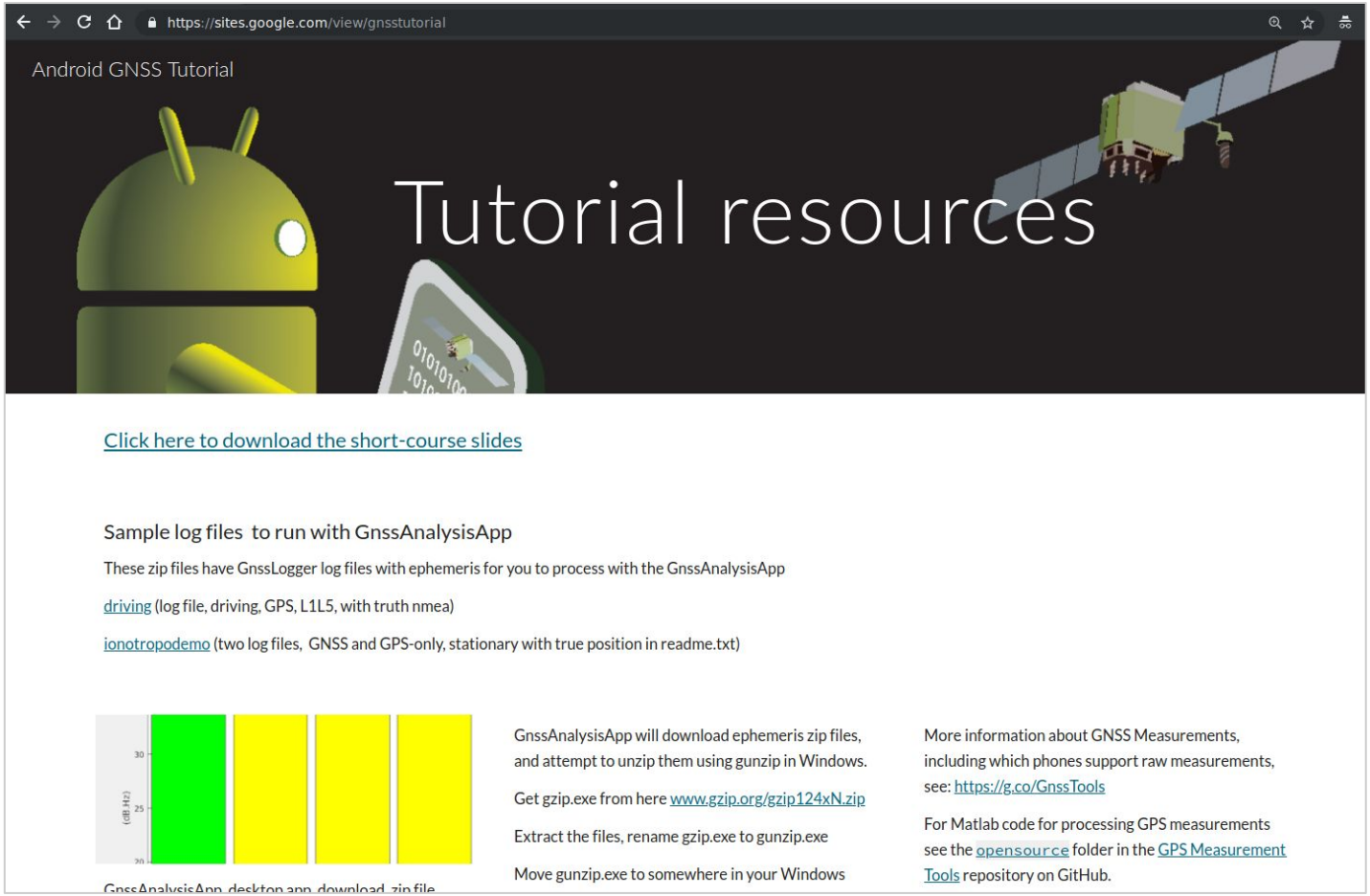
Mobile Phones for Ionospheric Measurements

ION Africa Outreach,
Regional Workshop on GNSS and Space Weather
Rabat, Morocco, May 2022

Frank van Diggelen, Mohammed Khider

Google

This course has a support site: https://sites.google.com/view/gnsstutorial



Overview

1. Raw GNSS Measurements
2. Logging Tools
3. How to get Raw Pseudorange
4. Analysis Tools (and smoothed pseudorange)
5. Hands-on Exercises

Tips, tricks and knowledge to take away

- Use Desktop Analysis Tool to create the derived data file for you
  - this is where you find pseudorange, and smoothed pseudorange
- Do analysis on data from the derived data file
- Beware the weak, and low elevation, signals - filter them out before analyzing the atmosphere
  - use elevation mask (≥ 15 degrees)
  - use CustomDataFilter.m
- Be aware of inter-system and inter-signal biases
- Beware of the ADR (Accumulated Delta Range) residual analysis for inferring anything about the atmosphere - there is a 1 mm/s drift in the residuals (cause unknown).

Location APIs, Measurement APIs

aka *Google Play Services* aka *Google Mobile Service*
Most Android phones have this (not China)

Location APIs, android.gms.location

- Places
- Geofencing
- Fused Location Provider (FLP)
- Fit
- Activity Recognition
- Nearby

All Android phones have this

Measurement/Sensor APIs, in android.location

- Location
- GnssMeasurement
- GnssClock

GNSS Raw Measurements
All phones with:
GNSS chips build date ≥ 2016
OS ≥ Android N (Nougat)

Overview

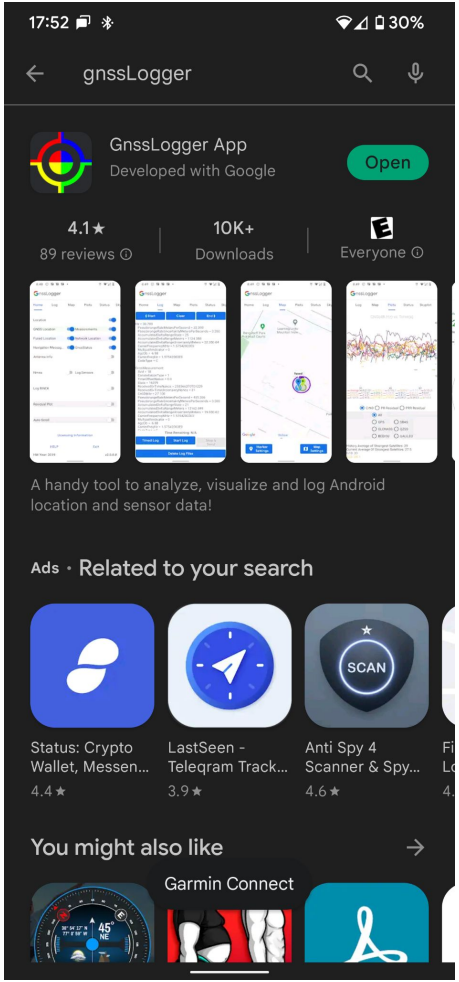
1. Raw GNSS Measurements
2. Logging Tools
3. How to get Raw Pseudorange
4. Analysis Tools (and smoothed pseudorange)
5. Hands-on Exercises

GNSS Logger

GNSS Analysis

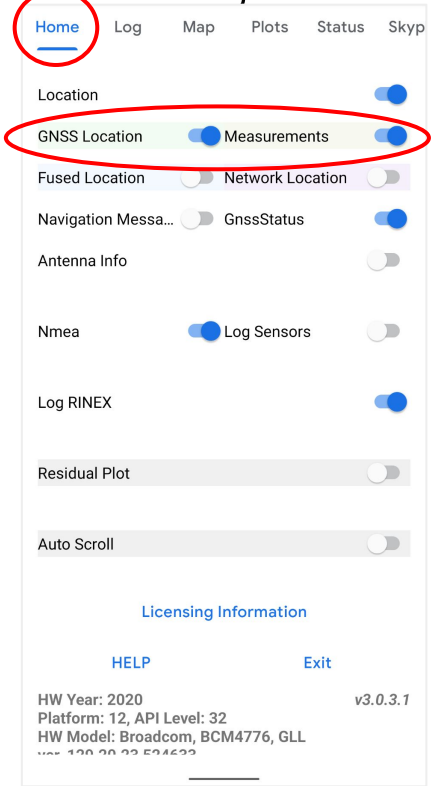
https://g.co/GnssTools

GnssLogger App for phone, see app store:

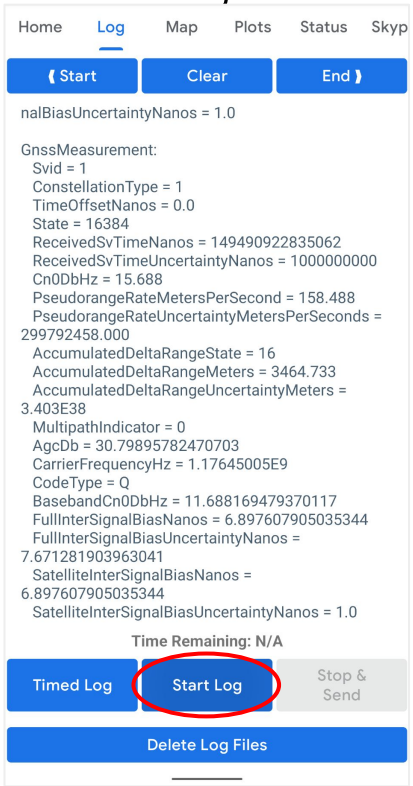


Logging the raw data on your phone:

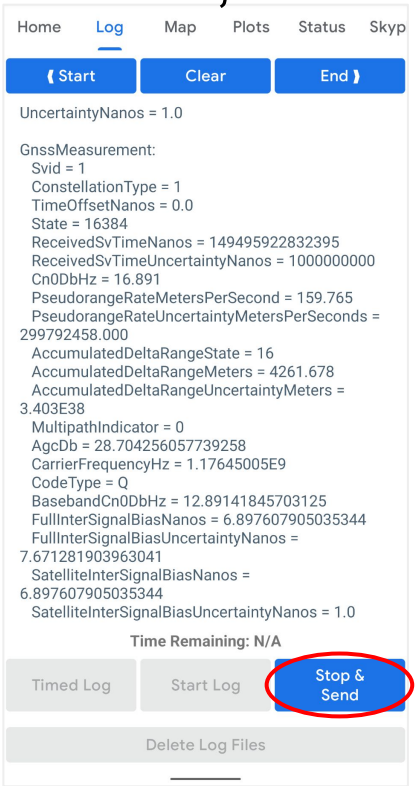
1,



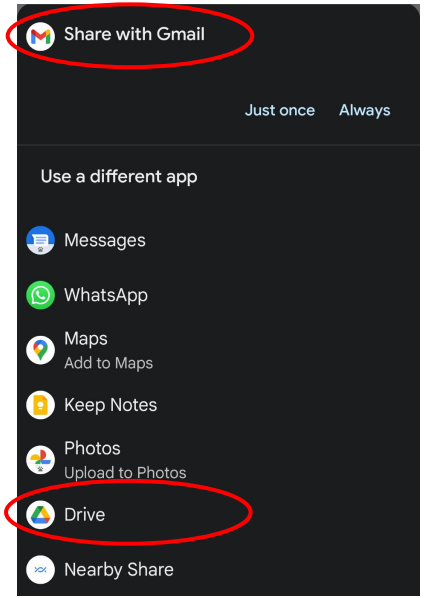
2,



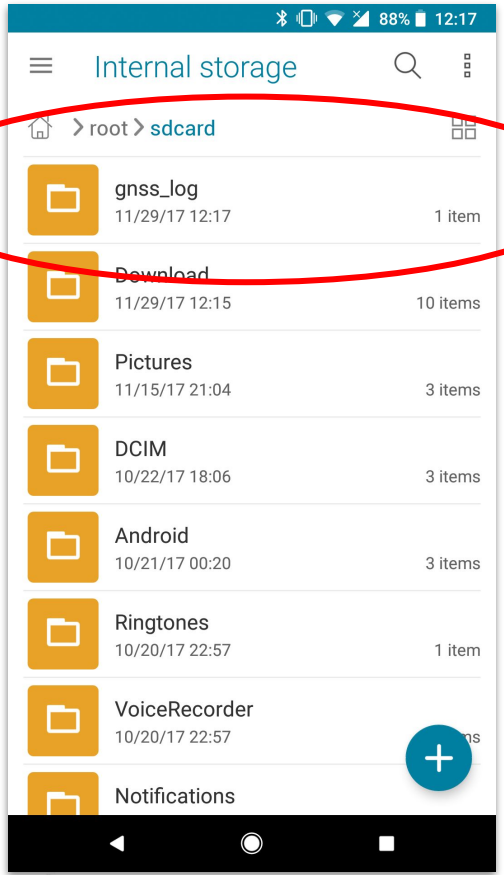
3,



4.



