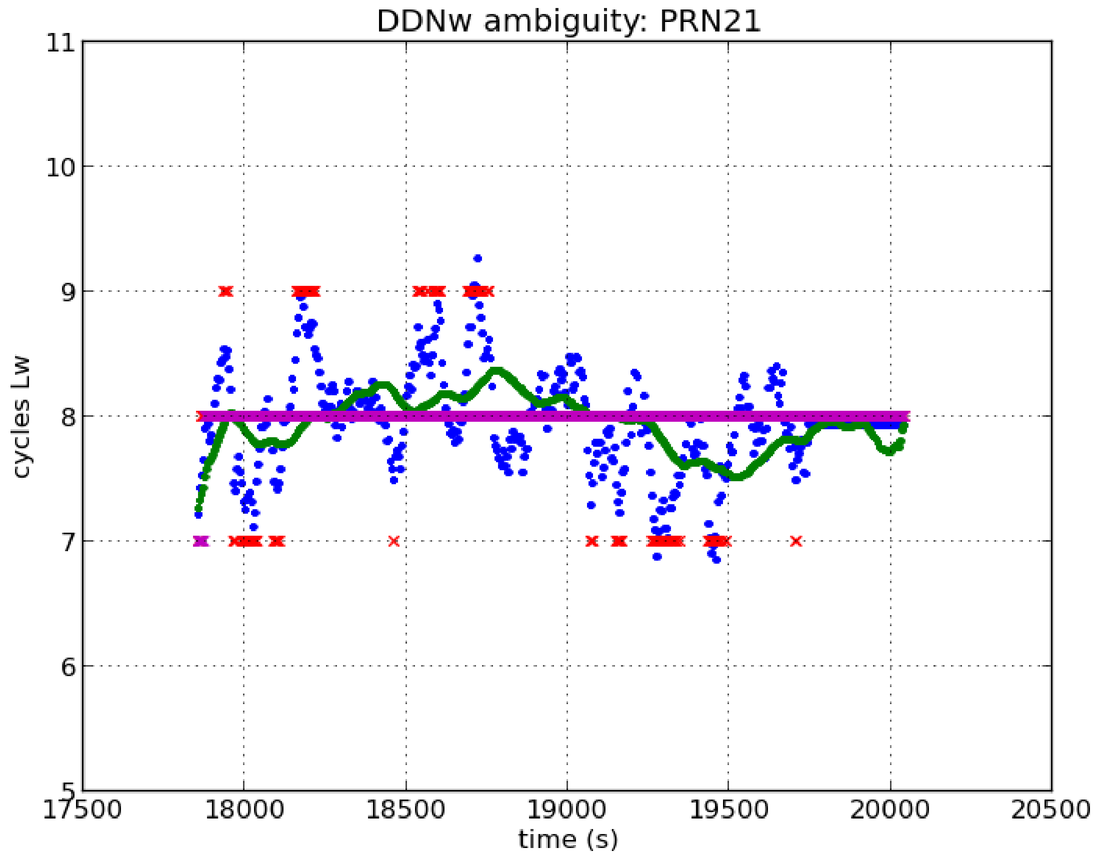


A2 Dual Frequency Ambiguity Fixing

```
graph.py -f DDNws.dat -x2 -y3 -c '($1==21)' -s.
         -f DDNws.dat -x2 -y5 -c '($1==21)' -sx --cl r
         -f DDNws.dat -x2 -y4 -c '($1==21)' -s. --cl g
         -f DDNws.dat -x2 -y6 -c '($1==21)' -sx --cl m
         --yn 5 --yx 11 --xl "time (s)" --yl "cycles Lw" -t "DDNw ambig. PRN21"
```

PRN21 plot:

➔ DDNw=8

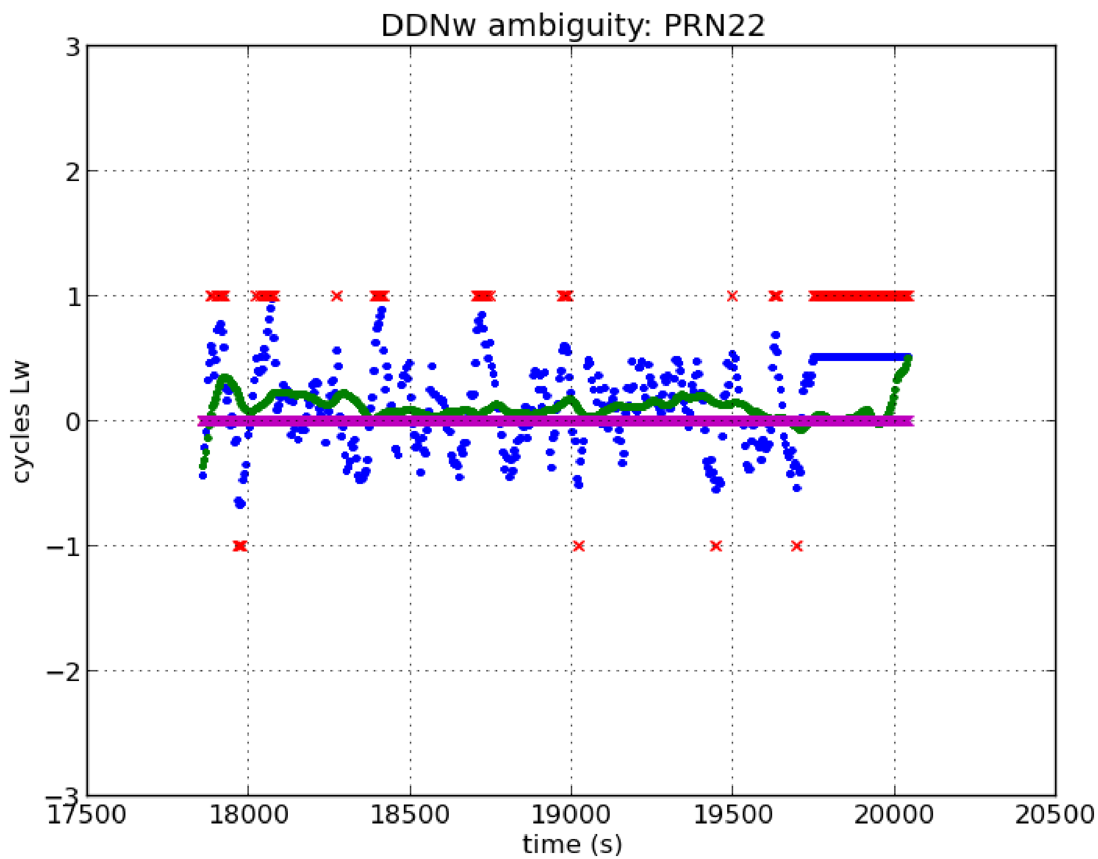


A2 Dual Frequency Ambiguity Fixing

```
graph.py -f DDNws.dat -x2 -y3 -c '($1==22)' -s.
         -f DDNws.dat -x2 -y5 -c '($1==22)' -sx --cl r
         -f DDNws.dat -x2 -y4 -c '($1==22)' -s. --cl g
         -f DDNws.dat -x2 -y6 -c '($1==22)' -sx --cl m
         --yn -3 --yx 3 --xl "time (s)" --yl "cycles Lw" -t "DDNw ambig. PRN22"
```

PRN22 plot:

➔ DDNw=0

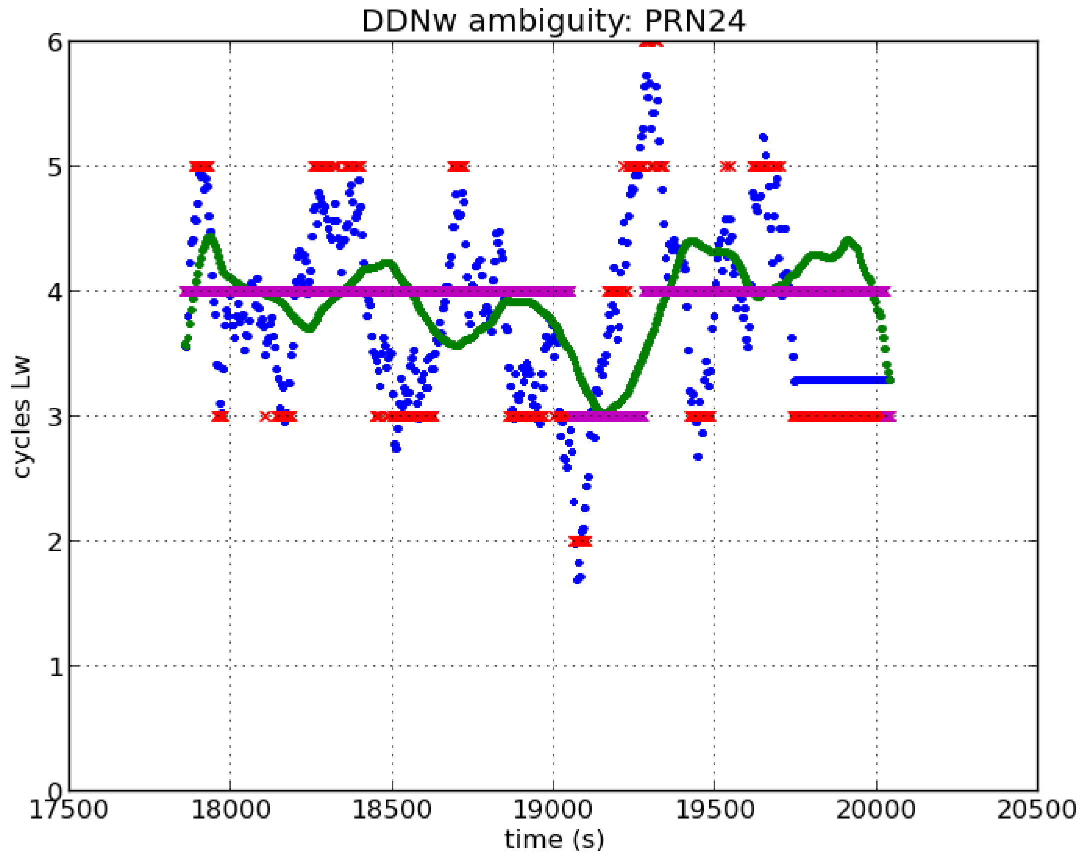


A2 Dual Frequency Ambiguity Fixing

```
graph.py -f DDNws.dat -x2 -y3 -c '($1==24)' -s.
         -f DDNws.dat -x2 -y5 -c '($1==24)' -sx --cl r
         -f DDNws.dat -x2 -y4 -c '($1==24)' -s. --cl g
         -f DDNws.dat -x2 -y6 -c '($1==24)' -sx --cl m
         --yn 0 --yx 6 --xl "time (s)" --yl "cycles Lw" -t "DDNw ambig. PRN24"
```

PRN24 plot:

➔ DDNw=4



A2.2 Fixing DDN1 from DDNw and DDL1, DDL2

Using the previous DDNw fixed values, estimate graphically the DDN1 ambiguity (i.e. try to identify the true ambiguity from the plot).

Hint:

From file **DD_UPC1_UPC2_06_ALL.dat**, and using the fixed DDNw, generate the file **DDN1.dat** with the following content:

```
  1    2   3      4
[PRN sec DDN1  nint(DDN1)]
```

where:

$$DDN1=(DDL1-DDL2-\lambda_2\, DDNw)/(\lambda_1-\lambda_2)$$

```
PRN = [ 03  07  16  18  19  21  22  24]
DDNw= [  3   0   3  -3   2   8   0   4]
```

A2 Dual Frequency Ambiguity Fixing

```
-----------------------------  DD_UPC1_UPC2_06_ALL.dat -------------------------
   1     2    3    4    5    6     7     8     9    10     11    12     13     14   15   16   17
[UPC1 UPC2 06 PRN DoY sec DDP1 DDL1 DDP2 DDL2 DDRho DDTrop DDIon El1 Az1 El2 Az2   ]
------------------------------------------------------------------------------------
```

From file **DD_UPC1_UPC2_06_ALL.dat**, and using the fixed DDNw, generate the file **DDN1.dat** with the following content:

```
   1     2    3        4
[PRN sec DDN1   nint(DDN1)]
```

where:

$$DDN1=(DDL1-DDL2-\lambda_2\, DDNw)/(\lambda_1-\lambda_2)$$

Execute, for instance for PRN**24** (with **DDNw=4**):

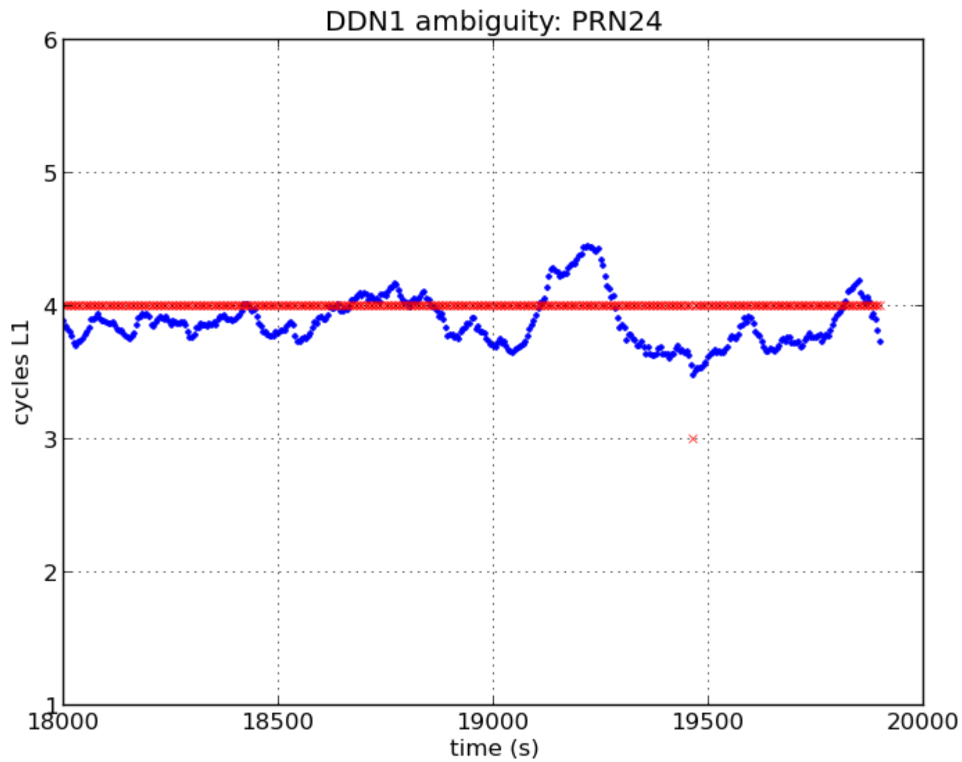
```
gawk 'BEGIN{c=299792458;f0=10.23e+6;f1=154*f0;f2=120*f0;l1=c/f1;l2=c/f2}
                                     {Nw=4;if ($4==24) {amb=($8-$10-l2*Nw)/(l1-l2);
  print $4,$6,amb,int(amb+0.5*amb/sqrt(amb*amb))}}' DD_UPC1_UPC2_06_ALL.dat  >
                                                                          DDN1_PRN24
```

A2 Dual Frequency Ambiguity Fixing

```
graph.py -f DDN1_PRN24 -x2 -y3 -c '($1==24)' -s.
         -f DDN1_PRN24 -x2 -y4 -c '($1==24)' -sx
         --cl r --yn 1 --yx 6 --xl "time (s)"
         --yl "cycles L1" -t "DDN1 ambiguity: PRN24"
```

PRN24 plot:

➔ DDN1=4



A2 Dual Frequency Ambiguity Fixing

```
----------------------------  DD_UPC1_UPC2_06_ALL.dat -------------------------
   1     2    3    4    5    6     7     8     9    10     11    12     13     14   15   16   17
[UPC1 UPC2 06 PRN DoY sec DDP1 DDL1 DDP2 DDL2 DDRho DDTrop DDIon El1 Az1 El2 Az2   ]
--------------------------------------------------------------------------------------
```

From file **DD_UPC1_UPC2_06_ALL.dat**, and using the fixed DDNw, generate the file **DDN1.dat** with the following content:

```
  1     2    3        4
[PRN sec DDN1   nint(DDN1)]
```

Other possibility is to execute the following sentence to generate the file for all satellites:

```
gawk 'BEGIN{for (i=0;i<100;i++) {getline <"sat.ambNw";Nw[$1*1]=$2}}
{c=299792458;f0=10.23e+6;f1=154*f0;f2=120*f0;l1=c/f1;l2=c/f2}
{amb=($8-$10-l2*Nw[$4*1])/(l1-l2);if (amb!=0){sign=amb/sqrt(amb*amb)} else{sign=0};
               print $4,$6,amb,int(amb+0.5*sign)}' DD_UPC1_UPC2_06_ALL.dat > DDN1.dat
```

Where **sat.ambNw** is a file containing the DDNw ambiguities:

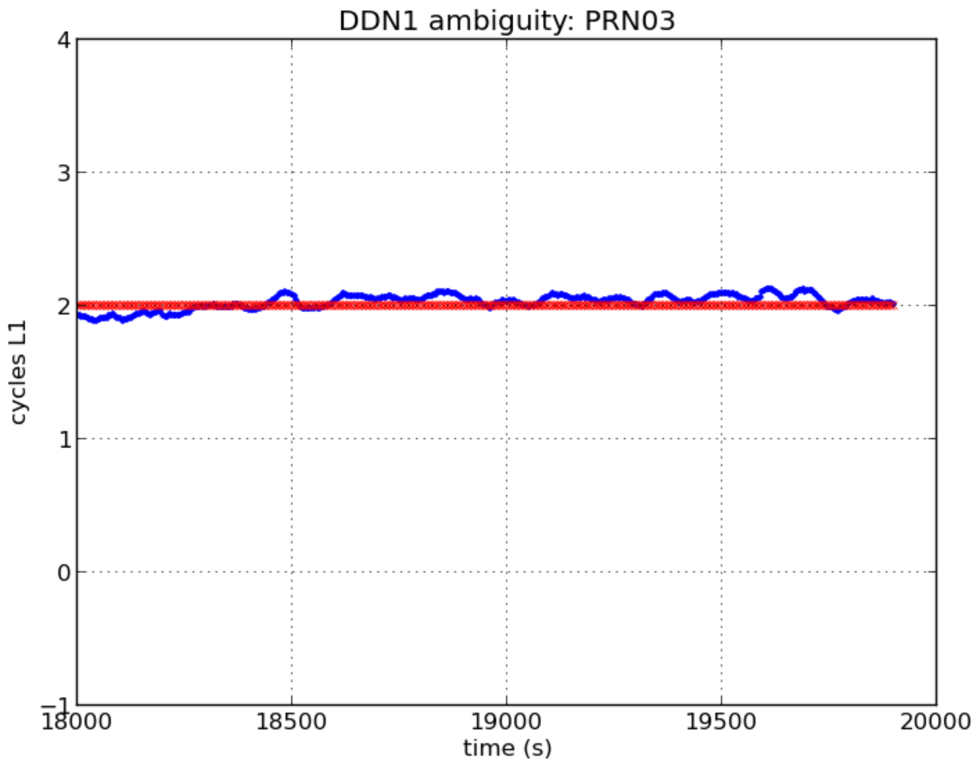
```
03  3
07  0
16  3
18 -3
19  2
21  8
22  0
24  4
```

A2 Dual Frequency Ambiguity Fixing

```
graph.py -f DDN1_PRN24 -x2 -y3 -c '($1==03)' -s.
         -f DDN1_PRN24 -x2 -y4 -c '($1==03)' -sx
         --cl r --yn -1 --yx 4 --xl "time (s)"
         --yl "cycles L1" -t "DDN1 ambiguity: PRN03"
```

PRN03 plot:

➔ DDN1=2

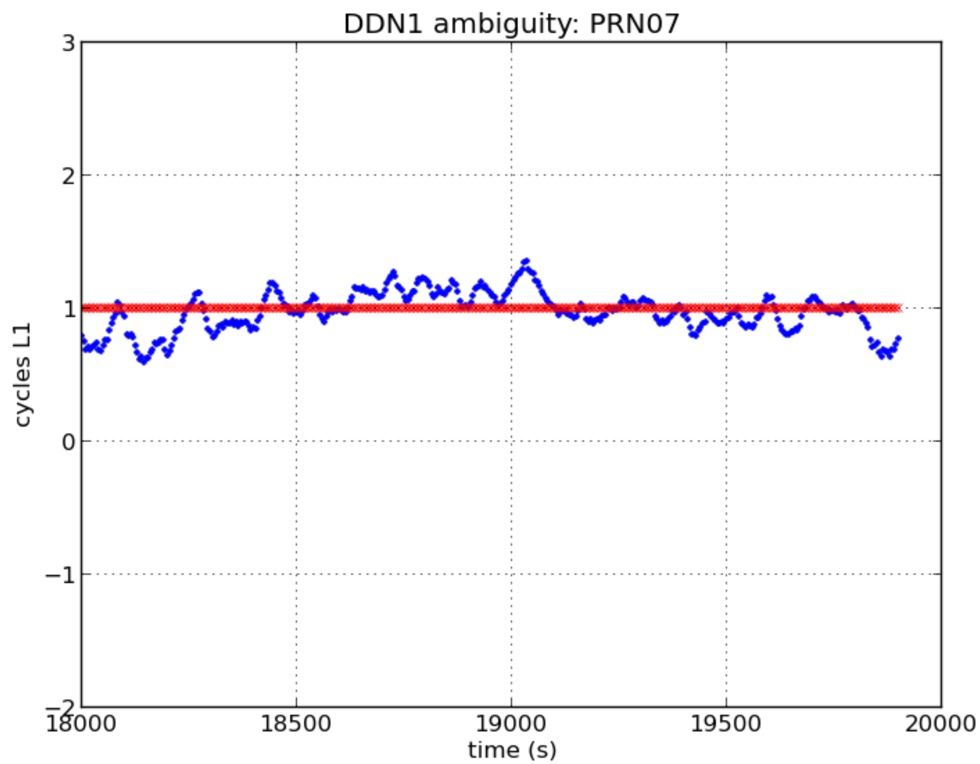


A2 Dual Frequency Ambiguity Fixing

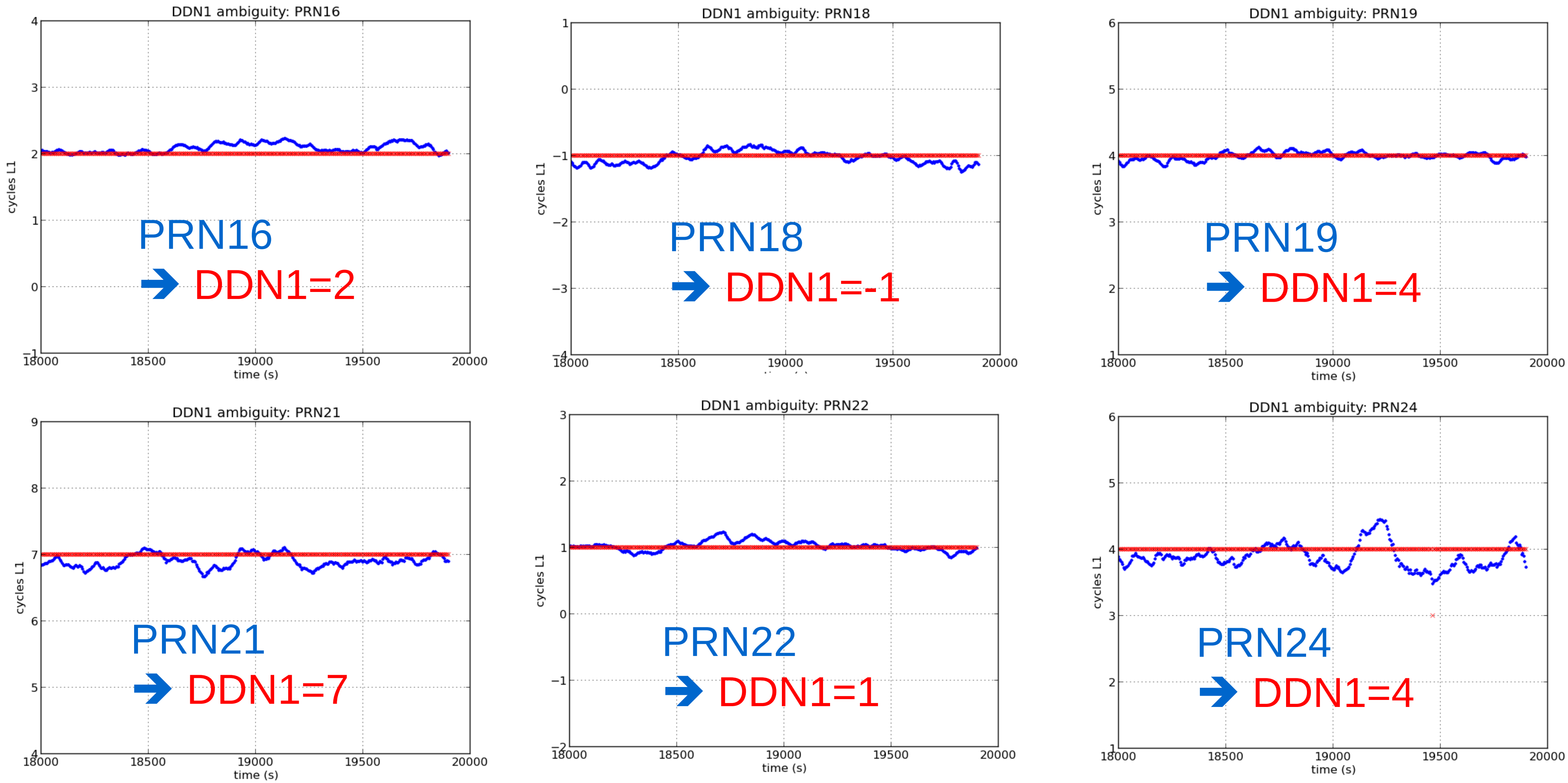
```
graph.py -f DDN1_PRN24 -x2 -y3 -c '($1==07)' -s.
         -f DDN1_PRN24 -x2 -y4 -c '($1==07)' -sx
         --cl r --yn -2 --yx 3 --xl "time (s)"
         --yl "cycles L1" -t "DDN1 ambiguity: PRN07"
```

PRN07 plot:

→ DDN1=1



A2 Dual Frequency Ambiguity Fixing



A2.3 Fixing DDN2

Using the previous DDNw and DDN1 ambiguities fixed, fix the DDN2 ambiguity:

```
PRN = [ 03   07   16   18   19   21   22   24]
DDNw= [  3    0    3   -3    2    8    0    4]
DDN1= [  2    1    2   -1    4    7    1    4]
```

Hint: The DDN2 can be easily computed by: **DDN2=DDN1-DDNw**

Then:

```
DDN2= [-1    1    -1    2    2   -1    1    0]
```

OVERVIEW

- **Introduction:** gLAB processing in command line.
- **Preliminary computations:** Data files.
- **Session A:** Fixing DD ambiguities one at a time: UPC1-UPC2.
- **Session B: Assessing the fixed ambiguities in navigation:** Differential positioning of UPC1-UPC2 receivers.
- **Session C:** Fixing DD ambiguities with LAMBDA method.

Note: UPC1-UPC2 receivers baseline: 37.95 metres.

Session B

Assessing the fixed ambiguities in navigation:
Differential positioning of UPC1-UPC2 receivers

(baseline: 37.95 metres)

B. Assessing the fixed ambiguities in navigation

- The DDN1 and DDN2 ambiguities have been fixed in the previous Session A using the “cascade method”.

- The obtained results (i.e. the DDN1 and DDN2 ambiguities) will be assessed in this Session B by computing the navigation solution using the carrier phases repaired with the fixed DDN1 and DDN2 ambiguities

- Finally, in next Session C, the ambiguities will be fixed using the LAMBDA method and the performance with the cascade method will be compared.

B. Assessing the fixed ambiguities in navigation

- After repairing the carrier ambiguities, these measurements will be used to navigate.
- Indeed, once the integer ambiguities are known, the carrier phase measurements become like "unambiguous pseudoranges", accurate at the centimetre level or better.
- Thence, the positioning is straightforward following the same procedure as with pseudoranges.
- Nevertheless, a wrong ambiguity fix can degrade the position solution significantly.

B. Repairing the DDL1 and DDL2 with the ambiguities fixed

B1. Repair the DDL1 and DDL2 carrier measurements with the DDN1 and DDN2 ambiguities FIXED and plot results to analyze the data.

Write in it the DDN1 and DDN2 ambiguities fixed in previous exercise in file **N1N2.dat** ➔

```
03  2 -1
07  1  1
16  2 -1
18 -1  2
19  4  2
21  7 -1
22  1  1
24  4  0
```

Using the previous file **N1N2.dat** and "DD_UPC1_UPC2_06_ALL.dat", generate a file with the following content:

```
-----------------------------  DD_UPC1_UPC2_06_ALL.fixL1L2 ------------------------------------
   1    2   3   4   5   6    7    8    9   10    11    12     13    14  15  16  17     18        19
[UPC1 UPC2 06 PRN DoY sec DDP1 DDL1 DDP2 DDL2 DDRho DDTrop DDIon El1 Az1 El2 Az2  λ1 DDN1  λ2 DDN2]
                                                               <---- UPC2 ---->
-------------------------------------------------------------------------------------------------
```

Note: This file is identical to file "DD_UPC1_UPC2_06_ALL.dat", but with the ambiguities added in the last fields #18 and #19.

B. Repairing the DDL1 and DDL2 with the ambiguities fixed

a) From previous file, generate a the file "sat.ambL1L2" with the following content:

1	2	3	4	5
[PRN	DDN1	DDN2	λ_1DDN1	λ_2DDN2]

```
gawk 'BEGIN{c=299792458;f0=10.23e+6;l1=c/(154*f0);l2=c/(120*f0)}
      {printf "%02i %3i %3i %14.4f %14.4f \n", $1,$2,$3,$2*l1,$3*l2}' N1N2.dat >
                                                                   sat.ambL1L2
```

b) Generate the "DD_UPC1_UPC2_06_30.fixL1L2" file:

```
cat DD_UPC1_UPC2_06_ALL.dat|
         gawk 'BEGIN{for (i=1;i<1000;i++) {getline <"sat.ambL1L2";A1[$1]=$4;A2[$1]=$5}}
  {printf "%s %02i %02i %s %14.4f %14.4f %14.4f %14.4f %14.4f %14.4f %14.4f %14.4f
                            %14.4f %14.4f %14.4f %14.4f %14.4f %14.4f %14.4f\n",
       $1,$2,$3,$4,$5,$6,$7,$8,$9,$10,$11,$12,$13,$14,$15,$16,$17,A1[$4], A2[$5]}' >
                                                        DD_UPC1_UPC2_06_ALL.fixL1L2
```

B. Repairing the DDL1 and DDL2 with the ambiguities fixed

```
------------------------------  DD_UPC1_UPC2_06_ALL.fixL1L2 ----------------------------------------
   1    2   3   4   5   6    7    8    9   10    11    12     13     14  15  16  17    18      19
[UPC1 UPC2 06 PRN DoY sec DDP1 DDL1 DDP2 DDL2 DDRho DDTrop DDIon El1 Az1 El2 Az2 λ1 DDN1  λ2 DDN2]
                                                                 <---- UPC2 ---->
-----------------------------------------------------------------------------------------------------
```

c) Make and discuss the following plots for DDL1

```
graph.py -f DD_UPC1_UPC2_06_ALL.fixL1 -x6 -y'($8-$18-$11)'
        -so --yn -0.06 --yx 0.06 -l "(DDL1-lambda1*DDN1)-DDrho" --xl "time (s)" --yl "m"
```

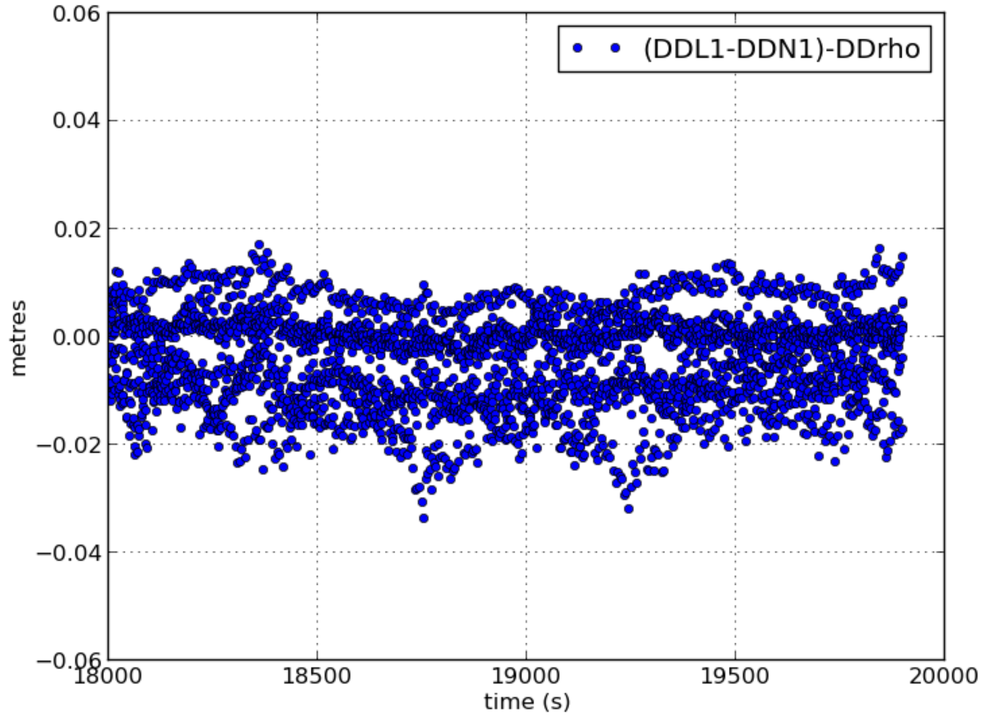
```
graph.py -f DD_UPC1_UPC2_06_ALL.fixL1 -x6 -y'($8-$11)'
        -so --yn -5 --yx 5 -l "(DDL1)-DDrho" --xl "time (s)" --yl "metres"
```

```
graph.py -f DD_UPC1_UPC2_06_ALL.fixL1 -x6 -y'($8-$18)'
        -so --yn -20 --yx 20 -l "(DDL1-lambda1*DDN1)" --xl "time (s)" --yl "metres"
```