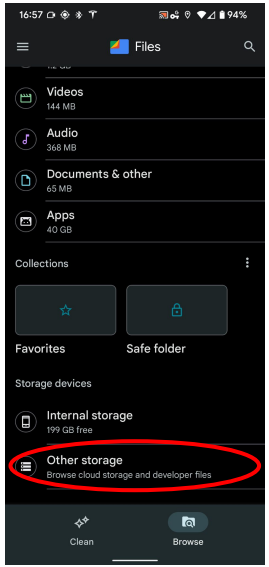
Logged Data is stored locally, on the phone:

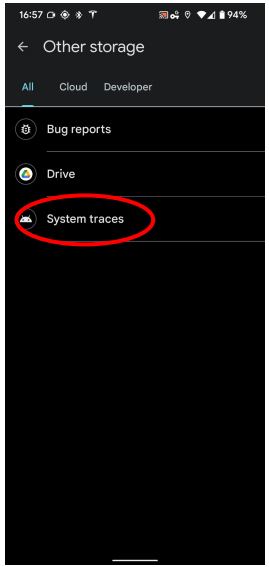


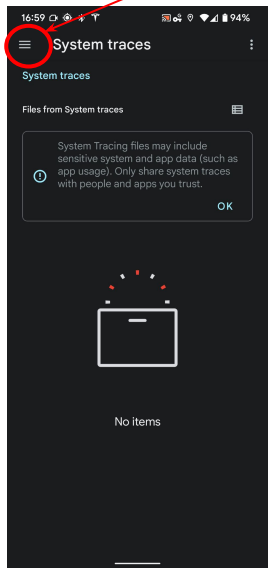
GNSS Logger data is stored locally on your phone.
Google doesn't get any data from these log files.

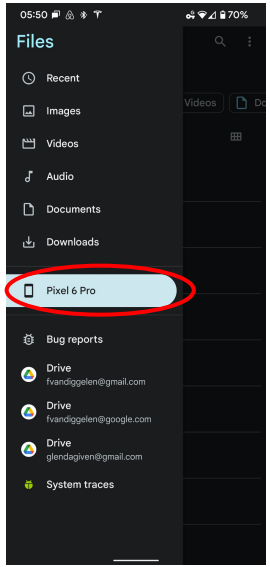
From Android 11+

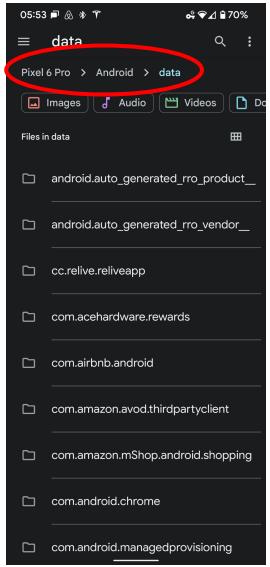
Files app: Other storage, System traces, “Hamburger” menu ☰, *Phone name*, Android, data, com.google.android.apps.location.gps.gnsslogger, files,gnss_log

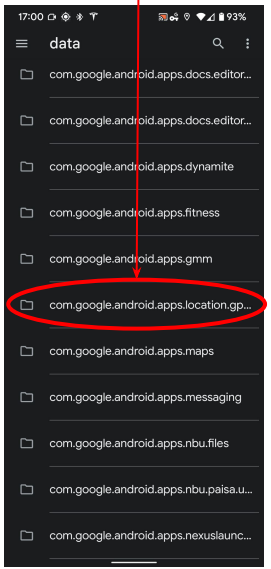


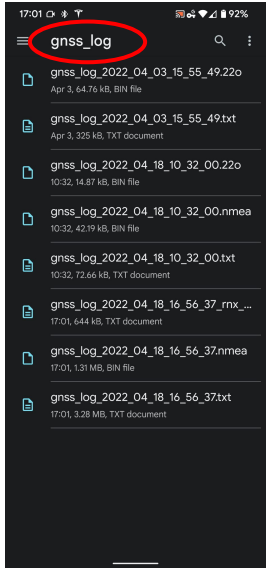












or, using adb (Android Debug Bridge):

adb root

adb pull . /storage/self/primary/Android/data/com.google.android.apps.location.gps.gnsslogger/files/gnss_log/ /destinationFolder

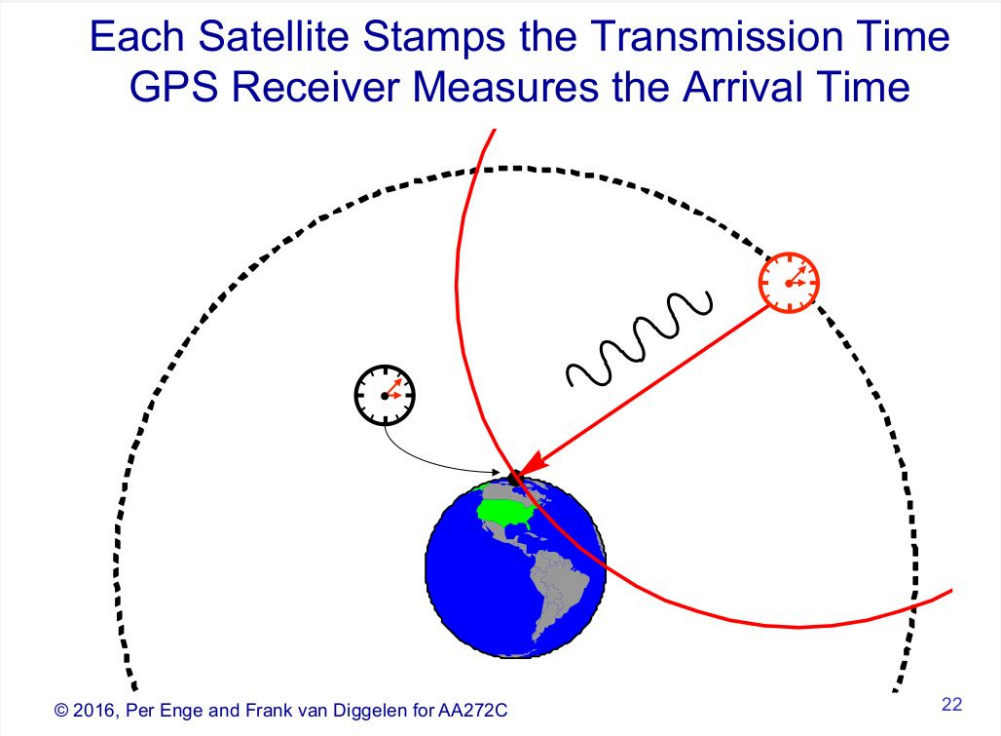
Google GNSS Logger data is stored locally on your phone. Google doesn’t get any data from these log files.

Overview

1. Raw GNSS Measurements
2. Logging Tools
3. How to get Raw Pseudorange
4. Analysis Tools (and smoothed pseudorange)
5. Hands-on Exercises
6. Future: Apps and Research

Most important takeaway:
Understand tRx and tTx

A reminder of what we mean by “pseudorange”



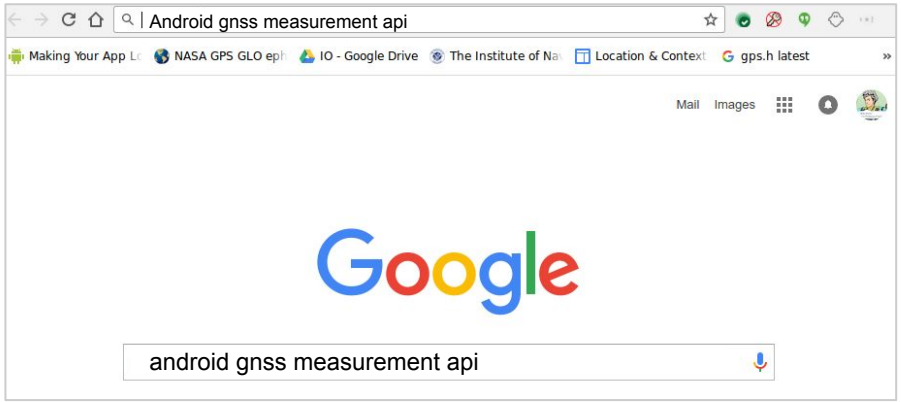
Source: Stanford University, Course AA272C “Introduction to GPS”, Per Enge & Frank van Diggelen

Pseudorange (in units of time)
= Arrival Time - Transmission Time

Pseudorange (in units of distance)
= (Arrival Time - Transmission Time)*c

To find the GNSS Measurement APIs ...

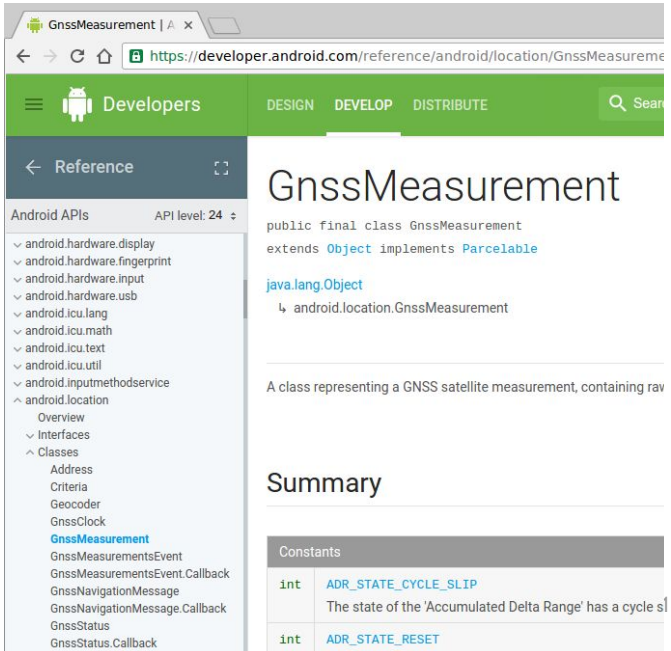
1)



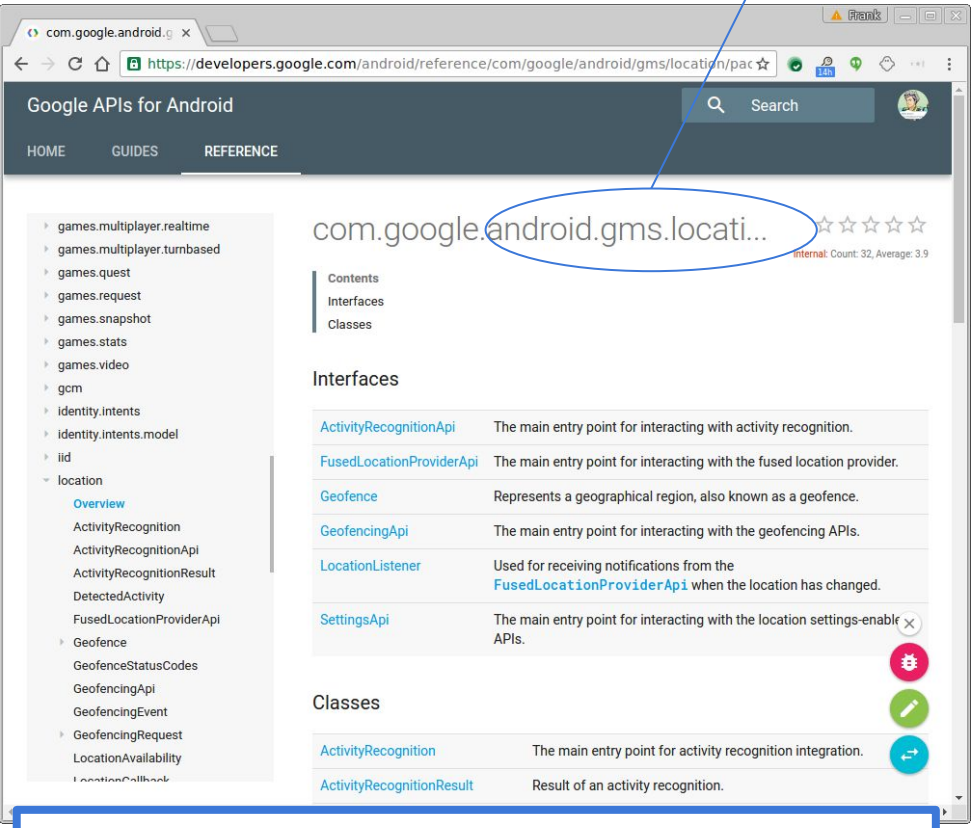
2)

GnssMeasurement | Android Developers
https://developer.**android**.com/.../**android**/.../**GnssMeasurement**.h... ▾ Android ▾
A class representing a **GNSS** satellite **measurement**, containing raw and ... This **GNSS measurement's** tracking state has code lock. .... Added in **API** level 24.