B. Repairing the DDL1 and DDL2 with the ambiguities fixed

```
------------------------------ DD_UPC1_UPC2_06_ALL.fixL1L2 -------------------------------------
   1    2    3   4    5   6    7    8    9    10    11     12     13    14   15   16   17    18       19
[UPC1 UPC2 06 PRN DoY sec DDP1 DDL1 DDP2 DDL2 DDRho DDTrop DDIon El1 Az1 El2 Az2  λ1 DDN1  λ2 DDN2]
                                                                     <---- UPC2 ---->
-----------------------------------------------------------------------------------------------
```

```
graph.py -f DD_UPC1_UPC2_06_ALL.fixL1
         -x6 -y'($8-$18-$11)'
         -so --yn -0.06 --yx 0.06
         -l "(DDL1-λ1 DDN1)-DDrho"
         --xl "time (s)" --yl "m"
```



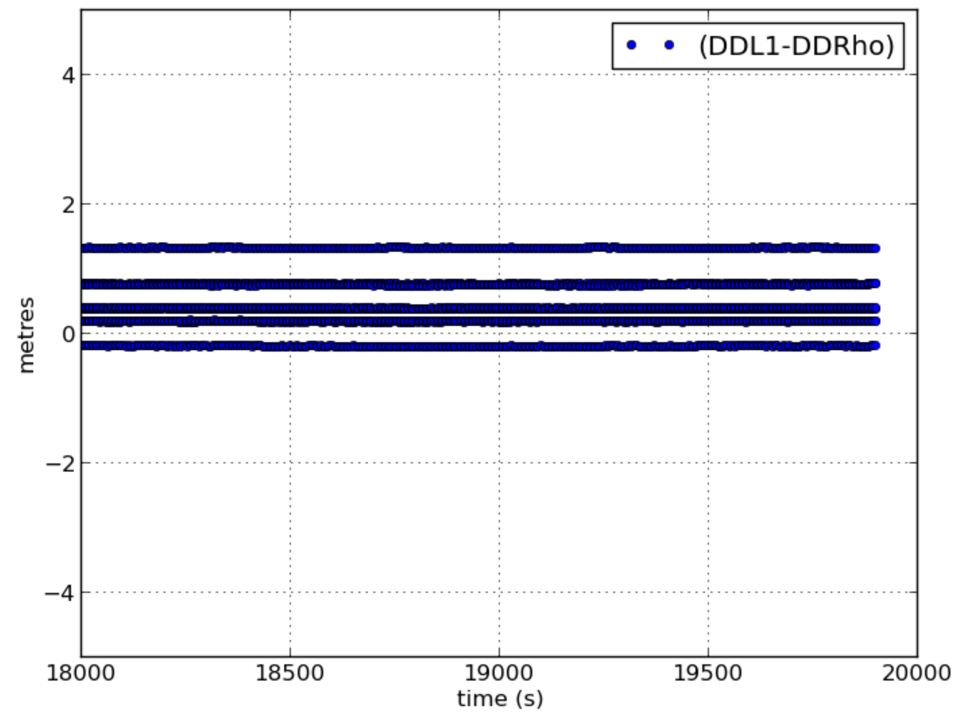
Questions:

Explain what is the meaning of this plot.

B. Repairing the DDL1 and DDL2 with the ambiguities fixed

```
----------------------------  DD_UPC1_UPC2_06_ALL.fixL1L2 -------------------------------------
   1    2   3   4   5   6    7    8    9    10    11    12     13     14  15  16  17    18       19
[UPC1 UPC2 06 PRN DoY sec DDP1 DDL1 DDP2 DDL2 DDRho DDTrop DDIon El1 Az1 El2 Az2  λ1 DDN1  λ2 DDN2]
                                                                <---- UPC2 ---->
-----------------------------------------------------------------------------------------------
```

```
graph.py -f DD_UPC1_UPC2_06_ALL.fixL1
         -x6 -y'($8-$11)'
         -so --yn -5 --yx 5
         -l "DDL1-DDrho"
         --xl "time (s)" --yl "m"
```

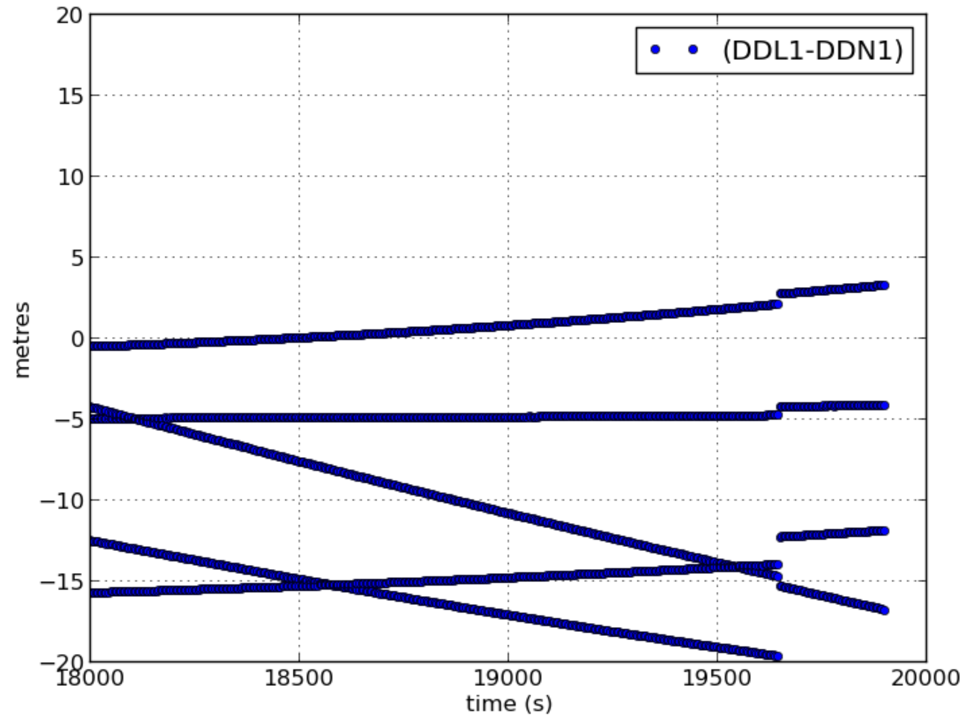


Questions:
Explain what is the meaning of this plot.

B. Repairing the DDL1 and DDL2 with the ambiguities fixed

```
-----------------------------  DD_UPC1_UPC2_06_ALL.fixL1L2 ----------------------------------------
   1     2    3   4    5   6     7     8     9   10     11    12     13     14  15   16   17     18      19
[UPC1 UPC2 06 PRN DoY sec DDP1 DDL1 DDP2 DDL2 DDRho DDTrop DDIon El1 Az1 El2 Az2  λ1 DDN1  λ2 DDN2]
                                                                     <---- UPC2 ---->
----------------------------------------------------------------------------------------------------
```

```
graph.py -f DD_UPC1_UPC2_06_ALL.fixL1
         -x6 -y'($8-$18)'
         -so --yn -20 --yx 20
         -l "(DDL1-λ1 DDN1)"
         --xl "time (s)" --yl "m"
```



Questions:

1.- Explain what is the meaning of this plot.

2.- Why a trend and a discontinuity appears?

B. Repairing the DDL1 and DDL2 with the ambiguities fixed

```
-----------------------------  DD_UPC1_UPC2_06_ALL.fixL1L2 -----------------------------------
   1    2   3   4   5   6    7    8    9   10    11    12     13    14  15  16  17    18      19
[UPC1 UPC2 06 PRN DoY sec DDP1 DDL1 DDP2 DDL2 DDRho DDTrop DDIon El1 Az1 El2 Az2  λ1 DDN1  λ2 DDN2]
                                                                  <---- UPC2 ---->
-----------------------------------------------------------------------------------------------
```

d) Make and discuss the following plots for DDL2

```
graph.py -f DD_UPC1_UPC2_06_ALL.fixL1 -x6 -y'($10-$19-$11)'
        -so --yn -0.06 --yx 0.06 -l "(DDL2-lambda2*DDN2)-DDrho" --xl "time (s)" --yl "m"
```

```
graph.py -f DD_UPC1_UPC2_06_ALL.fixL1 -x6 -y'($10-$11)'
        -so --yn -5 --yx 5 -l "(DDL2)-DDrho" --xl "time (s)" --yl "metres"
```

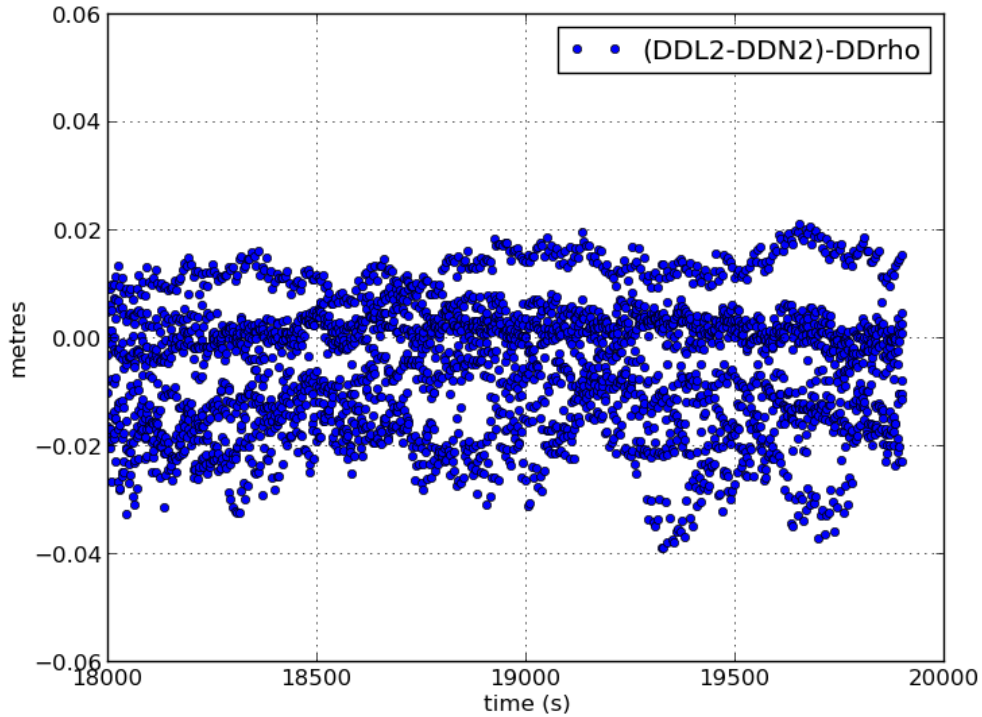
```
graph.py -f DD_UPC1_UPC2_06_ALL.fixL1 -x6 -y'($10-$19)'
        -so --yn -20 --yx 20 -l "(DDL2-lambda2*DDN2)" --xl "time (s)" --yl "metres"
```

B. Repairing the DDL1 and DDL2 with the ambiguities fixed

```
------------------------------  DD_UPC1_UPC2_06_ALL.fixL1L2 ----------------------------------------
   1     2   3   4   5   6    7    8    9   10    11    12    13    14  15  16  17    18      19
[UPC1 UPC2 06 PRN DoY sec DDP1 DDL1 DDP2 DDL2 DDRho DDTrop DDIon El1 Az1 El2 Az2  λ1 DDN1  λ2 DDN2]
                                                                  <---- UPC2 ---->
-----------------------------------------------------------------------------------------------------
```

```
graph.py -f DD_UPC1_UPC2_06_ALL.fixL1
         -x6 -y'($9-$19-$11)'
         -so --yn -0.06 --yx 0.06
         -l "(DDL2-λ2 DDN2)-DDrho"
         --xl "time (s)" --yl "m"
```

Questions:
Explain what is the meaning of this plot.

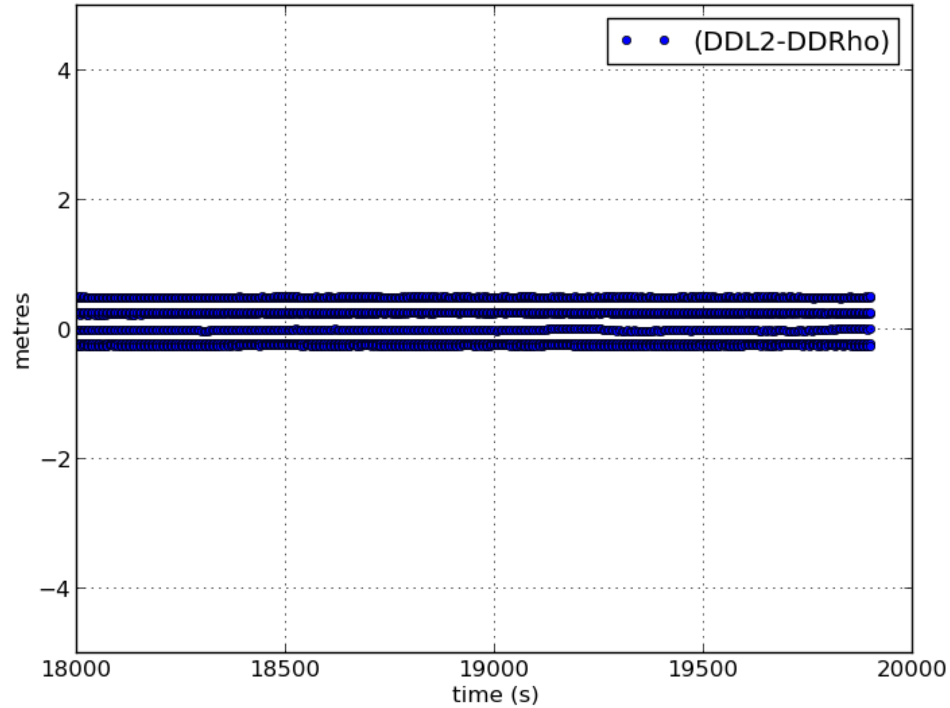


B. Repairing the DDL1 and DDL2 with the ambiguities fixed

```
------------------------------  DD_UPC1_UPC2_06_ALL.fixL1L2 ----------------------------------------
   1    2   3   4   5   6    7    8    9   10    11    12    13    14  15  16  17     18       19
[UPC1 UPC2 06 PRN DoY sec DDP1 DDL1 DDP2 DDL2 DDRho DDTrop DDIon El1 Az1 El2 Az2  λ1 DDN1  λ2 DDN2]
                                                                   <---- UPC2 ---->
-----------------------------------------------------------------------------------------------------
```

```
graph.py -f DD_UPC1_UPC2_06_ALL.fixL1
         -x6 -y'($9-$11)'
         -so --yn -5 --yx 5
         -l "DDL2-DDrho"
         --xl "time (s)" --yl "m"
```

Questions:
Explain what is the meaning of this plot.



B. Repairing the DDL1 and DDL2 with the ambiguities fixed

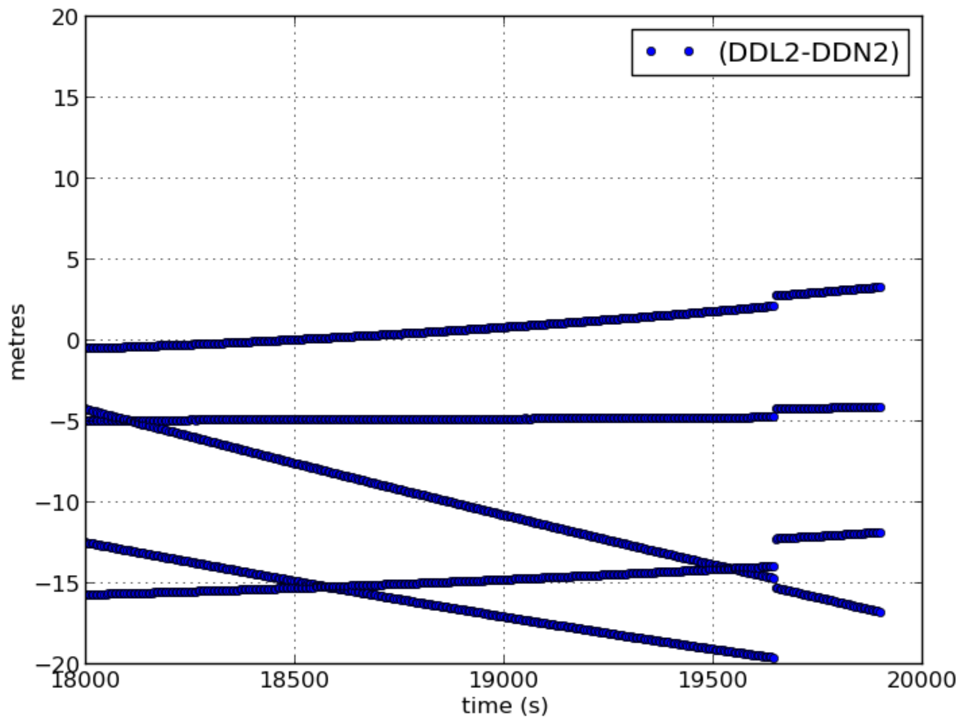
```
------------------------------ DD_UPC1_UPC2_06_ALL.fixL1L2 ----------------------------------------
   1     2    3    4    5    6     7     8     9    10     11    12     13     14   15   16   17     18       19
[UPC1 UPC2 06 PRN DoY sec DDP1 DDL1 DDP2 DDL2 DDRho DDTrop DDIon El1 Az1 El2 Az2  λ1 DDN1  λ2 DDN2]
                                                                   <---- UPC2 ---->
---------------------------------------------------------------------------------------------------
```

```
graph.py -f DD_UPC1_UPC2_06_ALL.fixL1
         -x6 -y'($9-$198)'
         -so --yn -20 --yx 20
         -l "(DDL2-λ2 DDN2)"
         --xl "time (s)" --yl "m"
```

Questions:

1.- Explain what is the meaning of this plot.