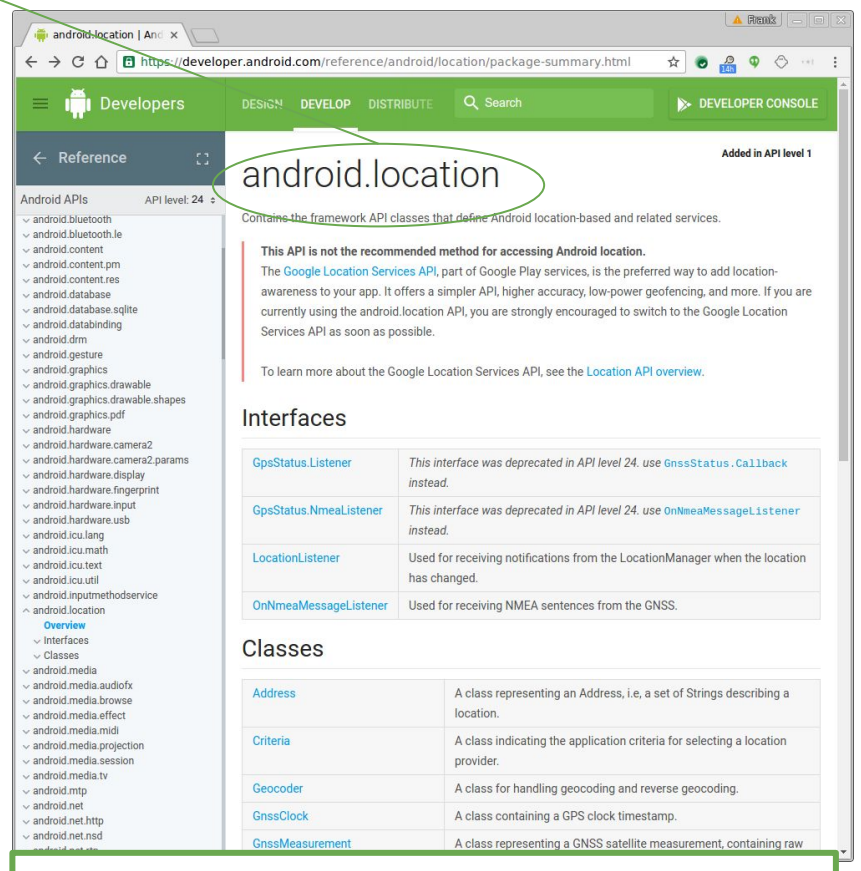
3)



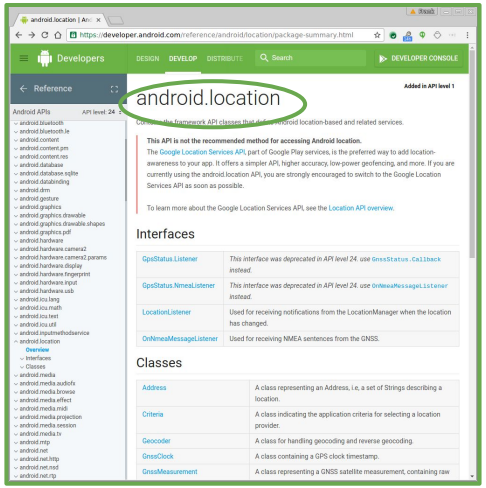
In Section 1 we saw the difference between this, and this.



Google Play Services, higher abstractions of location



android.location, for Raw Measurements

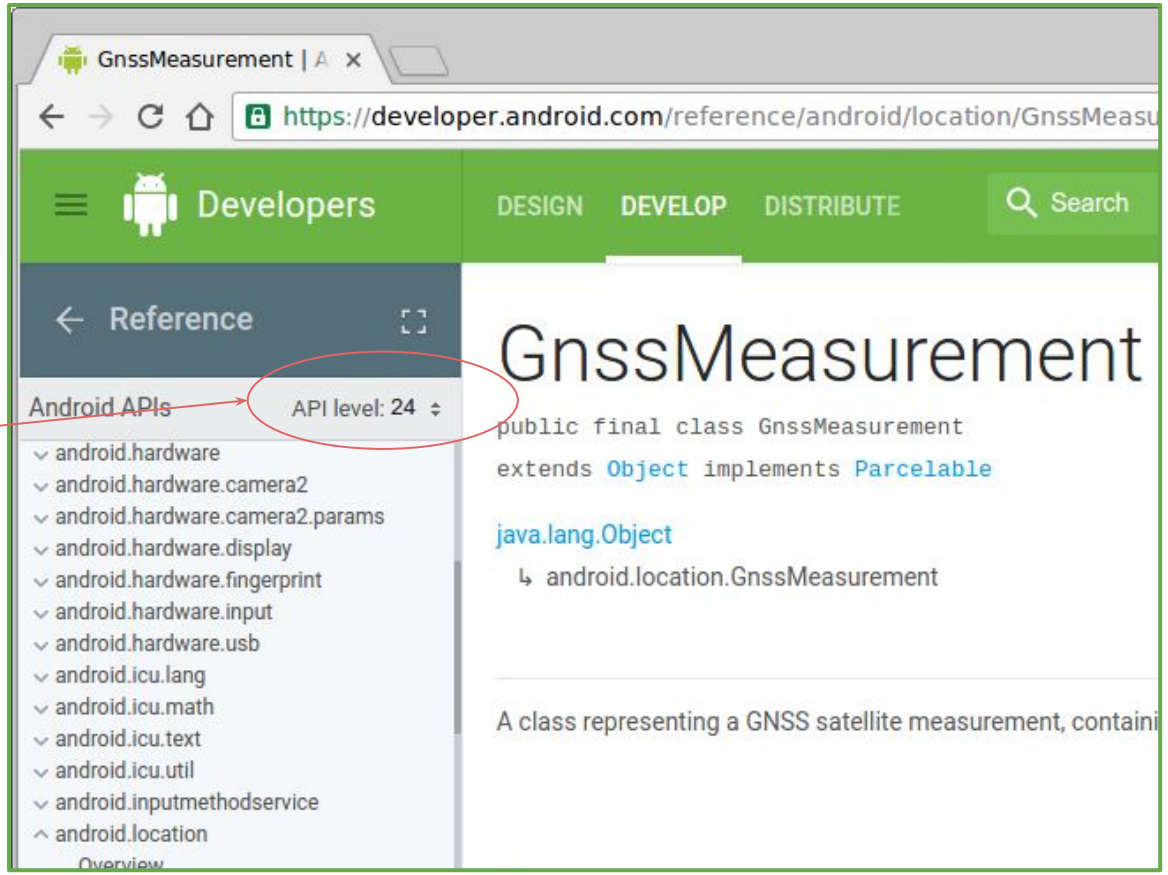


Stylistic note for this section:

Green-outlined box,
means an extract from android.location APIs

How to make sure you're looking at Android release "N" or later

API level 24 = Release N



https://developer.android.com/reference/android/location/GnssMeasurementsEvent.html

Developers DESIGN DEVELOP DISTRIBUTE Search DEVELOPER CONSOLE

Reference

Android APIs API level: 24

- android.inputmethodservice
- android.location
  - Overview
  - Interfaces
  - Classes
    - Address
    - Criteria
    - Geocoder
    - GnssClock
    - GnssMeasurement
    - **GnssMeasurementsEvent**
    - GnssMeasurementsEvent.Callback
    - GnssNavigationMessage
    - GnssNavigationMessage.Callback
    - GnssStatus
    - GnssStatus.Callback
    - GpsSatellite
    - GpsStatus
    - Location
    - LocationManager
    - LocationProvider
    - SettingInjectorService
- android.media
- android.media.audiofx
- android.media.browse
- android.media.effect
- android.media.midi
- android.media.projection
- android.media.session
- android.media.tv
- android.mtp
- android.net
- android.net.http
- android.net.nsd
- android.net.rtp
- android.net.sip
- android.net.wifi
- android.net.wifi.p2p
- android.net.wifi.p2p.nsd

A class implementing a container for data associated with a measurement event. Events are delivered to registered instances of `GnssMeasurementsEvent.Callback`.

Summary

Nested classes	
class	`GnssMeasurementsEvent.Callback` Used for receiving GNSS satellite measurements from the GNSS engine.

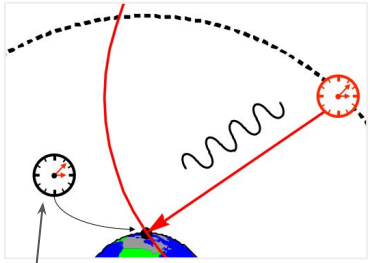
Inherited constants	
From interface an	

Fields	
public static fina	
Creator<GnssMeasu	

Public methods	
int	`describeContents()` Describe the kinds of special objects contained in this Parcelable instance's marshaled representation.
GnssClock	`getClock()` Gets the GNSS receiver clock information associated with the measurements for the current event.
Collection<GnssMeasurement>	`getMeasurements()` Gets a read-only collection of measurements associated with the current event.
String	`toString()`

GnssClock	`getClock()` Gets the GNSS receiver clock information associated with the measurements for the current event.
Collection<GnssMeasurement>	`getMeasurements()` Gets a read-only collection of measurements associated with the current event.

GnssMeasurementEvent.Callback



This is the time tag of the GNSS Measurement

https://developer.android.com/reference/android/location/GnssClock.html

Developers DESIGN DEVELOP DISTRIBUTE Search DEVELOPER CONSOLE

Reference

Android APIs API level: 24

- android.icu.text
- android.icu.util
- android.inputmethodservice
- android.location
  - Overview
  - Interfaces
  - Classes
    - Address
    - Criteria
    - Geocoder
    - **GnssClock**
    - GnssMeasurement
    - GnssMeasurementsEvent

GnssClock

Added in API level 24

Summary: Inherited Constants | Fields | Methods | Inherited Methods | [Expand All]

```
public final class GnssClock
extends Object implements Parcelable
```

java.lang.Object
↳ android.location.GnssClock

Public methods

int	`describeContents()` Describe the kinds of special objects contained in this Parcelable instance's marshaled representation.
double	`getBiasNanos()` Gets the clock's sub-nanosecond bias.
double	`getBiasUncertaintyNanos()` Gets the clock's Bias Uncertainty (1-Sigma) in nanoseconds.
double	`getDriftNanosPerSecond()` Gets the clock's Drift in nanoseconds per second.
double	`getDriftUncertaintyNanosPerSecond()` Gets the clock's Drift Uncertainty (1-Sigma) in nanoseconds per second.
long	`getFullBiasNanos()` Gets the difference between hardware clock (`getTimeNanos()`) inside GPS receiver and the true GPS time since 0000Z, January 6, 1980, in nanoseconds.
int	`getHardwareClockDiscontinuityCount()` Gets count of hardware clock discontinuities.
int	`getLeapSecond()` Gets the leap second associated with the clock's time.
long	`getTimeNanos()` Gets the GNSS receiver internal hardware clock value in nanoseconds.
double	`getTimeUncertaintyNanos()` Gets the clock's time Uncertainty (1-Sigma) in nanoseconds.

getTimeNanos

Added in API level 24

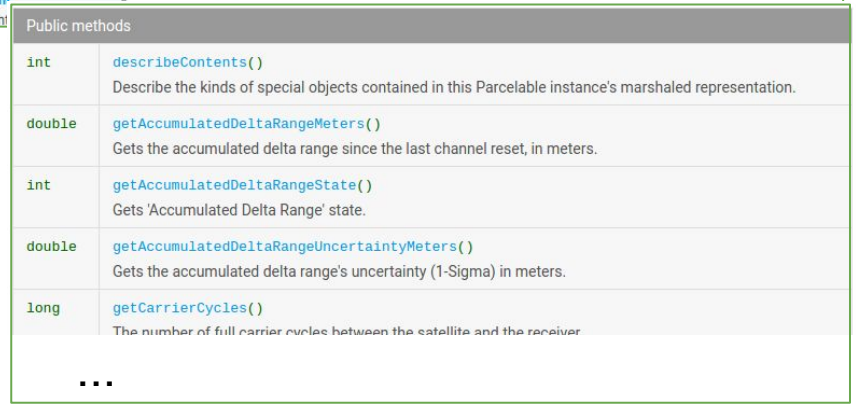
long getTimeNanos ()

Gets the GNSS receiver internal hardware clock value in nanoseconds.

This value is expected to be monotonically increasing while the hardware clock remains powered on. For the case of a hardware clock that is not continuously on, see the getHardwareClockDiscontinuityCount() field.

The GPS time can be derived by subtracting the sum of getFullBiasNanos() and getBiasNanos() (when they are available) from this value. Sub-nanosecond accuracy can be provided by means of getBiasNanos().

The error estimate for this value (if applicable) is getTimeUncertaintyNanos().



Public methods

- describeContents()
- getAccumulatedDeltaRangeMeters()
- getAccumulatedDeltaRangeState()
- getAccumulatedDeltaRangeUncertaintyMeters()
- getCarrierCycles()
- getCarrierFrequencyHz()
- getCarrierPhase()
- getCarrierPhaseUncertainty()
- getCn0DbHz()
- getConstellationType()
- getMultipathIndicator()
- getPseudorangeRateMetersPerSecond()
- getPseudorangeRateUncertaintyMetersPerSecond()
- getReceivedSvTimeNanos()
- getReceivedSvTimeUncertaintyNanos()
- getSnrInDb()
- getState()
- getSvid()
- getTimeOffsetNanos()
- hasCarrierCycles()
- hasCarrierFrequencyHz()
- hasCarrierPhase()
- hasCarrierPhaseUncertainty()
- hasSnrInDb()
- toString()
- writeToParcel(Parcel parcel, int flags)

So, how do you get pseudorange?

https://developer.android.com/reference/android/location/GnssMeasurement.html

Developers DESIGN DEVELOP DISTRIBUTE Search

Reference

Android APIs API level: 24

- android.inputmethodservice
- android.location
  - Overview
  - Interfaces
  - Classes
    - Address
    - Criteria
    - Geocoder
    - GnssClock
    - GnssMeasurement
    - GnssMeasurement

GnssMeasurement

Summary: Constants | Inher
Methods | Inherite

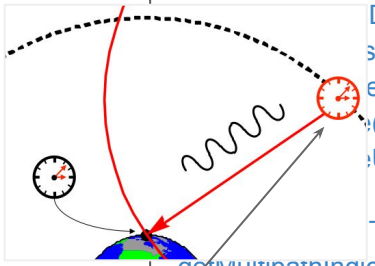
```
public final class GnssMeasurement
extends Object implements Parcelable
```

java.lang.Object
↳ android.location.GnssMeasurement

Public methods

int	describeContents() Describe the kinds of special objects contained in this Parcelable instance's marshaled representation.
double	getAccumulatedDeltaRangeMeters() Gets the accumulated delta range since the last channel reset, in meters.
int	getAccumulatedDeltaRangeState() Gets 'Accumulated Delta Range' state.
double	getAccumulatedDeltaRangeUncertaintyMeters() Gets the accumulated delta range's uncertainty (1-Sigma) in meters.
long	getCarrierCycles() The number of full carrier cycles between the satellite and the receiver.

...



Public methods

- describeContents()
- getAccumulatedDeltaRangeMeters()
- getAccumulatedDeltaRangeState()
- DeltaRangeUncertaintyMeters()
- s()
- encyHz()
- ()
- Uncertainty()
- Type()
- getMultipathIndicator()
- getPseudorangeRateMetersPerSecond()
- getPseudorangeRateUncertaintyMetersPerSecond()
- getReceivedSvTimeNanos()
- getReceivedSvTimeUncertaintyNanos()
- getSnrInDb()
- getState()
- getSvid()
- getTimeOffsetNanos()
- hasCarrierCycles()
- hasCarrierFrequencyHz()
- hasCarrierPhase()
- hasCarrierPhaseUncertainty()
- hasSnrInDb()
- toString()
- writeToParcel(Parcel parcel, int flags)

You get ReceivedSvTimeNanos,
and from that you make the pseudorange

Why isn't pseudorange provided explicitly?

Because about half of all GNSS location on smartphones happens *before* time (TOW) is fully known.

getReceivedSvTimeNanos

```
long getReceivedSvTimeNanos ()
```

Gets the received GNSS satellite time, at the measurement time, in nanoseconds.

For GPS & QZSS, this is:

- Received GPS Time-of-Week at the measurement time, in nanoseconds.
- The value is relative to the beginning of the current GPS week.

Given the highest sync state that can be achieved, per each satellite, valid range for this field can be:

```
     Searching       : [ 0       ]   : STATE_UNKNOWN
     C/A code lock   : [ 0   1ms ]   : STATE_CODE_LOCK is set
     Bit sync        : [ 0  20ms ]   : STATE_BIT_SYNC is set
     Subframe sync   : [ 0    6s ]   : STATE_SUBFRAME_SYNC is set
     TOW decoded     : [ 0 1week ]   : STATE_TOW_DECODED is set
```

Smartphone GNSSes make extensive use of measurements long before TOW is decoded. That is how they get TTFF of 1 to 2 seconds [1]

These measurements are considered invalid in traditional GNSS. So you can use Android raw measurements to create RTCM and RINEX format log files, but not vice-versa without losing information.

For example: when only this bit is set, ReceivedSvTimeNanos is a value from zero to one millisecond.

[1] See "*A-GPS*" Book, van Diggelen, Chapter 4 "Coarse Time Navigation: Instant GPS"

Examples of SvTime < 20ms

K	M	N	O	AB
Svid	State	ReceivedSvTimeNanc	ReceivedSvTimeUncertaintyNanos	ConstellationType
7	17	4161153	6	5
9	1074	7769046	14	6
22	1074	6586891	6	6
30	1074	6540385	11	6
2	47	164773920061633	17	1

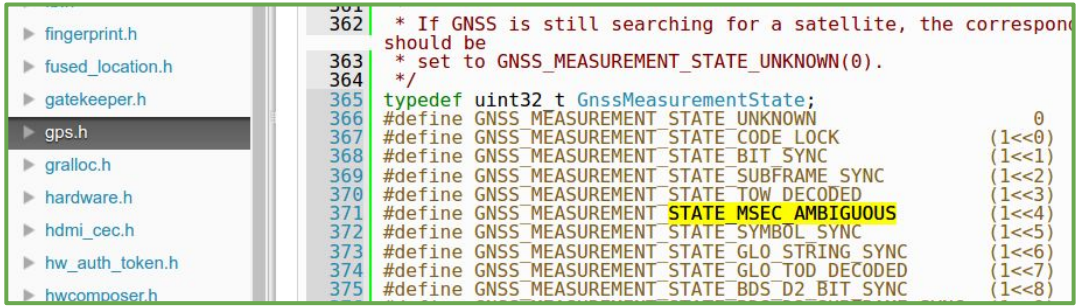
BeiDou

Galileo

GPS

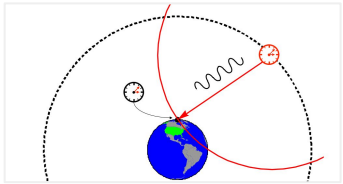
4,161,153 ns = 4.1 ms

State = 17 = 0001 0001



For the rest of this short-course, we'll focus on GPS measurements where TOW is known.

How to get pseudorange:



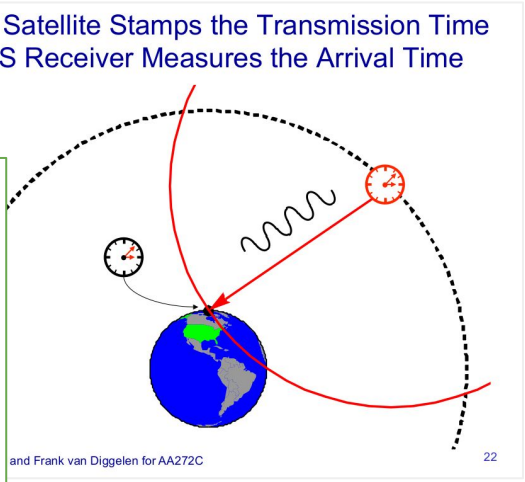
Pseudorange
= (Arrival Time - Transmission Time)*c

Two methods:

1) construct it from the various time values (complicated)

2) let the Gnss Analysis Tools desktop app to it (recommended)

How to get GPS pseudorange (1):



From GnssClock

public long getTimeNanos ()

Added in API level 24

Gets the GNSS receiver internal hardware clock value in nanoseconds.

This value is expected to be monotonically increasing while the hardware clock remains powered on. For the case of a hardware clock that is not continuously on, see the getHardwareClockDiscontinuityCount() field.

The GPS time can be derived by subtracting the sum ofgetFullBiasNanos() and getBiasNanos() (when they are available) from this value. Sub-nanosecond accuracy can be provided by means ofgetBiasNanos().

From GnssMeasurement

public long getReceivedSvTimeNanos ()

Added in API level 24

Gets the received GNSS satellite time, at the measurement time, in nanoseconds.

For GPS & QZSS, this is:

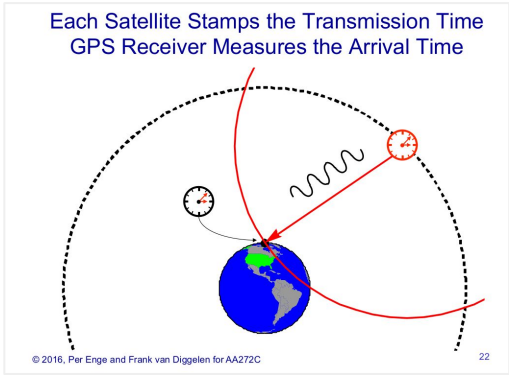
Received GPS Time-of-Week at the measurement time, in nanoseconds.

The value is relative to the beginning of the current GPS week.

getFullBiasNanos()
Gets the difference between hardware clock (getTimeNanos()) inside GPS receiver and the true GPS time since 0000Z, January 6, 1980, in nanoseconds.