2.- Why a trend and a discontinuity appears?



B2 Assessing the fixed ambiguities in navigation

- After repairing the carrier ambiguities, these measurements will be used to navigate.

- Indeed, once the integer ambiguities are known, the carrier phase measurements become like “unambiguous pseudoranges”, accurate at the centimetre level or better.

- Thence, the positioning is straightforward following the same procedure as with pseudoranges.

- Nevertheless, a wrong ambiguity fix can degrade the position solution significantly.

B2.1 UPC1-UPC2 Baseline vector estimation with DDL1 carrier (using the time-tagged reference station measurements)

- In this exercise we will considerer an implementation of differential positioning where the user estimates the baseline vector using the time-tagged measurements of the reference station.

- This approach is **usually referred to as relative positioning** and can be applied in some applications where the coordinates of the reference station are not accurately known and where the relative position vector between the reference station and the user is the main interest. Examples are formation flying, automatic landing on ships…

- Of course, the knowledge of the reference receiver location would allow the user to compute its absolute coordinates.

- This is a simple approach, where synchronism delays between the time tag measurements of the reference station and the user must be taken into account for real-time positioning.

B2.1 UPC1-UPC2 Baseline vector estimation with L1 carrier (using the time-tagged reference station measurements)

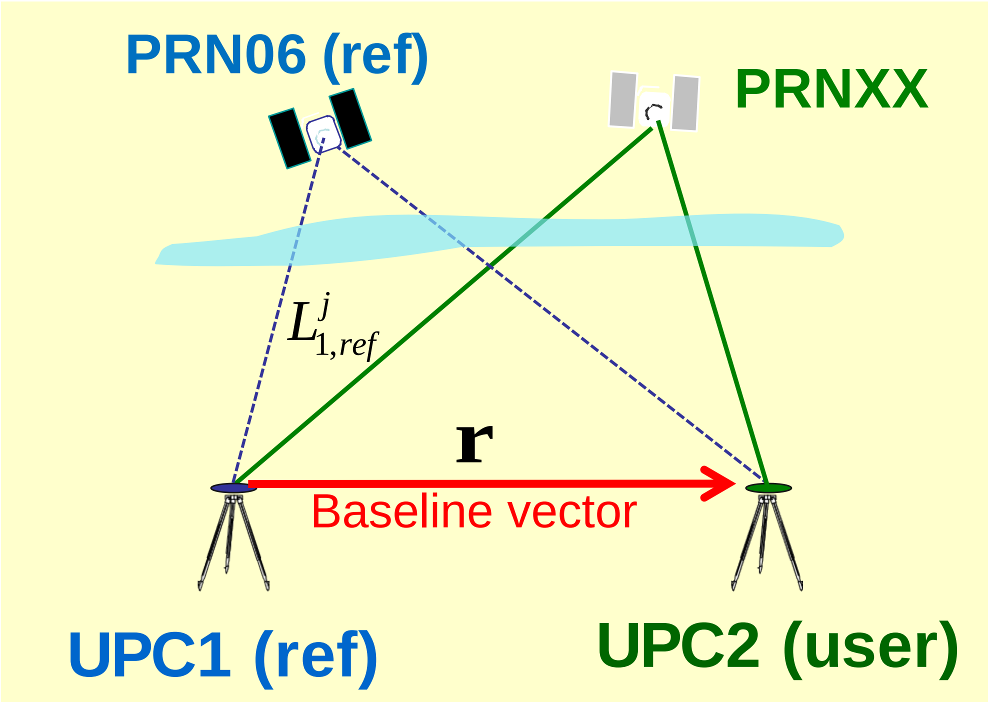
Data file

```
------------------------------  DD_UPC1_UPC2_06_ALL.fixL1L2 ----------------------------------------
   1     2    3    4    5    6     7    8     9    10     11     12     13     14   15   16   17       18       19
[UPC1 UPC2 06 PRN DoY sec DDP1 DDL1 DDP2 DDL2 DDRho DDTrop DDIon El1 Az1 El2 Az2  λ1 DDN1  λ2 DDN2]
                                                                           <---- UPC2 ---->
------------------------------------------------------------------------------------------------------
```

Where the elevation (EL) and azimuth (AZ) are taken from station **UPC2** (the user)

and where, (EL1, AZ1) are for satellite PNR06 (reference) and (EL1, AZ1) are for satellite PRNXX

$L_{1,ref}^{j}$ Measurements broadcast by the reference station.



B2.1 UPC1-UPC2 Baseline vector estimation with P1 code (using the time-tagged reference station measurements)

Estimate the baseline vector between UPC1 and UPC2 receivers using the code measurements of file (DD_UPC1_UPC2_06_ALL.dat).

Note: Use the entire file (i.e. time interval [18000:19900]).

$$[\mathrm{DDL1}-\lambda_1\,\mathrm{DDN1}]=[\mathrm{Los\_k}-\mathrm{Los\_06}]*[\mathrm{baseline}]$$

Notation

$$\begin{bmatrix} DDL_1^{6,03}-\lambda_1 DDN_1^{6,03} \\ DDL_1^{6,07}-\lambda_1 DDN_1^{6,07} \\ \vdots \\ DDL_1^{6,24}-\lambda_1 DDN_1^{6,24} \end{bmatrix} = \begin{bmatrix} -\left(\hat{\boldsymbol{\rho}}^3-\hat{\boldsymbol{\rho}}^6\right)^T \\ -\left(\hat{\boldsymbol{\rho}}^7-\hat{\boldsymbol{\rho}}^6\right)^T \\ \vdots \\ -\left(\hat{\boldsymbol{\rho}}^{24}-\hat{\boldsymbol{\rho}}^6\right)^T \end{bmatrix}\mathbf{r}$$

$$DDL_1^{k\,j} \equiv \mathrm{DDL1(involving\ satellites}\ j\ \mathrm{and}\ k)$$

$$DDL_1^{k\,j} = DL_{1,usr}^{k\,j} - \boxed{DL_{1,ref}^{k\,j}}$$

$$= \left(L_{1,usr}^{j} - L_{1,usr}^{k}\right) - \boxed{\left(L_{1,ref}^{j} - L_{1,ref}^{k}\right)}$$

$L_{1,ref}^{j}$ Measurements broadcast by the reference station.

Estimate the baseline vector between UPC1 and UPC2 receivers using the code measurements of file (DD_UPC1_UPC2_06_ALL.dat).

Note: Use the entire file (i.e. time interval [18000:19900]).

```
[DDL1- λ1 DDN1 ]=[Los_k - Los_06]*[baseline]
```

Notation

$$\begin{bmatrix} DDL_1^{6,03} - \lambda_1 DDN_1^{6,03} \\ DDL_1^{6,07} - \lambda_1 DDN_1^{6,07} \\ \vdots \\ DDL_1^{6,24} - \lambda_1 DDN_1^{6,24} \end{bmatrix} = \begin{bmatrix} -\left(\hat{\boldsymbol{\rho}}^3 - \hat{\boldsymbol{\rho}}^6\right)^T \\ -\left(\hat{\boldsymbol{\rho}}^7 - \hat{\boldsymbol{\rho}}^6\right)^T \\ \vdots \\ -\left(\hat{\boldsymbol{\rho}}^{24} - \hat{\boldsymbol{\rho}}^6\right)^T \end{bmatrix} \mathbf{r}$$

$\mathbf{r} \equiv$ Baseline vector

$DDL_1^{k\,j} \equiv$ DDL1(involving satellites j and k)

$\hat{\boldsymbol{\rho}}^k \equiv$ Line-Of-Sight unit vector to satelite k

$$\hat{\boldsymbol{\rho}}^k \equiv \left[\cos(El_k)\sin(Az_k), \quad \cos(El_k)\cos(Az_k), \quad \sin(El_k)\right]$$

B2.1 UPC1-UPC2 Baseline vector estimation with L1 carrier (using the time-tagged reference station measurements)

Using the DDL1 carrier with the ambiguities FIXED, compute the LS single epoch solution for the whole interval 180000< t <199000 with the program LS.f

Note: The program `LS.f` computes the Least Square solution for each measurement epoch of the input file (see the FORTRAN code `LS.f`)

The following procedure can be applied:

a) generate a file with the following content;

$$[\texttt{Time}],\ [\texttt{DDL1-}\lambda_1\texttt{DDN1}],\ [\ \texttt{Los\_k - Los\_06}]$$

where:

Time= seconds of day

DDL1- λ_1 **DDN1**= Prefit residulas (i.e., "y" values in program LS.f)

Los_k-Los_06 = The three components of the geometry matrix (i.e., matrix "a" in program LS.f)

B2.1 UPC1-UPC2 Baseline vector estimation with P1 code (using the time-tagged reference station measurements)

Justify that the next sentence builds the navigation equations system

See file content **in slide #31**

[DDL1- λ_1 DDN1]=[Los_k - Los_06]*[baseline]

```
cat DD_UPC1_UPC2_06_ALL.fixL1L2 | gawk 'BEGIN{g2r=atan2(1,1)/45}
                    {e1=$14*g2r;a1=$15*g2r;e2=$16*g2r;a2=$17*g2r;
 printf "%s %14.4f %8.4f %8.4f %8.4f \n",
  $6, $8-$18, -cos(e2)*sin(a2)+cos(e1)*sin(a1),
              -cos(e2)*cos(a2)+cos(e1)*cos(a1), -sin(e2)+sin(e1)}' > M.dat
```

Time

$$\begin{bmatrix} DDL_1^{6,03} - \lambda_1 DDN_1^{6,03} \\ DDL_1^{6,07} - \lambda_1 DDN_1^{6,07} \\ \vdots \\ DDL_1^{6,24} - \lambda_1 DDN_1^{6,24} \end{bmatrix} = \begin{bmatrix} -\left(\hat{\boldsymbol{\rho}}^3 - \hat{\boldsymbol{\rho}}^6\right)^T \\ -\left(\hat{\boldsymbol{\rho}}^7 - \hat{\boldsymbol{\rho}}^6\right)^T \\ \vdots \\ -\left(\hat{\boldsymbol{\rho}}^{24} - \hat{\boldsymbol{\rho}}^6\right)^T \end{bmatrix} \mathbf{r}$$

```
[DDL1- λ1 DDN1]    [  Los_k - Los_06]
 ------------      -------------------
  -3.3762           0.3398 -0.1028 0.0714
  -7.1131           0.1725  0.5972 0.0691
   4.3881          -0.6374  0.0227 0.2725
```

$$\hat{\boldsymbol{\rho}}^k \equiv \left[\cos(El_k)\sin(Az_k), \quad \cos(El_k)\cos(Az_k), \quad \sin(El_k)\right]$$

B2.1 UPC1-UPC2 Baseline vector estimation with L1 carrier (using the time-tagged reference station measurements)

```
----------------------------- DD_UPC1_UPC2_06_ALL.fixL1L2 -------------------------------------
   1    2   3   4   5   6    7    8    9   10    11    12     13    14  15  16  17     18       19
[UPC1 UPC2 06 PRN DoY sec DDP1 DDL1 DDP2 DDL2 DDRho DDTrop DDIon El1 Az1 El2 Az2  λ1 DDN1  λ2 DDN2]
                                                                <---- UPC2 ---->
-----------------------------------------------------------------------------------------------
```

$$[\mathrm{Time}],\ [\mathrm{DDL1} - \lambda_1\,\mathrm{DDN1}],\ [\mathrm{Los\_k} - \mathrm{Los\_06}]$$

The following sentence can be used:

```
cat DD_UPC1_UPC2_06_ALL.fixL1L2 | gawk 'BEGIN{g2r=atan2(1,1)/45}
  {e1=$14*g2r;a1=$15*g2r;e2=$16*g2r;a2=$17*g2r;printf "%s %14.4f
   %8.4f %8.4f %8.4f \n",$6,$8-$18,-cos(e2)*sin(a2)+cos(e1)*sin(a1),
        -cos(e2)*cos(a2)+cos(e1)*cos(a1),-sin(e2)+sin(e1)}' > L1model.dat
```

b) Compute the Least Squares solution:

```
cat L1model.dat |LS > L1fix.pos
```

Plot the baseline estimation error:

```
graph.py -f L1fix.pos -x1 -y'($2+27.4170)' -s.- -l "North error"
         -f L1fix.pos -x1 -y'($3+26.2341)' -s.- -l "East error"
         -f L1fix.pos -x1 -y'($4+ 0.0304)'  -s.- -l "UP error"
    --yn -.1 --yx .1 --xl "time (s)" --yl "error (m)" -t "Baseline error”
```

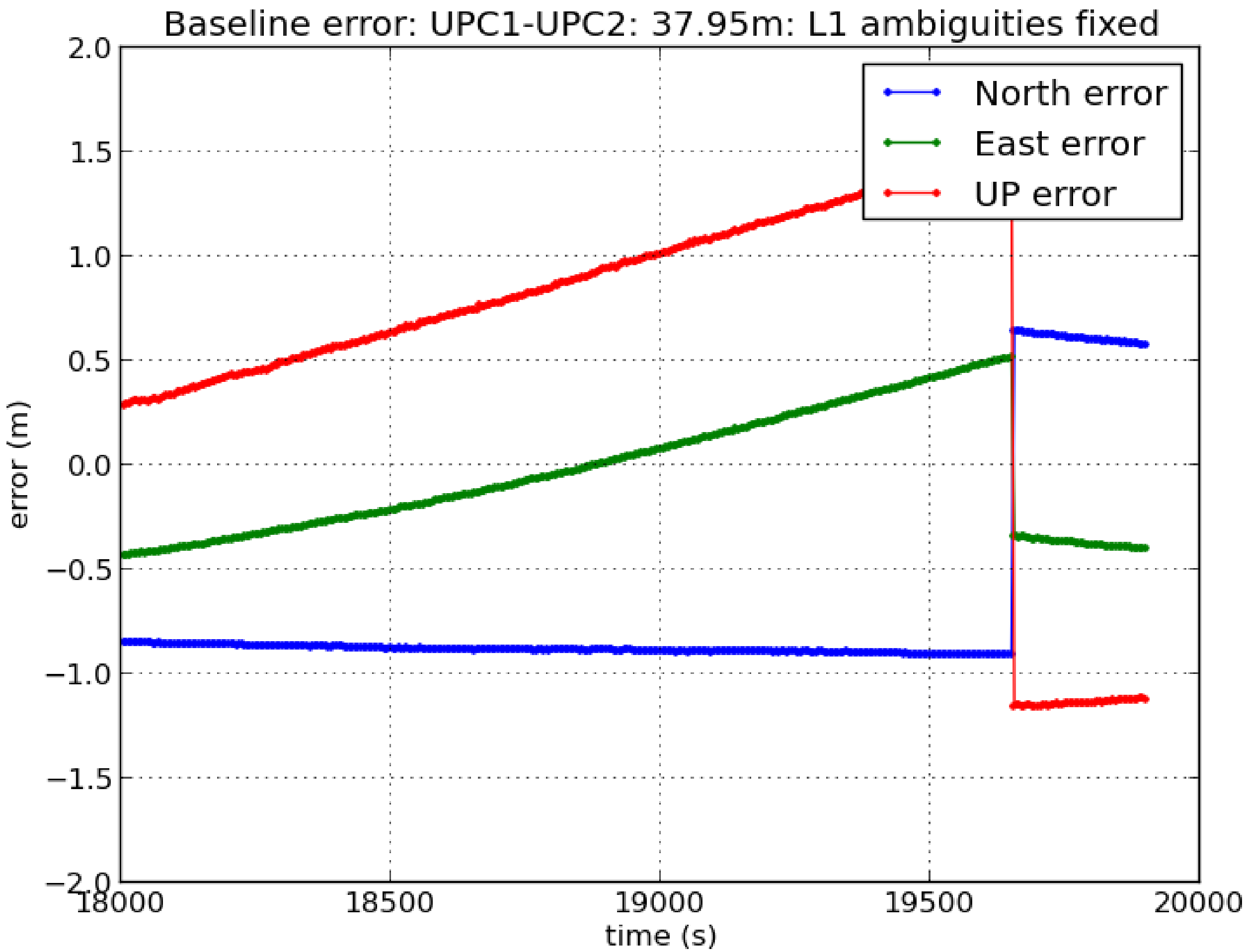
Note: An accurate estimate of baseline is:

bsl_enu =[-27.4170 -26.2341 -0.0304]

Use this determination to assess the baseline vector estimation error.

B2.1 UPC1-UPC2 Baseline vector estimation with L1 carrier (using the time-tagged reference station measurements)

L1 Baseline estimation error after fixing ambiguities



Questions:

1.- What is the expected accuracy when positioning with carrier after fixing ambiguities?

2.- Discuss why a trend and a discontinuity appears?

B2.2. UPC1-UPC2 differential positioning with L1 carrier (using the computed differential corrections)

- In the previous exercise we have considered an implementation of differential positioning where the user estimates the baseline vector from the time-tagged measurements of the reference station.

- In the next exercises, we will consider the common implementation of Differential positioning, where the reference receiver coordinates are accurately known and used to compute range corrections for each tracked satellite in view. Then, the user applies these corrections to improve the positioning.

- Unlike in the previous implementation, the synchronism errors between the time-tagged measurements will be not critical in this approach, as the differential corrections vary slowly.

Using code DDL1 measurements, estimate the coordinates of receiver UPC2 taking UPC1 as a reference receiver.