You must adjust the GnssClock value to the same time reference as GnssMeasurement

How to get GPS pseudorange (2):



From GnssClock

The GPS time can be derived by subtracting the sum of `getFullBiasNanos()` and `getBiasNanos()`

tRxNanos = TimeNanos - (FullBiasNanos + BiasNanos) - WeekNumberNanos

Referenced to start time

Referenced to GPS Epoch

Referenced to GPS Week

From GnssMeasurement

tTxNanos = ReceivedSvTimeNanos

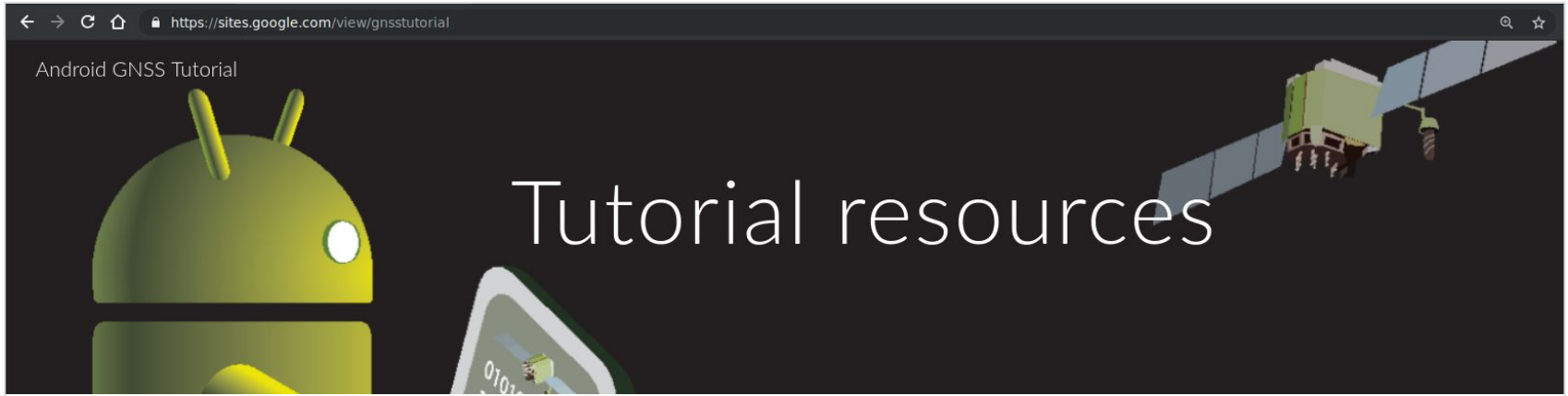
Referenced to GPS Week

getReceivedSvTimeNanos ()

For GPS & QZSS, this is:

- Received GPS Time-of-Week at the measurement time, in nanoseconds.
- The value is relative to the beginning of the current GPS week

Access the spreadsheet for the pseudorange exercise: https://sites.google.com/view/gnsstutorial



File Edit View Insert Format Data Tools Add-ons Help

$ % .0 .00 123 Arial 10

TimeNanos

B	E	F	J
imeNanos	FullBiasNanos	BiasNanos	HardwareCloc
72076939000000	-1151285108458178048	0	188
72076939000000	-1151285108458178048	0	188
72076939000000	-1151285108458178048	0	188

raw.csv prsWithPrInMeters Large number problem

Example Spreadsheet for working with raw measurements directly

← scroll down to find this

Google Android App: GnssLogger, all these values are read to a log file.

tRxNanos = TimeNanos - (FullBiasNanos + BiasNanos) - WeekNumberNanos

tTxNanos = ReceivedSvTimeNanos

	B	D	E	F	G	H	I	J	K
1	TimeNanos	TimeUncertaintyNanos	FullBiasNanos	BiasNanos	BiasUncertaintyNanos	Svid	TimeOffsetNanos	State	ReceivedSvTimeNanos
2	72076939000000		-1151285108458178048	0	26.54	2	0	15	422785326362991
3	72076939000000		-1151285108458178048	0	26.54	3	0	15	422785311363053
4	72076939000000		-1151285108458178048	0	26.54	6	0	15	422785328163761
5	72076939000000		-1151285108458178048	0	26.54	12	0	15	422785324936930
6	72076939000000		-1151285108458178048	0	26.54	17	0	15	422785318856058
7	72076939000000		-1151285108458178048	0	26.54	19	0	15	422785325657035
8	72076939000000		-1151285108458178048	0	26.54	24	0	15	422785327049795
9	72076939000000		-1151285108458178048	0	26.54	25	0	15	422785314216671
10	72076939000000		-1151285108458178048	0	26.54	28	0	15	422785314964982

prs.csv | prsWithPrInMeters | Large number problem

%GPS Week number:

```
weekNumber = floor(-double(gnssRaw.FullBiasNanos)*1e-9/GpsConstants.WEEKSEC);
```

Code snippet from MATLAB/gpstools/opensource/ProcessGnssMeas.m

Google Android App: GnssLogger, and one more thing: TimeOffsetNanos

double getTimeOffsetNanos ()
Gets the time offset at which the measurement was taken in nanoseconds.
The reference receiver's time from which this is offset is specified by getTimeNanos().
The sign of this value is given by the following equation:

measurement time = TimeNanos + TimeOffsetNanos

The value provides an individual time-stamp for the measurement, and allows sub-nanosecond accuracy.

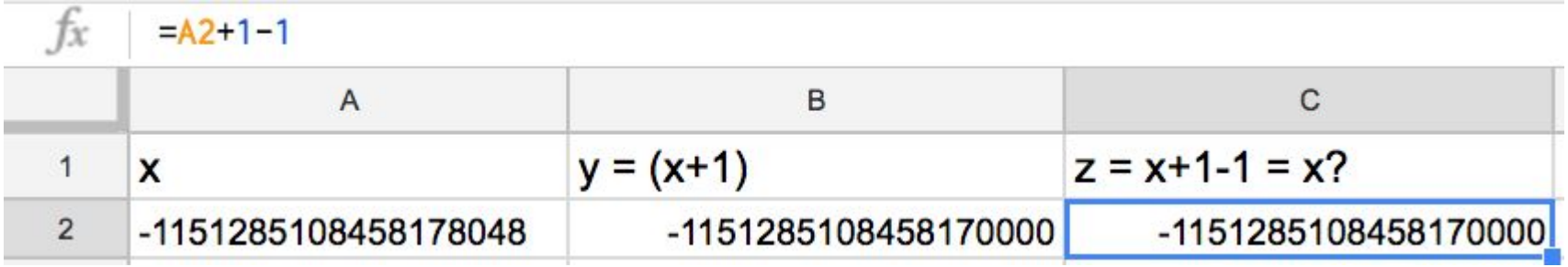
	B	D	E	F	G	H	I	J	K
1	TimeNanos	TimeUncertaintyNanos	FullBiasNanos	BiasNanos	BiasUncertaintyNanos	Svid	TimeOffsetNanos	State	ReceivedSvTimeNanos
2	72076939000000		-1151285108458178048	0	26.54	2	0	15	422785326362991
3	72076939000000		-1151285108458178048	0	26.54	3	0	15	422785311363053
4	72076939000000		-1151285108458178048	0	26.54	6	0	15	422785328163761
5	72076939000000		-1151285108458178048	0	26.54	12	0	15	422785324936930
6	72076939000000		-1151285108458178048	0	26.54	17	0	15	422785318856058
7	72076939000000		-1151285108458178048	0	26.54	19	0	15	422785325657035
8	72076939000000		-1151285108458178048	0	26.54	24	0	15	422785327049795
9	72076939000000		-1151285108458178048	0	26.54	25	0	15	422785314216671
10	72076939000000		-1151285108458178048	0	26.54	28	0	15	422785314964982

tRxNanos = TimeNanos - (FullBiasNanos + BiasNanos) - WeekNumberNanos *from previous slide*

= (TimeNanos + TimeOffsetNanos) - (FullBiasNanos + BiasNanos) - WeekNumberNanos

Making pseudoranges in the worksheet.

ARITHMETIC HEALTH WARNING:
We use worksheets for *illustration only* - because you will get precision errors of the order of 1000 ns because of the large numbers, especially FullBiasNanos



See sample code in later slide for how to do this without losing resolution

Making GPS pseudoranges in the worksheet *(for illustration only)*

tRxNanos = (TimeNanos + TimeOffsetNanos) - (FullBiasNanos + BiasNanos) - *WeekNumberNanos* = *tRxNanos*

fx =(B2+I2)-(E2+F2)-K2*604800000000000

	B	E	F	H	I	J	K	L	M	N	O
1	TimeNanos	FullBiasNanos	BiasNanos	Svid	TimeOffsetNanos	State	*Week #*	*tRxNanos*	TxNanos = ReceivedSvTimeNanos	*Pseudorange in ms 1e-6*(tRxNanos-tTxNanos)*	*Pseudorange in meters*
2	72076939000000	-1151285108458178048	0	2	0	15	*1903*	*422785397169920*	422785326362991	*70.807*	*21,227,383*
3	72076939000000	-1151285108458178048	0	3	0	15	*1903*	*422785397169920*	422785311363053	*85.807*	*25,724,252*
4	72076939000000	-1151285108458178048	0	6	0	15	*1903*	*422785397169920*	422785328163761	*69.006*	*20,687,526*
5	72076939000000	-1151285108458178048	0	12	0	15	*1903*	*422785397169920*	422785324936930	*72.233*	*21,654,906*
6	72076939000000	-1151285108458178048	0	17	0	15	*1903*	*422785397169920*	422785318856058	*78.314*	*23,477,905*
7	72076939000000	-1151285108458178048	0	19	0	15	*1903*	*422785397169920*	422785325657035	*71.513*	*21,439,024*
8	72076939000000	-1151285108458178048	0	24	0	15	*1903*	*422785397169920*	422785327049795	*70.120*	*21,021,485*
9	72076939000000	-1151285108458178048	0	25	0	15	*1903*	*422785397169920*	422785314216671	*82.953*	*24,868,758*
10	72076939000000	-1151285108458178048	0	28	0	15	*1903*	*422785397169920*	422785314964982	*82.205*	*24,644,420*
11	72077939000000	-1151285108458178048	0	2	0	15	*1903*	*422786397169920*	422786326364257	*70.806*	*21,227,004*
12	72077939000000	-1151285108458178048	0	3	0	15	*1903*	*422786397169920*	422786311362544	*85.807*	*25,724,404*

prs.csv | prsWithPrInMeters | Large number problem

Key:
Italics = derived values
Normal = read from GnssLogger

Making GPS pseudoranges in Matlab

First we get gnssRaw from: [gnssRaw] = ReadGnssLogger(dirName,prFileName);

```
%GPS Week number:
weekNumber = floor(-double(gnssRaw.FullBiasNanos)*1e-9/GpsConstants.WEEKSEC);

%compute time of measurement relative to start of week
%subtract big longs (i.e. time from 1980) before casting time of week as double
WEEKNANOS = int64(GpsConstants.WEEKSEC*1e9);
weekNumberNanos = int64(weekNumber)*int64(GpsConstants.WEEKSEC*1e9);
tRxNanos = gnssRaw.TimeNanos -gnssRaw.FullBiasNanos - weekNumberNanos;
%tRxNanos is now since the beginning of the week

%subtract the fractional offsets TimeOffsetNanos and BiasNanos:
tRxSeconds  = (double(tRxNanos)-gnssRaw.TimeOffsetNanos-gnssRaw.BiasNanos)*1e-9;
tTxSeconds  = double(gnssRaw.ReceivedSvTimeNanos)*1e-9;

prSeconds  = tRxSeconds - tTxSeconds;

PrM            = prSeconds*GpsConstants.LIGHTSPEED;
```

Note: when we deal with these numbers, subtract large integers first.

Code snippet from MATLAB/gpstools/opensource/ProcessGnssMeas.m

Open-source code

GNSS Logger apk (app) and GNSS Analysis source code is available on GitHub

- See links on https://g.co/GnssTools
- GNSS Logger is Java code
- GNSS Analysis is Matlab code
- In both cases only GPS analysis code is available as open-source
- You can submit contributions (e.g. add other GNSS open-source code).

The compiled version of GNSS Logger and GNSS Analysis are fully GNSS compatible (i.e. GPS, GLONASS, BeiDou Galileo, QZSS).

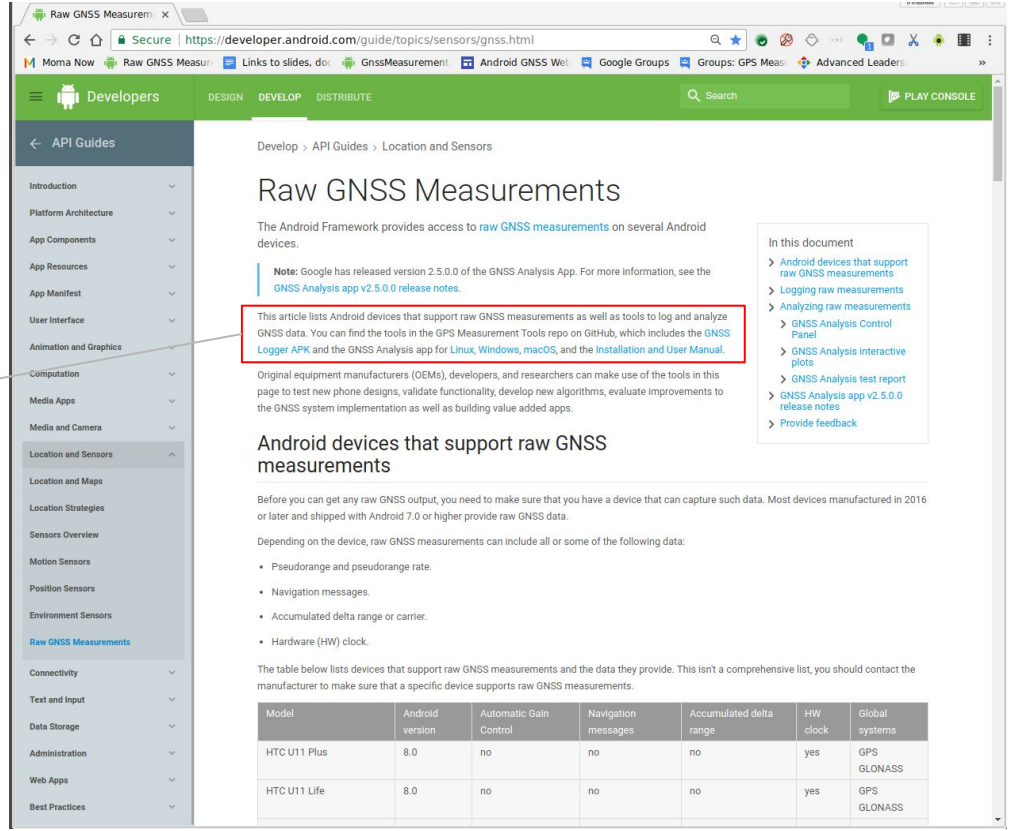
Overview

1. Raw GNSS Measurements
2. Logging Tools
3. How to get Raw Pseudorange
4. Analysis Tools (incl. pseudorange and smoothed pseudorange)
5. Hands-on Exercises
6. Future: Apps and Research

https://g.co/GnssTools

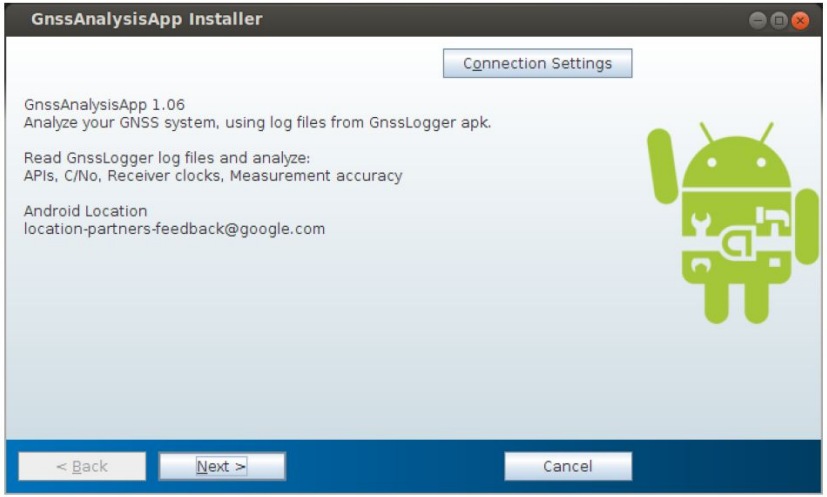
Links to tools:

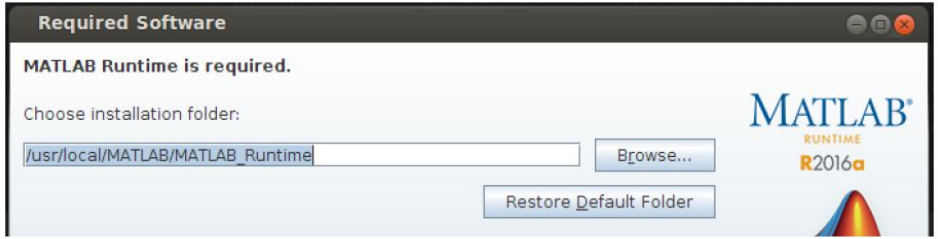
... find the tools in the GPS Measurement Tools repo on GitHub, which includes the GNSS Logger APK and the GNSS Analysis app for
Linux,
Windows,
macOS,
and the
Installation and User Manual.