Justify that the associated equations system is given by:

$$[\mathrm{DDL1-DDRho-}\lambda_1\,\mathrm{DDN1}]=[\mathrm{Los\_k - Los\_06}]*[\mathrm{dr}]$$

Notation

$$\begin{bmatrix} DDL_1^{6,03}-DD\rho^{6,03}-\lambda_1 DDN_1 \\ DDL_1^{6,07}-DD\rho^{6,07}-\lambda_1 DDN_1 \\ \vdots \\ DDL_1^{6,24}-DD\rho^{6,30}-\lambda_1 DDN_1 \end{bmatrix} = \begin{bmatrix} -\left(\hat{\boldsymbol{\rho}}^3-\hat{\boldsymbol{\rho}}^6\right)^T \\ -\left(\hat{\boldsymbol{\rho}}^7-\hat{\boldsymbol{\rho}}^6\right)^T \\ \vdots \\ -\left(\hat{\boldsymbol{\rho}}^{24}-\hat{\boldsymbol{\rho}}^6\right)^T \end{bmatrix} \mathbf{dr}$$

$$\mathbf{dr}=\mathbf{r}_{IND3}-\mathbf{r}_{0,IND3}$$

$$DDL_1^{k\,j} \equiv \text{DDL1(involving satellites } j \text{ and } k)$$

$$\hat{\boldsymbol{\rho}}^k \equiv \text{Line-Of-Sight unit vector to satelite } k$$

$$\hat{\boldsymbol{\rho}}^k \equiv \left[\cos(El_k)\sin(Az_k) \quad \cos(El_k)\cos(Az_k) \quad \sin(El_k)\right]$$

Using code DDL1 measurements, estimate the coordinates of receiver UPC2 taking UPC1 as a reference receiver.

Justify that the associated equations system is given by:

[DDL1- DDRho-λ_1 DDN1]=[Los_k - Los_06]*[dr]

Notation

$$\begin{bmatrix} DDL_1^{6,03} - DD\rho^{6,03} - \lambda_1 DDN_1 \\ DDL_1^{6,07} - DD\rho^{6,07} - \lambda_1 DDN_1 \\ \vdots \\ DDL_1^{6,24} - DD\rho^{6,30} - \lambda_1 DDN_1 \end{bmatrix} = \begin{bmatrix} -\left(\hat{\boldsymbol{\rho}}^3 - \hat{\boldsymbol{\rho}}^6\right)^T \\ -\left(\hat{\boldsymbol{\rho}}^7 - \hat{\boldsymbol{\rho}}^6\right)^T \\ \vdots \\ -\left(\hat{\boldsymbol{\rho}}^{24} - \hat{\boldsymbol{\rho}}^6\right)^T \end{bmatrix} \mathbf{dr}$$

$$DDL_1^{k\,j} \equiv \text{DDL1(involving satellites } j \text{ and } k)$$

$$DDL_1^{k\,j} - DD\rho^{k\,j} = D\left(L_{1,usr}^{k\,j} - \rho_{usr}^{k\,j}\right) - \boxed{D\left(L_{1,ref}^{k\,j} - \rho_{ref}^{k\,j}\right)}$$
$$= \left[\left(L_{1,usr}^{j} - \rho_{usr}^{j}\right) - \left(L_{1,usr}^{k} - \rho_{usr}^{k}\right)\right] - \boxed{\left[\left(L_{1,ref}^{j} - \rho_{ref}^{j}\right) - \left(L_{1,ref}^{k} - \rho_{ref}^{k}\right)\right]}$$

$$PRC_{L1,1}{}^{j} \equiv L_{1,ref}^{j} - \rho_{ref}^{j}$$

Computed corrections broadcast by the reference station.

B2.2. UPC1-UPC2 differential positioning with L1 carrier (using the computed differential corrections)

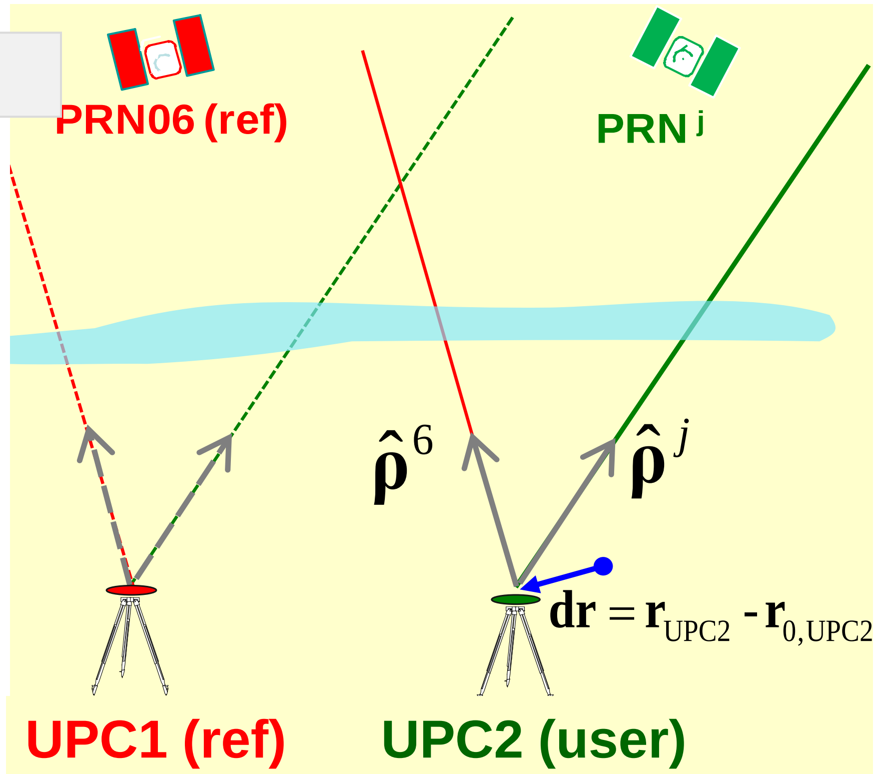
Using code DDL1 measurements, estimate the coordinates of receiver UPC2 taking UPC1 as a reference receiver.

Justify that the associated equations system is given by:

[DDL1- DDRho-λ_1 DDN1]=[Los_k-Los_06]*[dr]

$$\begin{bmatrix} DDL_1^{6,03} - DD\rho^{6,03} - \lambda_1 DDN_1 \\ DDL_1^{6,07} - DD\rho^{6,07} - \lambda_1 DDN_1 \\ \vdots \\ DDL_1^{6,24} - DD\rho^{6,30} - \lambda_1 DDN_1 \end{bmatrix} = \begin{bmatrix} -\left(\hat{\boldsymbol{\rho}}^3 - \hat{\boldsymbol{\rho}}^6\right)^T \\ -\left(\hat{\boldsymbol{\rho}}^7 - \hat{\boldsymbol{\rho}}^6\right)^T \\ \vdots \\ -\left(\hat{\boldsymbol{\rho}}^{24} - \hat{\boldsymbol{\rho}}^6\right)^T \end{bmatrix} \mathbf{dr}$$

$$\hat{\boldsymbol{\rho}}^j \equiv \left[\cos(El_j)\sin(Az_j), \cos(El_j)\cos(Az_j), \sin(El_j)\right]$$



B2.2. UPC1-UPC2 differential positioning with L1 carrier (using the computed differential corrections)

Justify that the next sentence builds the navigation equations system

See file content **in slide #43**

[DDL1- DDRho- λ_1 DDN1]=[Los_k - Los_06]*[dr]

```
cat DD_UPC1_UPC2_06_ALL.fixL1L2 | gawk 'BEGIN{g2r=atan2(1,1)/45}
                           {e1=$14*g2r;a1=$15*g2r;e2=$16*g2r;a2=$17*g2r;
 printf "%s %14.4f %8.4f %8.4f %8.4f \n",
 $6, $8-$11-$18, -cos(e2)*sin(a2)+cos(e1)*sin(a1),
                 -cos(e2)*cos(a2)+cos(e1)*cos(a1),-sin(e2)+sin(e1)}'> M.dat
```

Time

$$\begin{bmatrix} DDL_1^{6,03} - DD\rho^{6,03} - \lambda_1 DDN_1^{6,03} \\ DDL_1^{6,07} - DD\rho^{6,07} - \lambda_1 DDN_1^{6,07} \\ \vdots \\ DDL_1^{6,24} - DD\rho^{6,24} - \lambda_1 DDN_1^{6,24} \end{bmatrix} = \begin{bmatrix} -\left(\hat{\boldsymbol{\rho}}^3 - \hat{\boldsymbol{\rho}}^6\right)^T \\ -\left(\hat{\boldsymbol{\rho}}^7 - \hat{\boldsymbol{\rho}}^6\right)^T \\ \vdots \\ -\left(\hat{\boldsymbol{\rho}}^{24} - \hat{\boldsymbol{\rho}}^6\right)^T \end{bmatrix} \mathbf{r}$$

[DDL1-DDRho- λ_1 DDN1]	[Los_k - Los_06]		
-3.3762	0.3398	-0.1028	0.0714
-7.1131	0.1725	0.5972	0.0691
4.3881	-0.6374	0.0227	0.2725

$$\hat{\boldsymbol{\rho}}^k \equiv \left[\cos(El_k)\sin(Az_k), \quad \cos(El_k)\cos(Az_k), \quad \sin(El_k)\right]$$

B2.2. UPC1-UPC2 differential positioning with L1 carrier (using the computed differential corrections)

```
----------------------------  DD_UPC1_UPC2_06_ALL.fixL1L2 ------------------------------------
   1     2    3    4    5    6     7     8     9    10     11     12     13     14   15   16   17     18        19
[UPC1 UPC2 06 PRN DoY sec DDP1 DDL1 DDP2 DDL2 DDRho DDTrop DDIon El1 Az1 El2 Az2  λ1 DDN1  λ2 DDN2]
                                                                    <---- UPC2 ---->
-----------------------------------------------------------------------------------------------
```

[Time], [DDL1- DDRho $-\lambda_1$ DDN1], [Los_k - Los_06]

The following sentence can be used

```
cat DD_UPC1_UPC2_06_ALL.fixL1L2 | gawk 'BEGIN{g2r=atan2(1,1)/45}
  {e1=$14*g2r;a1=$15*g2r;e2=$16*g2r;a2=$17*g2r;printf "%s %14.4f
   %8.4f %8.4f %8.4f \n",$6,$8-$11-$18,-cos(e2)*sin(a2)+cos(e1)*sin(a1),
        -cos(e2)*cos(a2)+cos(e1)*cos(a1),-sin(e2)+sin(e1)}' > L1model.dat
```

b) Compute the Least Squares solution

```
cat L1model.dat |LS > L1fix.pos
```

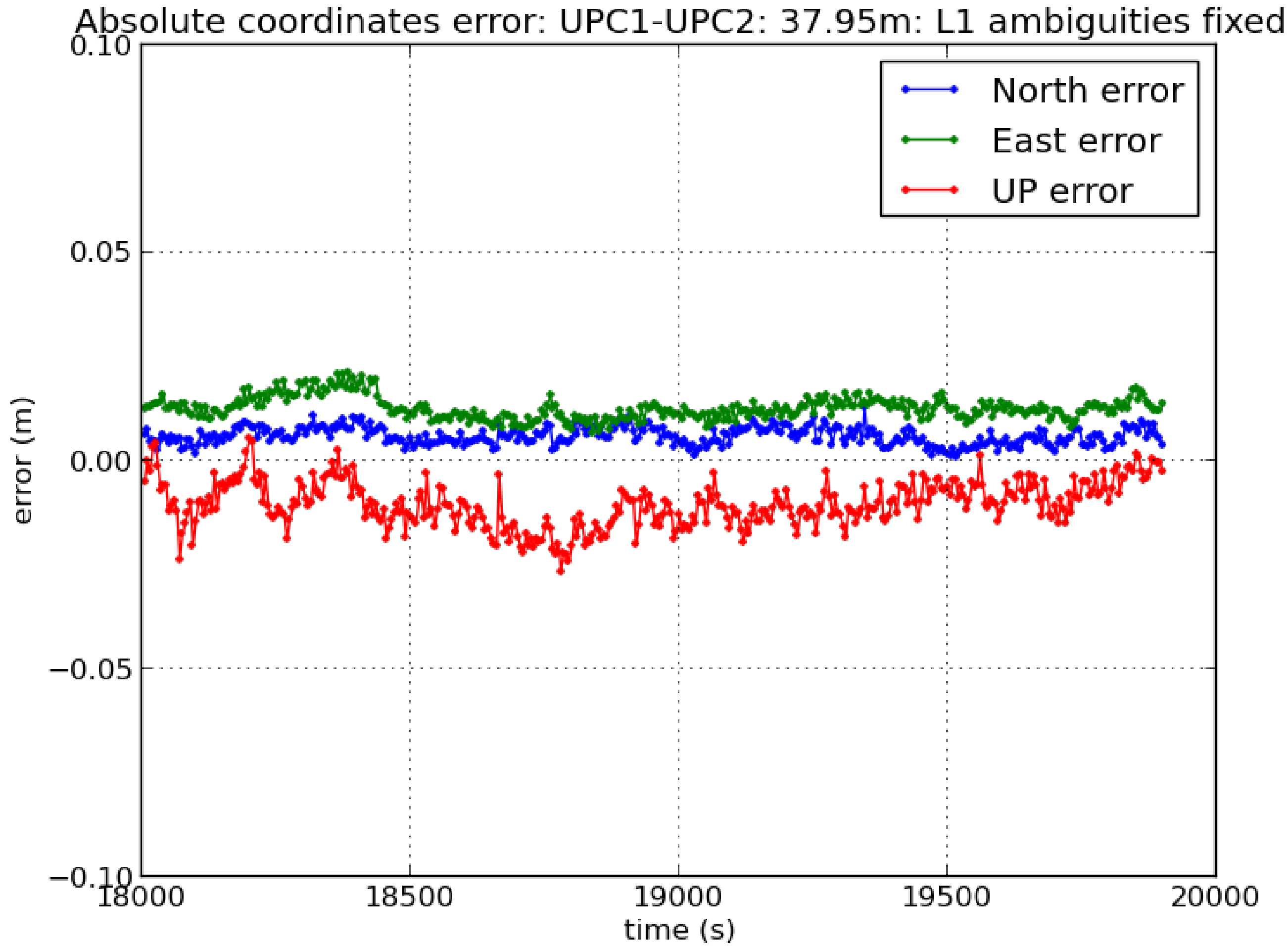
B2.2. UPC1-UPC2 differential positioning with L1 carrier (using the computed differential corrections)

Plot the absolute positioning error:

```
graph.py -f L1fix.pos -x1 –y2- s.- -l "North error"
         -f L1fix.pos -x1 –y3 -s.- -l "East error"
         -f L1fix.pos -x1 –y4 -s.- -l "UP error"
         --yn -.1 --yx .1 --xl "time (s)" --yl "error (m)"
         -t "Absolute positioning error with DDL1"
```

B2.2. UPC1-UPC2 differential positioning with L1 carrier (using the computed differential corrections)

L1 Differential positioning after fixing ambiguities



Questions:

Discuss why the results have improved, achieving centimetre level navigation.