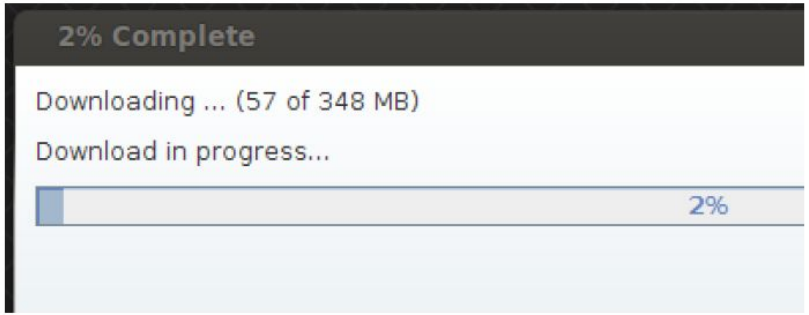


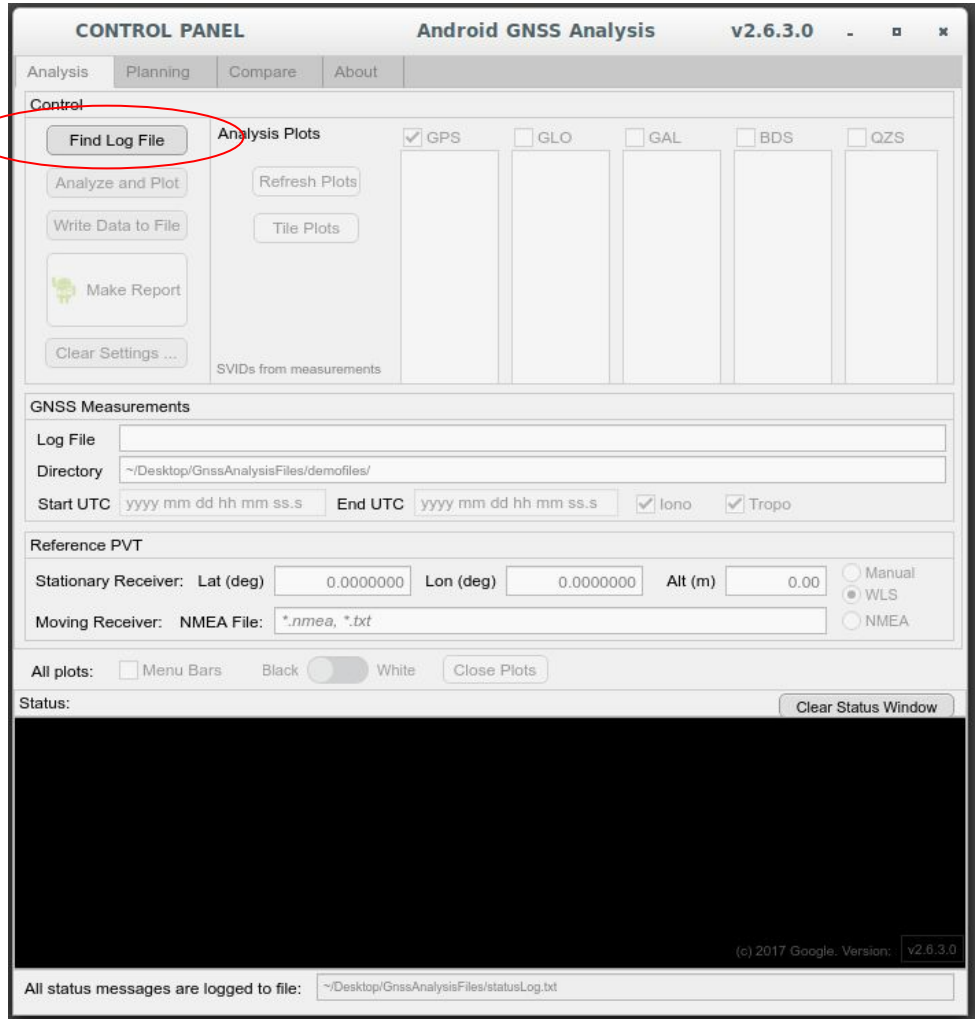
Installing desktop Analysis Tools

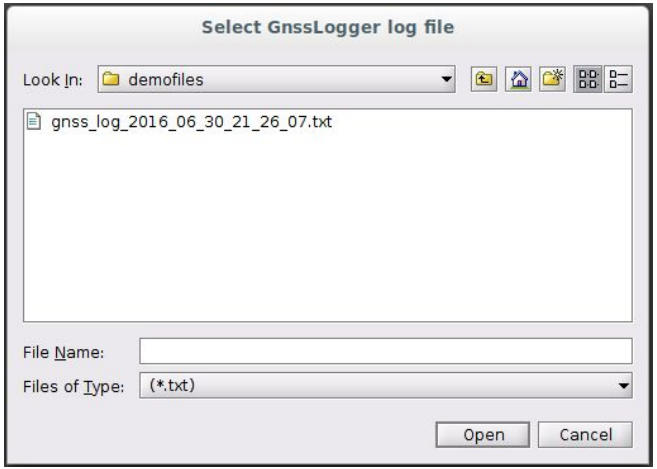
Read the manual, follow carefully

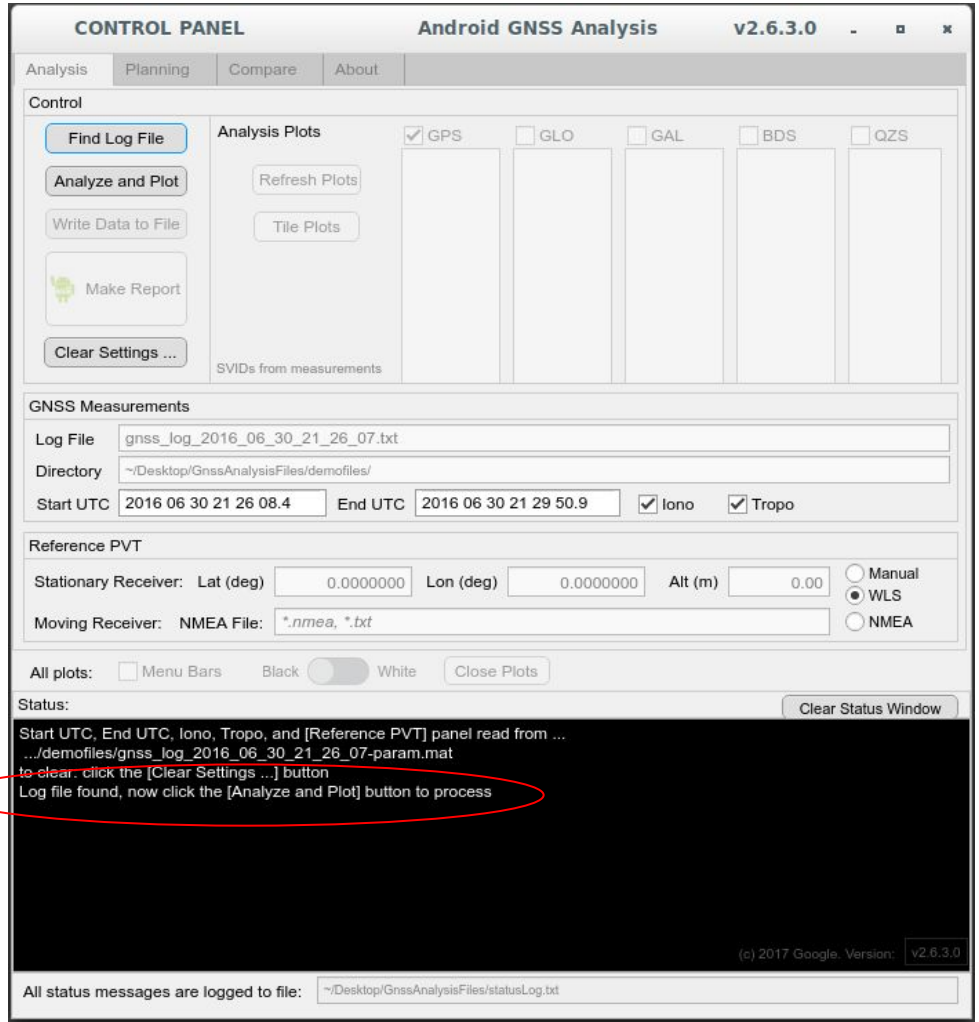




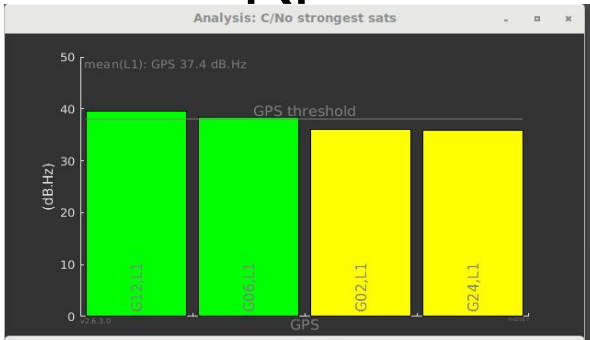


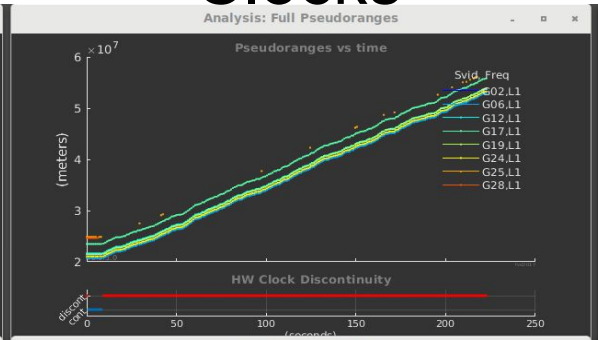

CONTROL PANEL
Android GNSS Analysis
v2.6.3.0
Analysis
Planning
Compare
About
Control
Find Log File
Analyze and Plot
Write Data to File
Make Report
Clear Settings ...
Analysis Plots
Refresh Plots
Tile Plots
SVIDs from measurements
GPS
GLO
GAL
BDS
QZS
GNSS Measurements
Log File
Directory
~/Desktop/GnssAnalysisFiles/demofiles/
Start UTC
yyyy mm dd hh mm ss.s
End UTC
yyyy mm dd hh mm ss.s
Iono
Tropo
Reference PVT
Stationary Receiver:
Lat (deg)
0.0000000
Lon (deg)
0.0000000
Alt (m)
0.00
Manual
WLS
NMEA
Moving Receiver:
NMEA File:
*.nmea, *.txt
All plots:
Menu Bars
Black
White
Close Plots
Status:
Clear Status Window
(c) 2017 Google. Version:
v2.6.3.0
All status messages are logged to file:
~/Desktop/GnssAnalysisFiles/statusLog.txt

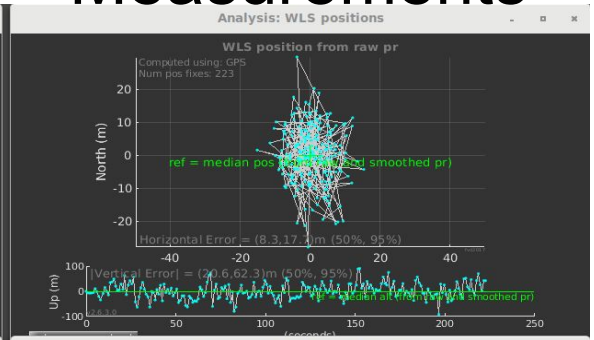

Select GnssLogger log file
Look In:
demofiles
gnss_log_2016_06_30_21_26_07.txt
File Name:
Files of Type:
(*.txt)
Open
Cancel

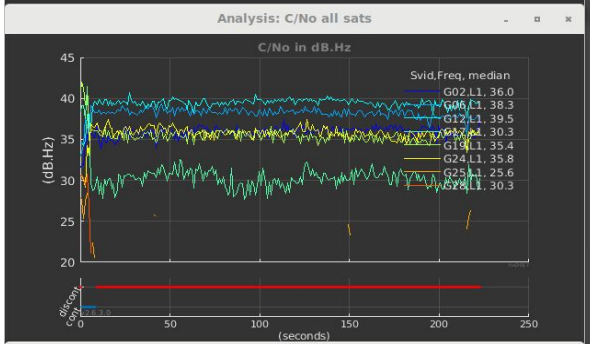

CONTROL PANEL
Android GNSS Analysis
v2.6.3.0
Analysis
Planning
Compare
About
Control
Find Log File
Analyze and Plot
Write Data to File
Make Report
Clear Settings ...
Analysis Plots
Refresh Plots
Tile Plots
SVIDs from measurements
GPS
GLO
GAL
BDS
QZS
GNSS Measurements
Log File
gnss_log_2016_06_30_21_26_07.txt
Directory
~/Desktop/GnssAnalysisFiles/demofiles/
Start UTC
2016 06 30 21 26 08.4
End UTC
2016 06 30 21 29 50.9
Iono
Tropo
Reference PVT
Stationary Receiver:
Lat (deg)
0.0000000
Lon (deg)
0.0000000
Alt (m)
0.00
Manual
WLS
NMEA
Moving Receiver:
NMEA File:
*.nmea, *.txt
All plots:
Menu Bars
Black
White
Close Plots
Status:
Clear Status Window
Start UTC, End UTC, Iono, Tropo, and [Reference PVT] panel read from ...
.../demofiles/gnss_log_2016_06_30_21_26_07-param.mat
to clear: click the [Clear Settings ...] button
Log file found, now click the [Analyze and Plot] button to process
(c) 2017 Google. Version:
v2.6.3.0
All status messages are logged to file:
~/Desktop/GnssAnalysisFiles/statusLog.txt

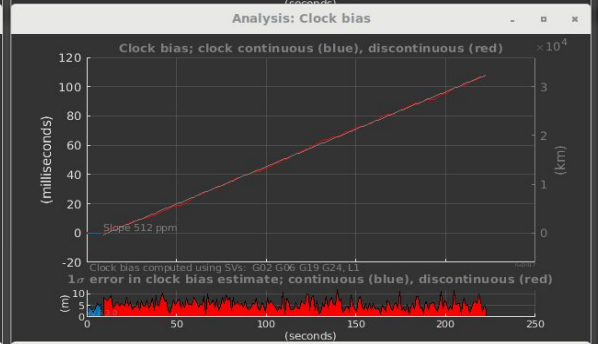
RF Clocks Measurements

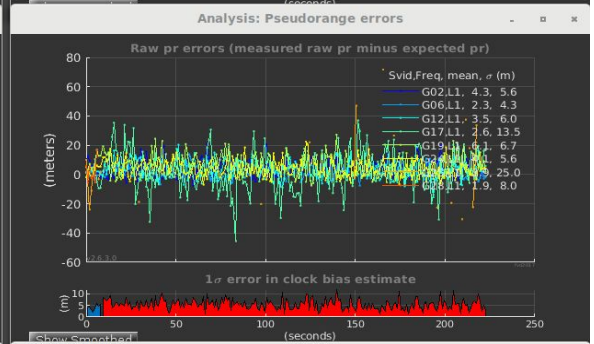


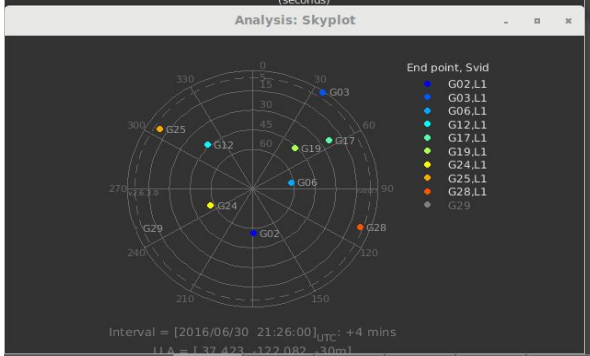


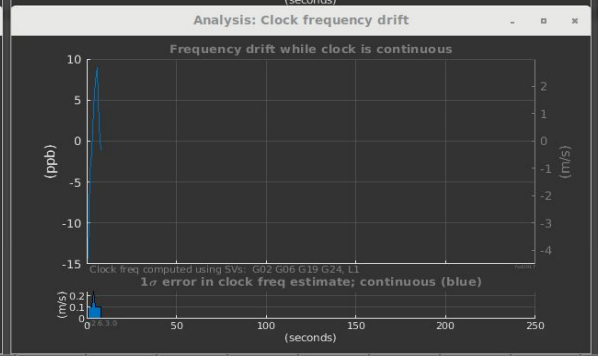


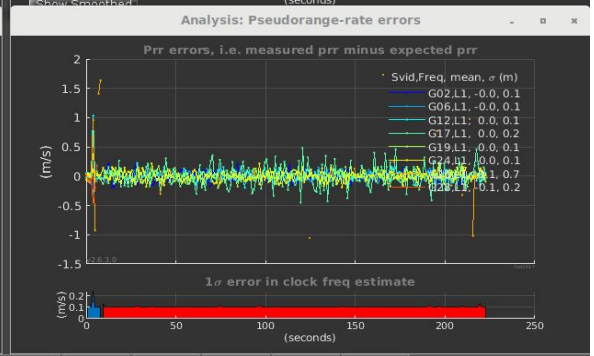


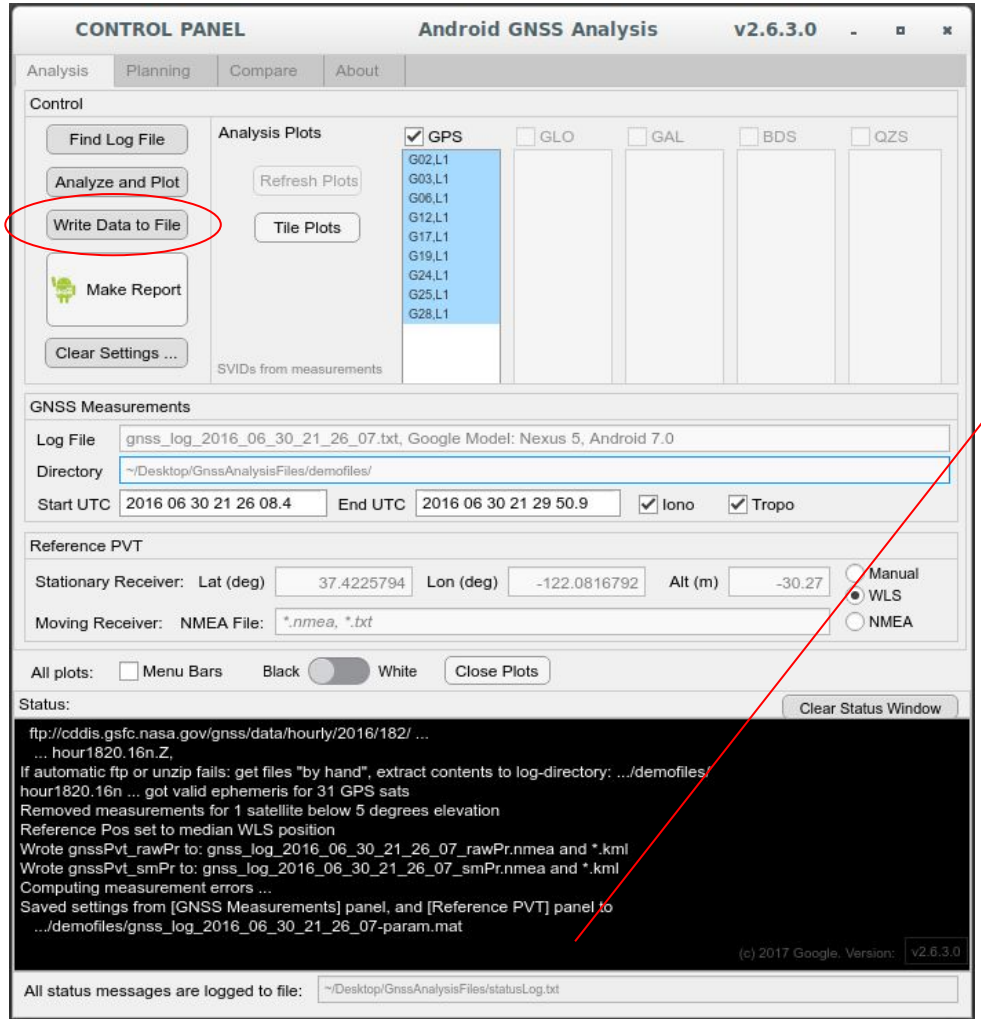




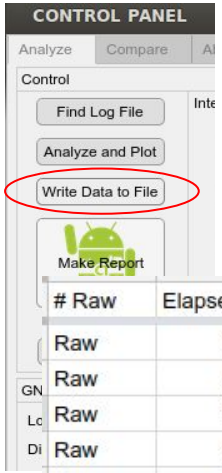









```
Writing derived data to a file ...
Wrote derived data to .../demofiles/gnss_log_2016_06_30_21_26_07.derived
```