B2.3. UPC1-UPC2 differential positioning with L2 carrier (using the computed differential corrections)

Repeat the previous positioning, but with the DDL2 carrier

```
-----------------------------  DD_UPC1_UPC2_06_ALL.fixL1L2 ---------------------------------------
   1     2    3    4    5    6     7     8     9    10     11    12     13     14   15   16   17     18       19
[UPC1 UPC2 06 PRN DoY sec DDP1 DDL1 DDP2 DDL2 DDRho DDTrop DDIon El1 Az1 El2 Az2  λ1 DDN1   λ2 DDN2]
                                                                    <---- UPC2 ---->
--------------------------------------------------------------------------------------------------
```

[Time], [DDL2- DDRho $-\lambda_2$DDN2], [Los_k - Los_06]

The following sentence can be used

```
cat DD_UPC1_UPC2_06_ALL.fixL1L2 | gawk 'BEGIN{g2r=atan2(1,1)/45}
  {e1=$14*g2r;a1=$15*g2r;e2=$16*g2r;a2=$17*g2r;printf "%s %14.4f
   %8.4f %8.4f %8.4f \n",$6,$9-$11-$19,-cos(e2)*sin(a2)+cos(e1)*sin(a1),
        -cos(e2)*cos(a2)+cos(e1)*cos(a1),-sin(e2)+sin(e1)}' > L2model.dat
```

Compute the Least Squares solution

```
cat L2model.dat |LS > L2fix.pos
```

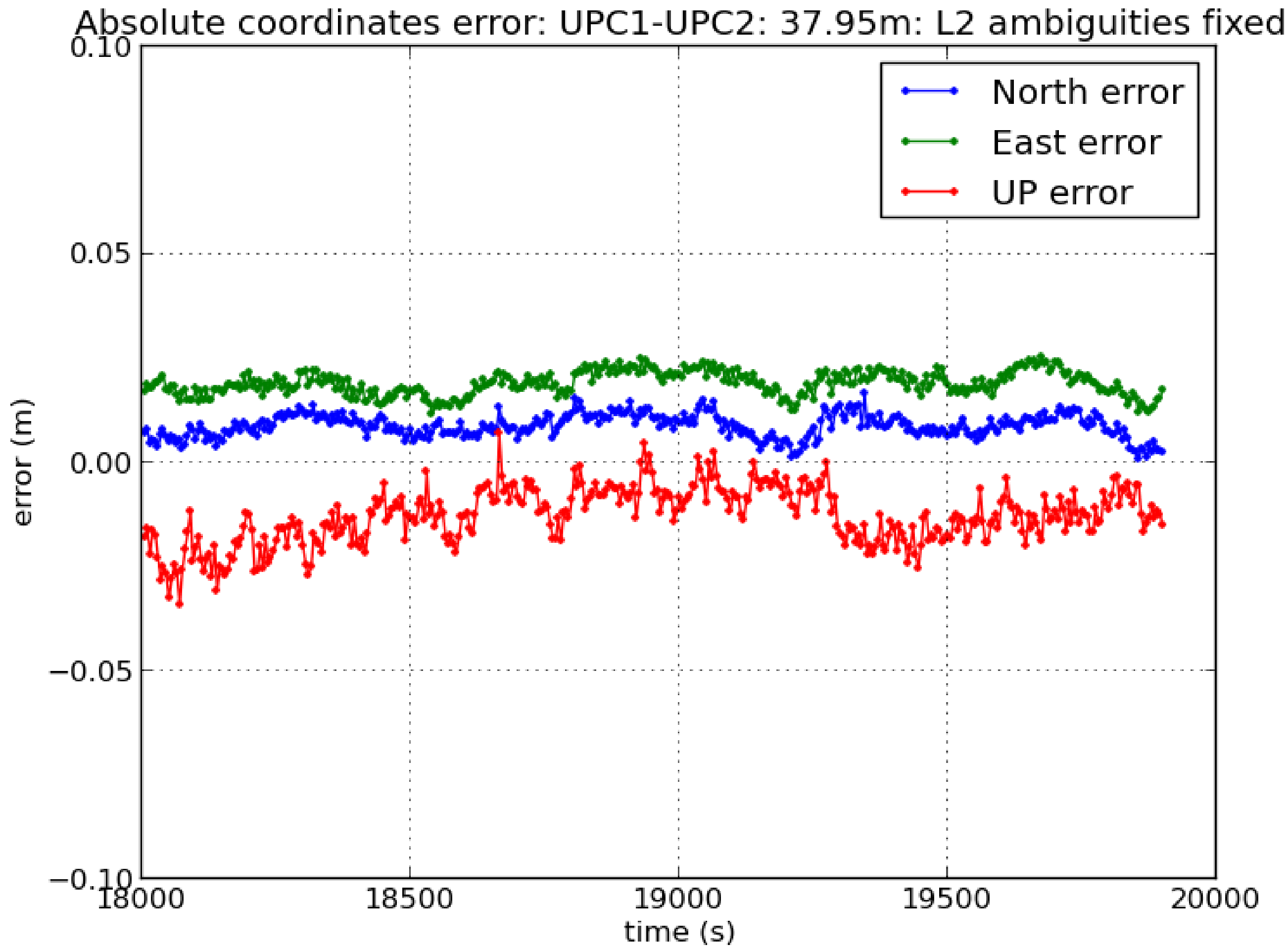
B2.2. UPC1-UPC2 differential positioning with L2 carrier (using the computed differential corrections)

Plot the absolute positioning error:

```
graph.py -f L2fix.pos -x1 –y2- s.- -l "North error"
         -f L2fix.pos -x1 –y3 -s.- -l "East error"
         -f L2fix.pos -x1 –y4 -s.- -l "UP error"
         --yn -.1 --yx .1 --xl "time (s)" --yl "error (m)"
         -t "Absolute positioning error with DDL2"
```

B2.2. UPC1-UPC2 differential positioning with L2 carrier (using the computed differential corrections)

L2 Differential positioning after fixing ambiguities



Questions:

Compare the results with the previous ones computed from DDL1.

B2.4. UPC1-UPC2 differential positioning with L1 carrier (using the computed differential corrections)

Analyze the effect of a wrong ambiguity fix over a single satellite (e.g. PRN07)

Simulate an error of 1 cycle in DDN1 for satellite PRN07 and compute the navigation solution:

1 cycle is added to the DDN1 of satellite PRN**07**

The following sentence can be used

```
cat DD_UPC1_UPC2_06_ALL.fixL1L2 | gawk 'BEGIN{g2r=atan2(1,1)/45;
                                        {if ($4==07){$18=$18+0.19029};}
  {e1=$14*g2r;a1=$15*g2r;e2=$16*g2r;a2=$17*g2r;printf "%s %14.4f
   %8.4f %8.4f %8.4f \n",$6,$8-$11-$18,-cos(e2)*sin(a2)+cos(e1)*sin(a1),
        -cos(e2)*cos(a2)+cos(e1)*cos(a1),-sin(e2)+sin(e1)}' > L1model.dat
```

Compute the Least Squares solution

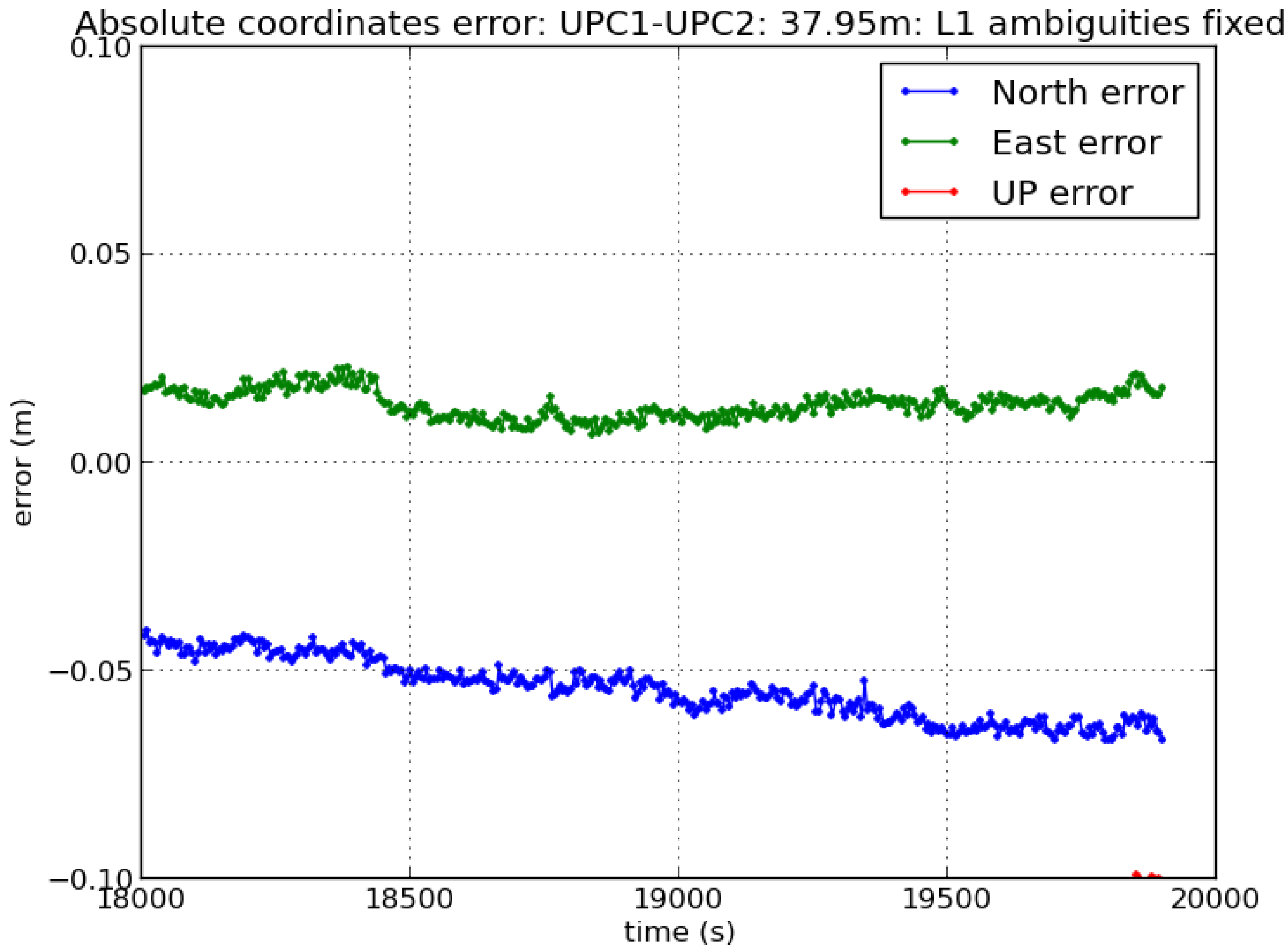
```
cat L1model.dat |LS > L1fix.pos
```

B2.4. UPC1-UPC2 differential positioning with L2 carrier (using the computed differential corrections)

Plot the absolute positioning error:

```
graph.py -f L2fix.pos -x1 –y2- s.- -l "North error"
         -f L2fix.pos -x1 –y3 -s.- -l "East error"
         -f L2fix.pos -x1 –y4 -s.- -l "UP error"
         --yn -.1 --yx .1 --xl "time (s)" --yl "error (m)"
         -t "Absolute positioning error with a wrong ambiguity fix"
```

L1 Differential positioning with a wrong ambiguity fix on a single satellite



Questions:

Discuss the results.

What is the effect of the wrong fix?

OVERVIEW

- **Introduction:** gLAB processing in command line.
- **Preliminary computations:** Data files.
- **Session A:** Fixing DD ambiguities one at a time: UPC1-UPC2.
- **Session B:** Assessing the fixed ambiguities in navigation:
  Differential positioning of UPC1-UPC2 receivers.
- **Session C:** Fixing DD ambiguities with LAMBDA method.

Note: UPC1-UPC2 receivers baseline: 37.95 metres.

Session C

Fixing DD ambiguities with LAMBDA method

(baseline: 37.95 metres)

Apply the LAMBDA method to fix the ambiguities.

Consider only the two epochs: t_1*=18000* and t_2*=18015.*

Note:
To avoid the synchronization issues, consider the Differential Positioning using the computed differential corrections, instead of the time-tagged measurements.

That is, we are going to solve the following navigation equations systems:

1. **Navigating with L1 carrier, to fix DDN1:**

   [DDL1-DDRho]=[Los_k-Los_06]*dr + [A]*[λ_1*DDN1]

2. **Navigating with L2 carrier, to fix DDN2:**

   [DDL2-DDRho]=[Los_k-Los_06]*dr + [A]*[λ_2*DDN2]

Consider again the previous problem of estimating $\Delta\mathbf{r}$, a 3-vector of real numbers , and $\mathbf{N}$ a *(K-1)*-vector of integers, which are solution of

$$\boxed{\mathbf{y} = \mathbf{G}\,\Delta\mathbf{r} + \lambda\,\mathbf{A}\,\mathbf{N} + \mathbf{v}}$$

The solution comprises the following steps:

1. Obtain the float solution and its covariance matrix: $\begin{bmatrix} \Delta\hat{\mathbf{r}} \\ \hat{\mathbf{N}} \end{bmatrix} \; ; \; \begin{bmatrix} \mathbf{P}_{\Delta\hat{\mathbf{r}}} & \mathbf{P}_{\Delta\hat{\mathbf{r}},\hat{\mathbf{N}}} \\ \mathbf{P}_{\Delta\hat{\mathbf{r}},\hat{\mathbf{N}}} & \mathbf{P}_{\hat{\mathbf{N}}} \end{bmatrix}$

2. Find the integer vector $\mathbf{N}$ which minimizes the cost function

$$\boxed{c(\mathbf{N}) = \left\| \mathbf{N} - \hat{\mathbf{N}} \right\|^2_{\mathbf{W}_{\hat{\mathbf{N}}}} = \left(\mathbf{N} - \hat{\mathbf{N}}\right)^T \mathbf{W}_{\hat{\mathbf{N}}} \left(\mathbf{N} - \hat{\mathbf{N}}\right)} \qquad \mathbf{W}_{\hat{\mathbf{N}}} = \mathbf{P}_{\hat{\mathbf{N}}}^{-1}$$

   a) Decorrelation: Using the $\mathbf{Z}$ transform, the ambiguity search space is re-parametrized to decorrelate the float ambiguities.
   b) Integer ambiguities estimation (e.g. by rounding or by using sequential conditional least-squares adjustment, together with a discrete search strategy).
   c) Using the $\mathbf{Z}^{-1}$ transform, the ambiguities are transformed to the original ambiguity space.

3. Obtain the 'fixed' solution $\Delta\mathbf{r}$, from the fixed ambiguities $\mathbf{N}$.

$$\boxed{\mathbf{y} - \lambda\,\mathbf{A}\,\mathbf{N} = \mathbf{G}\,\Delta\mathbf{r} + \mathbf{v}}$$

C1. DDN1 ambiguity fixing: Differential positioning using computed differential corrections from a reference receiver.

Consider only the two epochs: t_1=*18000* and t_2=*18015.*

The following procedure can be applied:

1. **Build-up the navigation system.**
2. **Compute the FLOATED solution**, solving the equations system with octave. Assess the accuracy of the floated solution.
3. **Apply the LAMBDA method to FIX the ambiguities**. Compare the results with the solution obtained by rounding directly the floated solution and by rounding the solution after decorrelation.

1. Building-up the navigation system

```
[DDL1-DDRho]= [Los_k - Los_06]*[dr] + [ A ]*[lambda1*DDN1]
```

Notation

$$\mathbf{y} = \mathbf{G}\,\mathbf{x}$$

$$\begin{bmatrix} DDL_1^{6,03} - DD\rho^{6,03} \\ DDL_1^{6,07} - DD\rho^{6,07} \\ \vdots \\ DDL_1^{6,24} - DD\rho^{6,24} \end{bmatrix} = \begin{bmatrix} -\left(\hat{\boldsymbol{\rho}}^3 - \hat{\boldsymbol{\rho}}^6\right)^T \\ -\left(\hat{\boldsymbol{\rho}}^7 - \hat{\boldsymbol{\rho}}^6\right)^T \\ \vdots \\ -\left(\hat{\boldsymbol{\rho}}^{24} - \hat{\boldsymbol{\rho}}^6\right)^T \end{bmatrix} \mathbf{dr} + \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1\, DDN_1^{6,03} \\ \lambda_1\, DDN_1^{6,07} \\ \vdots \\ \lambda_1\, DDN_1^{6,24} \end{bmatrix}$$

Where the vector of unknowns **x** **includes the user coordinates and ambiguities**

C1 Fixing the DDN1 ambiguities with LAMBDA Method

The receiver was not moving (static) during the data collection.
Thence, for each epoch we have the equations system:

$$\begin{bmatrix} DDL_1^{6,03}(t_1) - DD\rho^{6,03}(t_1) \\ DDL_1^{6,07}(t_1) - DD\rho^{6,07}(t_1) \\ \vdots \\ DDL_1^{6,24}(t_1) - DD\rho^{6,24}(t_1) \end{bmatrix} = \begin{bmatrix} -\left(\hat{\boldsymbol{\rho}}^{3}(t_1) - \hat{\boldsymbol{\rho}}^{6}(t_1)\right)^T \\ -\left(\hat{\boldsymbol{\rho}}^{7}(t_1) - \hat{\boldsymbol{\rho}}^{6}(t_1)\right)^T \\ \vdots \\ -\left(\hat{\boldsymbol{\rho}}^{30}(t_1) - \hat{\boldsymbol{\rho}}^{6}(t_1)\right)^T \end{bmatrix} \mathbf{dr} + \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1 \, DDN_1^{6,03} \\ \lambda_1 \, DDN_1^{6,07} \\ \vdots \\ \lambda_1 \, DDN_1^{6,24} \end{bmatrix}$$

$$\mathbf{y}_1 = \mathbf{G}_1 \, \mathbf{x}$$

y1:=y[t1]
G1:=G[t1]

$$\begin{bmatrix} DDL_1^{6,03}(t_2) - DD\rho^{6,03}(t_2) \\ DDL_1^{6,07}(t_2) - DD\rho^{6,07}(t_2) \\ \vdots \\ DDL_1^{6,24}(t_2) - DD\rho^{6,24}(t_2) \end{bmatrix} = \begin{bmatrix} -\left(\hat{\boldsymbol{\rho}}^{3}(t_2) - \hat{\boldsymbol{\rho}}^{6}(t_2)\right)^T \\ -\left(\hat{\boldsymbol{\rho}}^{7}(t_2) - \hat{\boldsymbol{\rho}}^{6}(t_2)\right)^T \\ \vdots \\ -\left(\hat{\boldsymbol{\rho}}^{30}(t_2) - \hat{\boldsymbol{\rho}}^{6}(t_2)\right)^T \end{bmatrix} \mathbf{dr} + \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1 \, DDN_1^{6,03} \\ \lambda_1 \, DDN_1^{6,07} \\ \vdots \\ \lambda_1 \, DDN_1^{6,24} \end{bmatrix}$$

$$\mathbf{y}_2 = \mathbf{G}_2 \, \mathbf{x}$$

y2:=y[t2]
G2:=G[t2]

```
[DDL1-DDRho]=[Los_k - Los_06]*[dr] + [ A ]*[lambda1*DDN1]
```

C1 Fixing the DDN1 ambiguities with LAMBDA Method

In the previous computation we have not taken into account the correlations between the double differences of measurements. This matrix will be used now, as the LAMBDA method will be applied to FIX the carrier ambiguities.

$$\mathbf{P_y} = 2\sigma^2 \begin{bmatrix} 2 & 1 & \cdots & 1 \\ 1 & 2 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 1 & 2 \end{bmatrix}$$

a) Show that the covariance matrix of DDL1 is given by $\mathbf{P_y}$

b) Given the measurement vectors (y) and Geometry matrices (G) for two epochs

y1:=y[t1] ; G1:=G[t1] ; Py

y2:=y[t2] ; G2:=G[t2] ; Py

show that the user solution and covariance matrix can be computed as:

P=inv(G1'*W*G1+G2'*W*G2);

where: **W=inv(Py)**

x=P*(G1'*W*y1+G2'*W*y2) ;

$$\mathbf{y} = \mathbf{G\,x}; \quad \mathbf{W} = \mathbf{P_y^{-1}}$$

$$\mathbf{x} = (\mathbf{G^T W G})^{-1}\mathbf{G^T W y}$$

$$\mathbf{P} = (\mathbf{G^T W G})^{-1}$$

C1 Fixing the DDN1 ambiguities with LAMBDA Method

The script **MakeL1DifMat.scr** builds the equations system

$$[DDL1-DDRho]=[\ Los\_k-\ Los\_06]*[dr]+[\ A\ ]*[\lambda_1*DDN1]$$

for the two epochs required $t_1=18000$ and $t_2=18015$, using the input file **DD_UPC1_UPC2_06_ALL.dat** generated before.

Execute:

```
MakeL1DifMat.scr DD_UPC1_UPC2_06_ALL.dat 18000 18015
```

The **OUTPUT**

are the files **M1.dat** and **M2.dat** associated with each epoch.

Where:

the columns of files **M.dat** are the vector **y** (first column) and Matrix **G** (next columns)

2. Compute the FLOATED solution (solving the equations system).

The following procedure can be applied

```
octave

load M1.dat
load M2.dat

y1=M1(:,1);
G1=M1(:,2:12);

y2=M2(:,1);
G2=M2(:,2:12);
Py=(diag(ones(1,8))+ones(8))*2e-4;
W=inv(Py);
```

```
P=inv(G1'*W*G1+G2'*W*G2);
x=P*(G1'*W*y1+G2'*W*y2);
```

Solution

```
x(1:3)'
1.4216 -0.6058  0.4035
```

3. Applying the LAMBDA method to FIX the ambiguities.

Compare the results with the solution obtained by rounding the floated solution. The following procedure can be applied **(justify the computations done)**

```
octave

c=299792458;
f0=10.23e+6;
f1=154*f0;
lambda1=c/f1
 a=x(4:11)/lambda1;
 Q=P(4:11,4:11);
```

1. Rounding directly the floated solution

```
round(a)'
0    6    -6    6    6   -1    8    9
```

2.- Rounding the decorrelated floated solution

```
[Qz,Zt,Lz,Dz,az,iZ] = decorrel (Q,a);
afix=iZ*round(az);
2    1    2    -1    4    7    1    4
```

3.- Decorrelation and integer LS search solution

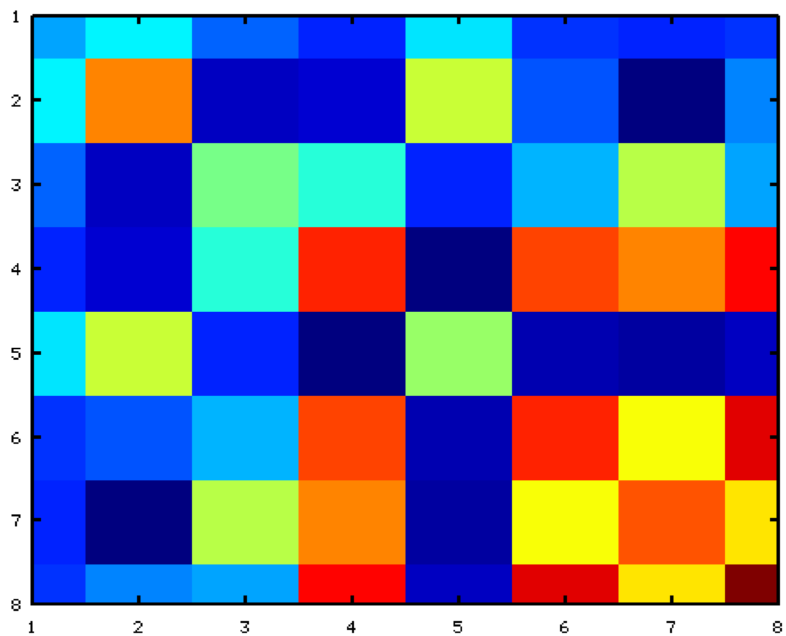
```
[Qz,Zt,Lz,Dz,az,iZ] = decorrel (Q,a);
[azfixed,sqnorm] = lsearch (az,Lz,Dz,2);
afixed=iZ*azfixed;
sqnorm(2)/sqnorm(1)
ans = 3.1069682814451
afixed(:,1)'
2    1    2    -1    4    7    1    4
```

Questions:

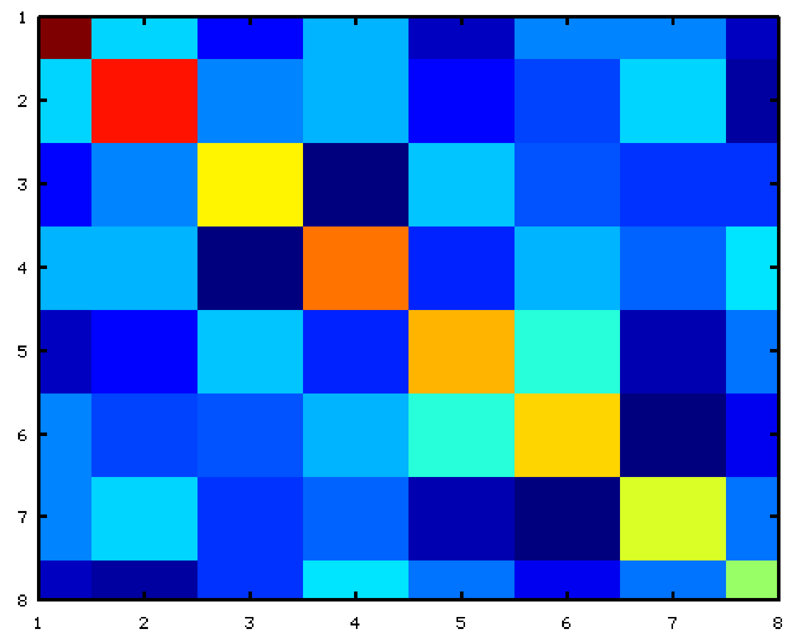
1. *Can the ambiguities be well fixed?*
2. *Is the test resolutive?*
3. *Compare the fixed ambiguities with those obtained in the previous exercises when fixing the ambiguities one at a time. Are the same results found?*
4. *What is the elapsed time to needed fix the ambiguities? And in the previous exercise when fixing the ambiguities one at a time?*
5. *The values found for the ambiguities are the same than in the previous case?*

C2 Checking the Z-transform matrix

a. Using the Octave/MATLAB program sentence **`imagesc`** display the covariance matrix of ambiguities before and after the decorrelation with the Z-matrix.



`imagesc(Q)`



`imagesc(Qz)`

C2 Checking the Z-transform matrix

b.- Show the content of the integer matrix Z

Note: The previous routines computes its transpose (Zt). Then: **Z=Zt'**.

```
Z =Zt’
          3   -5   -4   -5    6    7    4   -2
          3    2   -7   -6   -5   -4    9    3
         -4   -0   -5    8    3   -4    1   -3
          1   -5    1   -8    4   -1    1    8
         -0    1    2    7   -1   -8   -4    3
          8   -3   -1    4    2   -6   -1    4
         -5   -1    1   -6    1    0    1    4
         -5   -3   -0   -0    5   -2   -1    3
```

C2 Checking the Z-transform matrix

c.- Compute by hand the transformed covariance matrix **Qz**:

```
Z * Q * Z'
```

d.- Compute the decorrelated ambiguities **az**:

```
Z*a
    67.19877600816902
   -27.00815309344809
   -52.80792522348074
    49.18410614456196
   -53.87737144457776
   -11.70730035212100
    18.08081880749826
     4.09968790147667
```

C2 Checking the Z-transform matrix

e.- Round-off the decorrelated ambiguities:

```
Nz=round(Z*a)

    67   -27   -53   49   -54   -12   18   4
```

f.- Apply the inverse transform to these values:

```
format short

inv(Z)*Nz
    2.00000 1.00000 2.00000 -1.00000
    4.00000 7.00000 1.00000  4.00000
```

g. - Compare the previous results with the direct rounding of initial ambiguities "**a**":

```
round(a)

       0    -6    6    6   -1    12    8    9
```

C3. DDN2 ambiguity fixing: Differential positioning using computed differential corrections from a reference receiver.

Consider only the two epochs: $t_1=18000$ and $t_2=18015$.

The following procedure can be applied, as in the previous case:

1. **Build-up the navigation system.**
2. **Compute the FLOATED solution**, solving the equations system with octave. Assess the accuracy of the floated solution.
3. **Apply the LAMBDA method to FIX the ambiguities**. Compare the results with the solution obtained by rounding directly the floated solution and by rounding the solution after decorrelation.

1. Building-up the navigation system

```
[DDL2-DDRho]= [Los_k - Los_06]*[dr] + [ A ]*[lambda2*DDN2]
```

Notation

$$\begin{bmatrix} DDL_2^{6,03} - DD\rho^{6,03} \\ DDL_2^{6,07} - DD\rho^{6,07} \\ \vdots \\ DDL_2^{6,24} - DD\rho^{6,24} \end{bmatrix} = \begin{bmatrix} -\left(\hat{\boldsymbol{\rho}}^3 - \hat{\boldsymbol{\rho}}^6\right)^T \\ -\left(\hat{\boldsymbol{\rho}}^7 - \hat{\boldsymbol{\rho}}^6\right)^T \\ \vdots \\ -\left(\hat{\boldsymbol{\rho}}^{24} - \hat{\boldsymbol{\rho}}^6\right)^T \end{bmatrix} \mathbf{dr} + \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \lambda_2\, DDN_2^{6,03} \\ \lambda_2\, DDN_2^{6,07} \\ \vdots \\ \lambda_2\, DDN_2^{6,24} \end{bmatrix}$$

$$\mathbf{y} = \mathbf{G}\,\mathbf{x}$$

Where the vector of unknowns $\underline{\mathbf{x}}$ **includes the user coordinates and ambiguities**

C3 Fixing the DDN2 ambiguities with LAMBDA Method

The receiver was not moving (static) during the data collection.
Thence, for each epoch we have the equations system:

$$\begin{bmatrix} DDL_2^{6,03}(t_1) - DD\rho^{6,03}(t_1) \\ DDL_2^{6,07}(t_1) - DD\rho^{6,07}(t_1) \\ \vdots \\ DDL_2^{6,24}(t_1) - DD\rho^{6,24}(t_1) \end{bmatrix} = \begin{bmatrix} -\left(\hat{\boldsymbol{\rho}}^3(t_1) - \hat{\boldsymbol{\rho}}^6(t_1)\right)^T \\ -\left(\hat{\boldsymbol{\rho}}^7(t_1) - \hat{\boldsymbol{\rho}}^6(t_1)\right)^T \\ \vdots \\ -\left(\hat{\boldsymbol{\rho}}^{30}(t_1) - \hat{\boldsymbol{\rho}}^6(t_1)\right)^T \end{bmatrix} \mathbf{dr} + \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \lambda_2\, DDN_2^{6,03} \\ \lambda_2\, DDN_2^{6,07} \\ \vdots \\ \lambda_2\, DDN_2^{6,24} \end{bmatrix}$$

$$\mathbf{y}_1 = \mathbf{G}_1\,\mathbf{x}$$

y1:=y[t1]
G1:=G[t1]

$$\begin{bmatrix} DDL_2^{6,03}(t_2) - DD\rho^{6,03}(t_2) \\ DDL_2^{6,07}(t_2) - DD\rho^{6,07}(t_2) \\ \vdots \\ DDL_2^{6,24}(t_2) - DD\rho^{6,24}(t_2) \end{bmatrix} = \begin{bmatrix} -\left(\hat{\boldsymbol{\rho}}^3(t_2) - \hat{\boldsymbol{\rho}}^6(t_2)\right)^T \\ -\left(\hat{\boldsymbol{\rho}}^7(t_2) - \hat{\boldsymbol{\rho}}^6(t_2)\right)^T \\ \vdots \\ -\left(\hat{\boldsymbol{\rho}}^{30}(t_2) - \hat{\boldsymbol{\rho}}^6(t_2)\right)^T \end{bmatrix} \mathbf{dr} + \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \lambda_2\, DDN_2^{6,03} \\ \lambda_2\, DDN_2^{6,07} \\ \vdots \\ \lambda_2\, DDN_2^{6,24} \end{bmatrix}$$

$$\mathbf{y}_2 = \mathbf{G}_2\,\mathbf{x}$$

y2:=y[t2]
G2:=G[t2]

```
[DDL2-DDRho]=[Los_k - Los_06]*[dr] + [ A ]*[lambda2*DDN2]
```

C3 Fixing the DDN2 ambiguities with LAMBDA Method

In the previous sessions A and B we have not taken into account the correlations between the double differences of measurements. This matrix will be used now, as the LAMBDA method will be applied to FIX the carrier ambiguities.

a) Show that the covariance matrix of DDL2 is given by **$\mathbf{P_y}$**

$$\mathbf{P_y} = 2\sigma^2 \begin{bmatrix} 2 & 1 & \cdots & 1 \\ 1 & 2 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 1 & 2 \end{bmatrix}$$

b) Given the measurement vectors (y) and Geometry matrices (G) for two epochs

y1:=y[t1] ; G1:=G[t1] ; Py
y2:=y[t2] ; G2:=G[t2] ; Py

show that the user solution and covariance matrix can be computed as:

P=inv(G1'*W*G1+G2'*W*G2);

where: **W=inv(Py)**

x=P*(G1'*W*y1+G2'*W*y2) ;

$$\mathbf{y} = \mathbf{G\,x}; \quad \mathbf{W} = \mathbf{P_y^{-1}}$$

$$\mathbf{x} = (\mathbf{G}^T\mathbf{W}\mathbf{G})^{-1}\mathbf{G}^T\mathbf{W}\mathbf{y}$$

$$\mathbf{P} = (\mathbf{G}^T\mathbf{W}\mathbf{G})^{-1}$$

The script **MakeL2DifMat.scr** builds the equations system

```
[DDL2-DDRho]=[ Los_k- Los_06]*[dr] + [ A ]*[λ2*DDN2]
```

for the two epochs required $t_1=18000$ and $t_2=18015$, using the input file **DD_UPC1_UPC2_06_ALL.dat** generated before.

Execute:

```
MakeL2DifMat.scr DD_UPC1_UPC2_06_ALL.dat 18000 18015
```

The **OUTPUT**

are the files **M1.dat** and **M2.dat** associated with each epoch.

Where:

the columns of files **M.dat** are the vector **y** (first column) and Matrix **G** (next columns)

2. Computing the FLOATED solution (solving the equations system).

The following procedure can be applied

```
octave

load M1.dat
load M2.dat

y1=M1(:,1);
G1=M1(:,2:12);

y2=M2(:,1);
G2=M2(:,2:12);
Py=(diag(ones(1,8))+ones(8))*2e-4;
W=inv(Py);
```

```
P=inv(G1'*W*G1+G2'*W*G2);
x=P*(G1'*W*y1+G2'*W*y2);
```

Solution

```
x(1:3)'
  0.1442  -0.5154   0.5568
```

3. Applying the LAMBDA method to FIX the ambiguities.

Compare the results with the solution obtained by rounding the floated solution.
The following procedure can be applied **(justify the computations done)**

```
octave

c=299792458;
f0=10.23e+6;
f2=120*f0;
lambda2=c/f2
 a=x(4:11)/lambda2;
 Q=P(4:11,4:11);
```

1. Rounding directly the floated solution

```
round(a)'
-1   -2   1   2   1  -2   3   -1
```

2.- Rounding the decorrelated floated solution

```
[Qz,Zt,Lz,Dz,az,iZ] = decorrel (Q,a);
afix=iZ*round(az);
-1   1   -1   2   2   -1   1   0
```

3.- Decorrelation and integer LS search solution

```
[Qz,Zt,Lz,Dz,az,iZ] = decorrel (Q,a);
[azfixed,sqnorm] = lsearch (az,Lz,Dz,2);
afixed=iZ*azfixed;
sqnorm(2)/sqnorm(1)
ans = 3.54056715815950
afixed(:,1)'
-1   1   -1   2   2   -1   1   0
```

C3 Fixing the DDN2 ambiguities with LAMBDA Method

Questions:

1. *Can the ambiguities be well fixed?*
2. *Is the test resolutive?*
3. *Compare the fixed ambiguities with those obtained in the previous exercises when fixing the ambiguities one at a time. Are the same results found?*
4. *What is the elapsed time to needed fix the ambiguities? And in the previous exercise when fixing the ambiguities one at a time?*
5. *The values found for the ambiguities are the same than in the previous case?*

C4. Estimate the baseline vector between UPC1 and UPC2 receivers using the L1 carrier measurements of file (DD_UPC1_UPC2_06_ALL.dat).

Consider only the two epochs used in the previous exercise: t_1=14500 and t_2=14515

The following procedure can be applied, as in the previous case:

1. **Build-up the navigation system.**
2. **Compute the FLOATED solution**, solving the equations system with octave. Assess the accuracy of the floated solution.
3. **Apply the LAMBDA method to FIX the ambiguities**. Compare the results with the solution obtained by rounding directly the floated solution and by rounding the solution after decorrelation.

C4. UPC1-UPC2 Baseline vector estimation with L1 carrier (using the time-tagged reference station measurements)

C4.1 Estimate the baseline vector between UPC1 and UPC2 receivers using the L1 carrier measurements of file (DD_UPC1_UPC2_06_ALL.dat).

```
[DDL1]= [Los_k - Los_06]*[baseline] + [ A ]*[lambda1*DDN1]
```

Notation (for each epoch t)

$$\begin{bmatrix} DDL_1^{6,03} \\ DDL_1^{6,07} \\ \vdots \\ DDL_1^{6,24} \end{bmatrix} = \begin{bmatrix} -\left(\hat{\boldsymbol{\rho}}^3 - \hat{\boldsymbol{\rho}}^6\right)^T \\ -\left(\hat{\boldsymbol{\rho}}^7 - \hat{\boldsymbol{\rho}}^6\right)^T \\ \vdots \\ -\left(\hat{\boldsymbol{\rho}}^{24} - \hat{\boldsymbol{\rho}}^6\right)^T \end{bmatrix} \mathbf{r} + \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1 DDN_1^{6,03} \\ \lambda_1 DDN_1^{6,07} \\ \vdots \\ \lambda_1 DDN_1^{6,24} \end{bmatrix}$$

$$\mathbf{y} = \mathbf{G}\,\mathbf{x}$$

Where the vector of unknowns $\underline{\mathbf{x}}$ **includes the user coordinates and ambiguities**

C4. UPC1-UPC2 Baseline vector estimation with L1 carrier (using the time-tagged reference station measurements)

The receiver was not moving (static) during the data collection.
Therefore, for each epoch we have the equations system:

$$\begin{bmatrix} DDL_1^{6,03}(t_1) \\ DDL_1^{6,07}(t_1) \\ \vdots \\ DDL_1^{6,24}(t_1) \end{bmatrix} = \begin{bmatrix} -\left(\hat{\boldsymbol{\rho}}^3(t_1)-\hat{\boldsymbol{\rho}}^6(t_1)\right)^T \\ -\left(\hat{\boldsymbol{\rho}}^7(t_1)-\hat{\boldsymbol{\rho}}^6(t_1)\right)^T \\ \vdots \\ -\left(\hat{\boldsymbol{\rho}}^{24}(t_1)-\hat{\boldsymbol{\rho}}^6(t_1)\right)^T \end{bmatrix} \mathbf{r} + \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1\, DDN_1^{6,03} \\ \lambda_1\, DDN_1^{6,07} \\ \vdots \\ \lambda_1\, DDN_1^{6,24} \end{bmatrix}$$

$$\mathbf{y}_1 = \mathbf{G}_1\,\mathbf{x}$$

y1:=y[t1]
G1:=G[t1]

$$\begin{bmatrix} DDL_1^{6,03}(t_2) \\ DDL_1^{6,07}(t_2) \\ \vdots \\ DDL_1^{6,24}(t_2) \end{bmatrix} = \begin{bmatrix} -\left(\hat{\boldsymbol{\rho}}^3(t_2)-\hat{\boldsymbol{\rho}}^6(t_2)\right)^T \\ -\left(\hat{\boldsymbol{\rho}}^7(t_2)-\hat{\boldsymbol{\rho}}^6(t_2)\right)^T \\ \vdots \\ -\left(\hat{\boldsymbol{\rho}}^{24}(t_2)-\hat{\boldsymbol{\rho}}^6(t_2)\right)^T \end{bmatrix} \mathbf{r} + \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1\, DDN_1^{6,03} \\ \lambda_1\, DDN_1^{6,07} \\ \vdots \\ \lambda_1\, DDN_1^{6,24} \end{bmatrix}$$

$$\mathbf{y}_2 = \mathbf{G}_2\,\mathbf{x}$$

y2:=y[t2]
G2:=G[t2]

```
[DDL1]=[Los_k - Los_06]*[baseline] + [ A ]*[lambda1*DDN1]
```

C4. UPC1-UPC2 Baseline vector estimation with L1 carrier (using the time-tagged reference station measurements)

In the previous sessions A and B we have not taken into account the correlations between the double differences of measurements. This matrix will be used now, as the LAMBDA method will be applied to FIX the carrier ambiguities.

$$\mathbf{P_y} = 2\sigma^2 \begin{bmatrix} 2 & 1 & \cdots & 1 \\ 1 & 2 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 1 & 2 \end{bmatrix}$$

a) Show that the covariance matrix of DDL1 is given by $\mathbf{P_y}$

b) Given the measurement vectors (**y**) and Geometry matrices (**G**) for two epochs

y1:=y[t1] ; G1:=G[t1] ; Py

y2:=y[t2] ; G2:=G[t2] ; Py

show that the user solution and covariance matrix can be computed as:

P=inv(G1'*W*G1+G2'*W*G2);

x=P*(G1'*W*y1+G2'*W*y2) ;

where: **W=inv(Py)**

$$\mathbf{y} = \mathbf{G}\,\mathbf{x}; \quad \mathbf{W} = \mathbf{P_y^{-1}}$$

$$\mathbf{x} = (\mathbf{G^T W G})^{-1}\mathbf{G^T W y}$$

$$\mathbf{P} = (\mathbf{G^T W G})^{-1}$$

C4. UPC1-UPC2 Baseline vector estimation with L1 carrier (using the time-tagged reference station measurements)

The script **MakeL1BslMat.scr** builds the equations system

```
[DDL1]=[ Los_k- Los_06]*[baseline] + [ A ]*[λ1*DDN1]
```

for the two epochs required $t_1=18000$ and $t_2=18015$, using the input file **DD_UPC1_UPC2_06_ALL.dat** generated before.

Execute:

```
MakeL1BslMat.scr DD_UPC1_UPC2_06_ALL.dat 18000 18015
```

The **OUTPUT**

are the files **M1.dat** and **M2.dat** associated with each epoch.

Where:

the columns of files **M.dat** are the vector **y** (first column) and Matrix **G** (next columns)

C4. UPC1-UPC2 Baseline vector estimation with L1 carrier (using the time-tagged reference station measurements)

1. Computing the FLOATED solution (solving the equations system).

The following procedure can be applied:

```
octave

load M1.dat
load M2.dat

y1=M1(:,1);
G1=M1(:,2:12);

y2=M2(:,1);
G2=M2(:,2:12);
Py=(diag(ones(1,8))+ones(8))*2e-4;
W=inv(Py);
```

```
P=inv(G1'*W*G1+G2'*W*G2);
x=P*(G1'*W*y1+G2'*W*y2);

x(1:3)'
-24.5735   -27.1121     3.0021
bsl_enu =[-27.4170 -26.2341 -0.0304]

x(1:3)'-bsl_enu
ans= 2.84348  -0.87798 3.03248
```

C4. UPC1-UPC2 Baseline vector estimation with L1 carrier (using the time-tagged reference station measurements)

3. Applying the LAMBDA method to FIX the ambiguities.

Compare the results with the solution obtained by rounding the floated solution. The following procedure can be applied **(justify the computations done)**

```
octave

c=299792458;
f0=10.23e+6;
f1=154*f0;
lambda1=c/f1
 a=x(4:11)/lambda1;
 Q=P(4:11,4:11);
```

1. Rounding directly the floated solution

```
round(a)'
-4   -20   13   11   -14   19   10   16
```

2.- Rounding the decorrelated floated solution

```
[Qz,Zt,Lz,Dz,az,iZ] = decorrel (Q,a);
afix=iZ*round(az);
-1    -12    18    2    -4    8    9    4
```

3.- Decorrelation and integer LS search solution

```
[Qz,Zt,Lz,Dz,az,iZ] = decorrel (Q,a);
[azfixed,sqnorm] = lsearch (az,Lz,Dz,2);
afixed=iZ*azfixed;
sqnorm(2)/sqnorm(1)
ans = 1.22717645070483
afixed(:,1)'
-1    -12    18    2    -4    8    9    4
```

C4. UPC1-UPC2 Baseline vector estimation with L1 carrier (using the time-tagged reference station measurements)

Questions:

1. *Can the ambiguities be well fixed?*
2. *Is the test resolutive?*
3. *Compare the fixed ambiguities with those obtained in the previous exercises when fixing the ambiguities one at a time. Are the same results found?*
4. *What is the elapsed time to needed fix the ambiguities? And in the previous exercise when fixing the ambiguities one at a time?*
5. *The values found for the ambiguities are the same than in the previous case?*

C5. UPC1-UPC2 Baseline vector estimation with L2 carrier (using the time-tagged reference station measurements)

C5. Estimate the baseline vector between UPC1 and UPC2 receivers using the L2 carrier measurements of file (DD_UPC1_UPC2_06_ALL.dat).

Consider only the two epochs used in the previous exercise: t_1=14500 and t_2=14515

The following procedure can be applied, as in the previous case:

1. **Build-up the navigation system.**
2. **Compute the FLOATED solution**, solving the equations system with octave. Assess the accuracy of the floated solution.
3. **Apply the LAMBDA method to FIX the ambiguities**. Compare the results with the solution obtained by rounding directly the floated solution and by rounding the solution after decorrelation.

C5.1 Estimate the baseline vector between UPC1 and UPC2 receivers using the L2 carrier measurements of file (DD_UPC1_UPC2_06_ALL.dat).

```
[DDL2]= [Los_k - Los_06]*[baseline] + [ A ]*[lambda2*DDN2]
```

Notation (for each epoch t)

$$
\begin{bmatrix} DDL_2^{6,03} \\ DDL_2^{6,07} \\ \vdots \\ DDL_2^{6,24} \end{bmatrix} = \begin{bmatrix} -\left(\hat{\boldsymbol{\rho}}^3 - \hat{\boldsymbol{\rho}}^6\right)^T \\ -\left(\hat{\boldsymbol{\rho}}^7 - \hat{\boldsymbol{\rho}}^6\right)^T \\ \vdots \\ -\left(\hat{\boldsymbol{\rho}}^{24} - \hat{\boldsymbol{\rho}}^6\right)^T \end{bmatrix} \mathbf{r} + \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \lambda_2\, DDN_2^{6,03} \\ \lambda_2\, DDN_2^{6,07} \\ \vdots \\ \lambda_2\, DDN_2^{6,24} \end{bmatrix}
$$

$$\mathbf{y} = \mathbf{G}\,\mathbf{x}$$

Where the vector of unknowns $\underline{\mathbf{x}}$ **includes the user coordinates and ambiguities**

C5. UPC1-UPC2 Baseline vector estimation with L2 carrier (using the time-tagged reference station measurements)

The receiver was not moving (static) during the data collection.

Therefore, for each epoch we have the equations system:

$$\begin{bmatrix} DDL_2^{6,03}(t_1) \\ DDL_2^{6,07}(t_1) \\ \vdots \\ DDL_2^{6,24}(t_1) \end{bmatrix} = \begin{bmatrix} -\left(\hat{\boldsymbol{\rho}}^3(t_1)-\hat{\boldsymbol{\rho}}^6(t_1)\right)^T \\ -\left(\hat{\boldsymbol{\rho}}^7(t_1)-\hat{\boldsymbol{\rho}}^6(t_1)\right)^T \\ \vdots \\ -\left(\hat{\boldsymbol{\rho}}^{24}(t_1)-\hat{\boldsymbol{\rho}}^6(t_1)\right)^T \end{bmatrix} \mathbf{r} + \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \lambda_2\, DDN_2^{6,03} \\ \lambda_2\, DDN_2^{6,07} \\ \vdots \\ \lambda_2\, DDN_2^{6,24} \end{bmatrix}$$

$$\mathbf{y}_1 = \mathbf{G}_1\, \mathbf{x}$$

y1:=y[t1]
G1:=G[t1]

$$\begin{bmatrix} DDL_2^{6,03}(t_2) \\ DDL_2^{6,07}(t_2) \\ \vdots \\ DDL_2^{6,24}(t_2) \end{bmatrix} = \begin{bmatrix} -\left(\hat{\boldsymbol{\rho}}^3(t_2)-\hat{\boldsymbol{\rho}}^6(t_2)\right)^T \\ -\left(\hat{\boldsymbol{\rho}}^7(t_2)-\hat{\boldsymbol{\rho}}^6(t_2)\right)^T \\ \vdots \\ -\left(\hat{\boldsymbol{\rho}}^{24}(t_2)-\hat{\boldsymbol{\rho}}^6(t_2)\right)^T \end{bmatrix} \mathbf{r} + \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \lambda_2\, DDN_2^{6,03} \\ \lambda_2\, DDN_2^{6,07} \\ \vdots \\ \lambda_2\, DDN_2^{6,24} \end{bmatrix}$$

$$\mathbf{y}_2 = \mathbf{G}_2\, \mathbf{x}$$

y2:=y[t2]
G2:=G[t2]

```
[DDL2]=[Los_k - Los_06]*[baseline] + [ A ]*[lambda2*DDN2]
```

C5. UPC1-UPC2 Baseline vector estimation with L2 carrier (using the time-tagged reference station measurements)

In the previous sessions A and B we have not taken into account the correlations between the double differences of measurements. This matrix will be used now, as the LAMBDA method will be applied to FIX the carrier ambiguities.

a) Show that the covariance matrix of DDL2 is given by $\mathbf{P_y}$

$$\mathbf{P_y} = 2\sigma^2 \begin{bmatrix} 2 & 1 & \cdots & 1 \\ 1 & 2 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 1 & 2 \end{bmatrix}$$

b) Given the measurement vectors (**y**) and Geometry matrices (**G**) for two epochs

y1:=y[t1] ; G1:=G[t1] ; Py
y2:=y[t2] ; G2:=G[t2] ; Py

show that the user solution and covariance matrix can be computed as:

P=inv(G1'*W*G1+G2'*W*G2);

where: **W=inv(Py)**

x=P*(G1'*W*y1+G2'*W*y2) ;

$$\mathbf{y} = \mathbf{G}\,\mathbf{x}; \quad \mathbf{W} = \mathbf{P_y^{-1}}$$

$$\mathbf{x} = (\mathbf{G}^T\mathbf{W}\mathbf{G})^{-1}\mathbf{G}^T\mathbf{W}\mathbf{y}$$

$$\mathbf{P} = (\mathbf{G}^T\mathbf{W}\mathbf{G})^{-1}$$

C5. UPC1-UPC2 Baseline vector estimation with L2 carrier (using the time-tagged reference station measurements)

The script **MakeL1BslMat.scr** builds the equations system

```
[DDL2]=[ Los_k- Los_06]*[baseline] + [ A ]*[λ2*DDN2]
```

for the two epochs required $t_1=18000$ and $t_2=18015$, using the input file **DD_UPC1_UPC2_06_ALL.dat** generated before.

Execute:

```
MakeL2BslMat.scr DD_UPC1_UPC2_06_ALL.dat 18000 18015
```

The **OUTPUT**

are the files **M1.dat** and **M2.dat** associated with each epoch.

Where:

the columns of files **M.dat** are the vector **y** (first column) and Matrix **G** (next columns)

1. Computing the FLOATED solution (solving the equations system).

The following procedure can be applied:

```
octave

load M1.dat
load M2.dat

y1=M1(:,1);
G1=M1(:,2:12);

y2=M2(:,1);
G2=M2(:,2:12);
Py=(diag(ones(1,8))+ones(8))*2e-4;
W=inv(Py);
```

```
P=inv(G1'*W*G1+G2'*W*G2);
x=P*(G1'*W*y1+G2'*W*y2);

x(1:3)'
-25.85097  -27.02162    3.15538
bsl_enu =[-27.4170 -26.2341 -0.0304]

x(1:3)'-bsl_enu
ans= 1.5660  -0.7875 3.18578
```

3. Applying the LAMBDA method to FIX the ambiguities.

Compare the results with the solution obtained by rounding the floated solution.
The following procedure can be applied **(justify the computations done)**

```
octave

c=299792458;
f0=10.23e+6;
f2=120*f0;
lambda2=c/f2
 a=x(4:11)/lambda2;
 Q=P(4:11,4:11);
```

1. Rounding directly the floated solution

```
round(a)'
-5  -13  6  7  -9  4  4  4
```

2.- Rounding the decorrelated floated solution

```
[Qz,Zt,Lz,Dz,az,iZ] = decorrel (Q,a);
afix=iZ*round(az);
3   2   9   -27   13   -32  -13   -35
```

3.- Decorrelation and integer LS search solution

```
[Qz,Zt,Lz,Dz,az,iZ] = decorrel (Q,a);
[azfixed,sqnorm] = lsearch (az,Lz,Dz,2);
afixed=iZ*azfixed;
sqnorm(2)/sqnorm(1)
ans =  1.00508811343751
afixed(:,1)'
-3   7   -6   13   -3   19   -3   24
```

C5 Fixing the DDN2 ambiguities with LAMBDA Method

Questions:

1. Can the ambiguities be well fixed?
2. Is the test resolutive?
3. Compare the fixed ambiguities with those obtained in the previous exercises when fixing the ambiguities one at a time. Are the same results found?
4. What is the elapsed time to needed fix the ambiguities? And in the previous exercise when fixing the ambiguities one at a time?
5. The values found for the ambiguities are the same than in the previous case?

Thanks for your attention

Acknowledgements

- To the University of Delft for the MATLAB files of LAMBDA method.
- To Adrià Rovira-Garcia for his contribution to the editing of this material and gLAB updating and integrating this learning material into the GLUE.

The ESA/UPC GNSS-Lab Tool suite (gLAB) has been developed under the ESA Education Office contract N. P1081434.