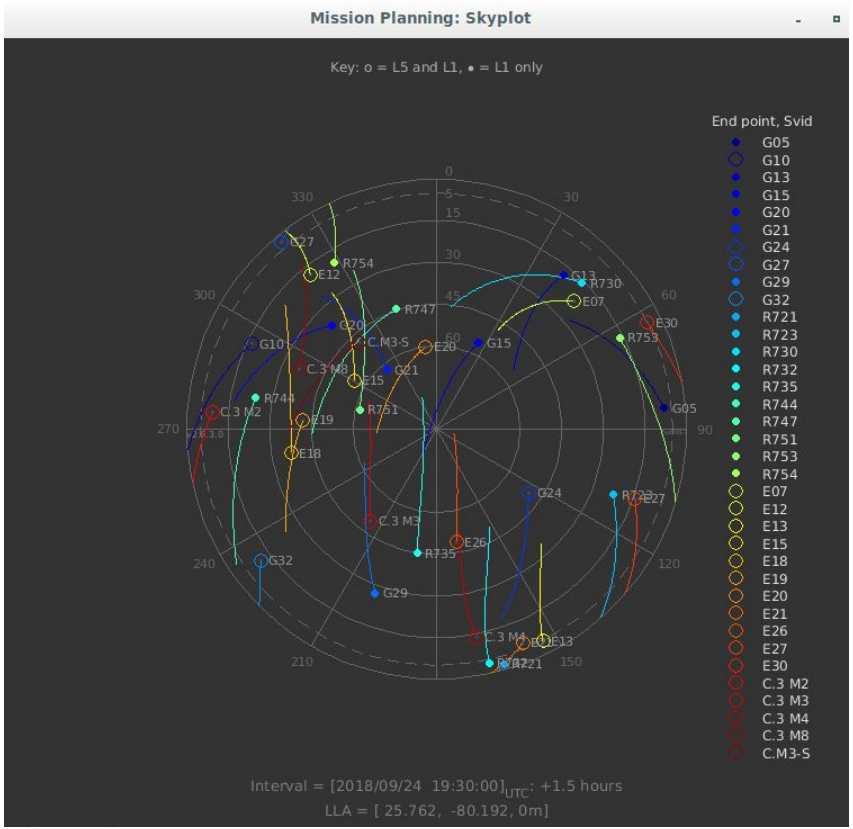
Log file of derived data

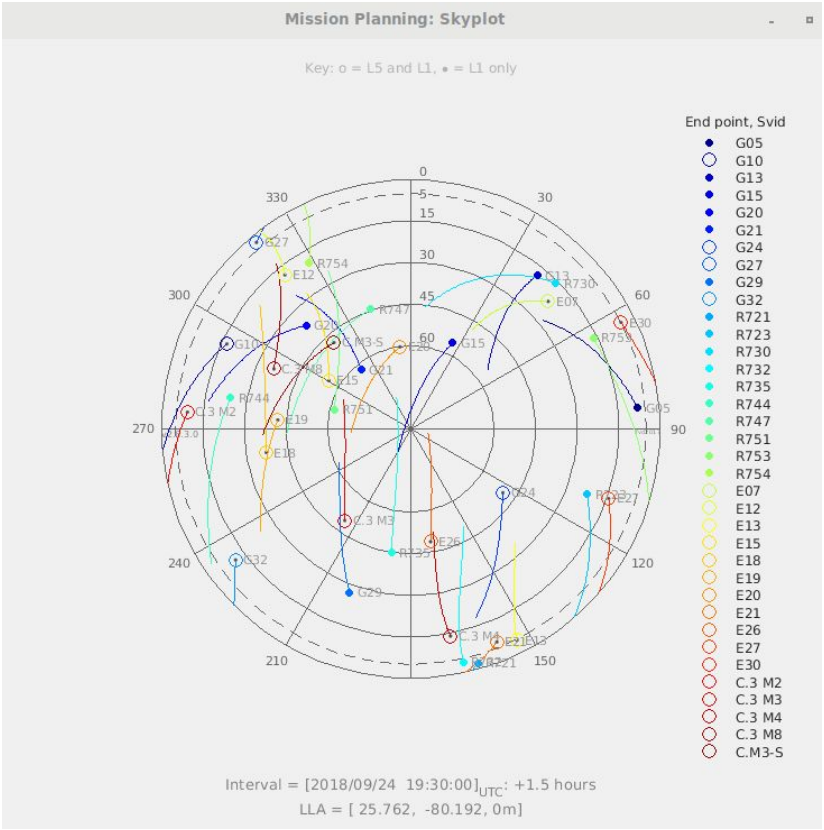
NOTE: Raw and Smoothed PR are in this log file

# Raw	ElapsedRealtime	TimeNanos	FullBiasNanos	BiasNanos	BiasUncertaintyNan	DriftNanosPerSe	DriftUncertaintyN	HardwareClockD	Svid	State	ReceivedSvTimeNanos
Raw	72066156	72077939000000	-1151285108458	0	29.04968824	-10.44367167	9.725224775	188	2	15	422786326364257
Raw	72066156	72077939000000	-1151285108458	0	29.04968824	-10.44367167	9.725224775	188	3	15	422786311362544
Raw	72066156	72077939000000	-1151285108458	0	29.04968824	-10.44367167	9.725224775	188	6	15	422786328163499
Raw	72066157	72077939000000	-1151285108458	0	29.04968824	-10.44367167	9.725224775	188	12	15	422786324938402
Raw	72066158	72077939000000	-1151285108458	0	29.04968824	-10.44367167	9.725224775	188	17	15	422786318854444
Raw	72066158	72077939000000	-1151285108458	0	29.04968824	-10.44367167	9.725224775	188	19	15	422786325655597
Raw	72066159	72077939000000	-1151285108458	0	29.04968824	-10.44367167	9.725224775	188	24	15	422786327049332
Raw	72066160	72077939000000	-1151285108458	0	29.04968824	-10.44367167	9.725224775	188	25	15	422786314218724
Raw	72066160	72077939000000	-1151285108458	0	29.04968824	-10.44367167	9.725224775	188	28	15	422786314963456

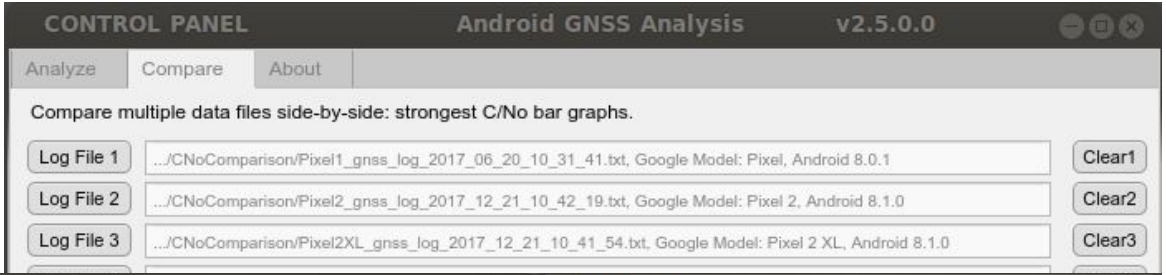
# MEAS	TimeNanos	Svid	CarrierFrequenc	Cn0DbHz	AzDeg	ElDeg	RawPrM	RawPrUncM	RawPrErrorM	SmPrM	SmPr
MEAS	72077939000000	6	1575420000	33.5	83.7	62.483	20690041.29	3.298	-3.098	20690038.62	
MEAS	72077939000000	12	1575420000	34.6	314.554	41.623	21656901.04	2.998	-2.457	21656898.63	
MEAS	72077939000000	17	1575420000	39.1	55.133	25.21	23480825.77	1.799	0.201	23480822.96	
MEAS	72077939000000	19	1575420000	42	43.12	48.227	21441891.39	1.199	-0.459	21441892.53	
MEAS	72077939000000	24	1575420000	30.4	250.632	57.707	21024060.15	4.197	-2.266	21024059.72	
MEAS	72077939000000	25	1575420000	27.5	303.286	7.635	24870579.66	5.696	-6.677	24870574.92	
MEAS	72077939000000	28	1575420000	30.6	109.618	8.509	24647314.63	4.197	-7.244	24647316.64	

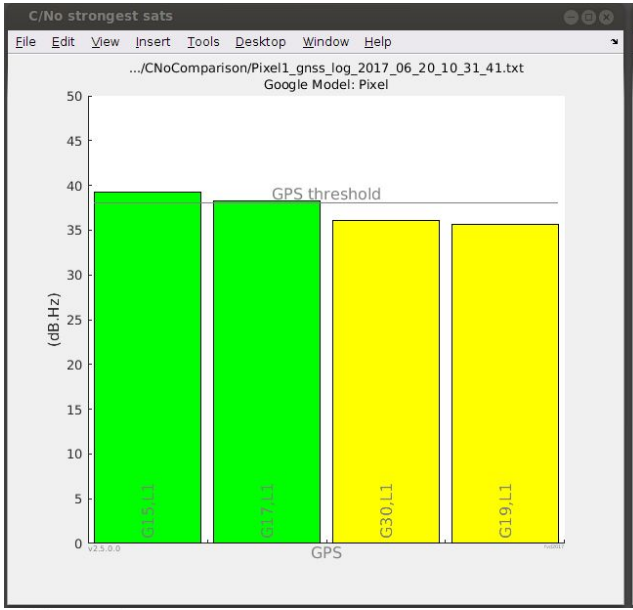
Other useful features of the tools: Mission Planner

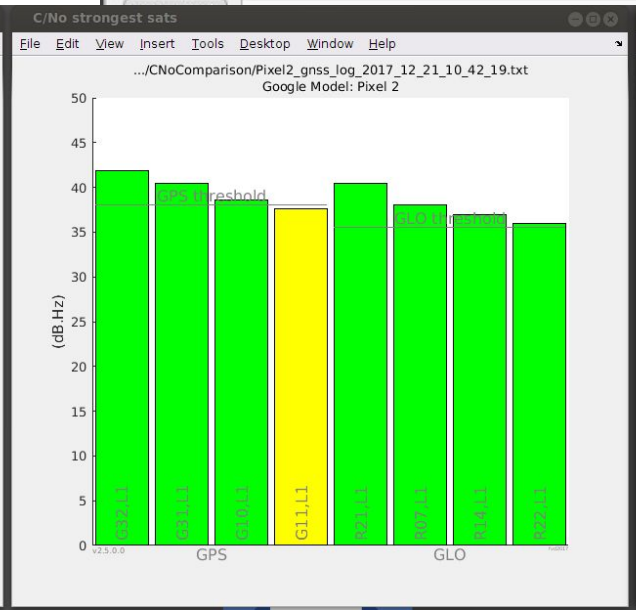


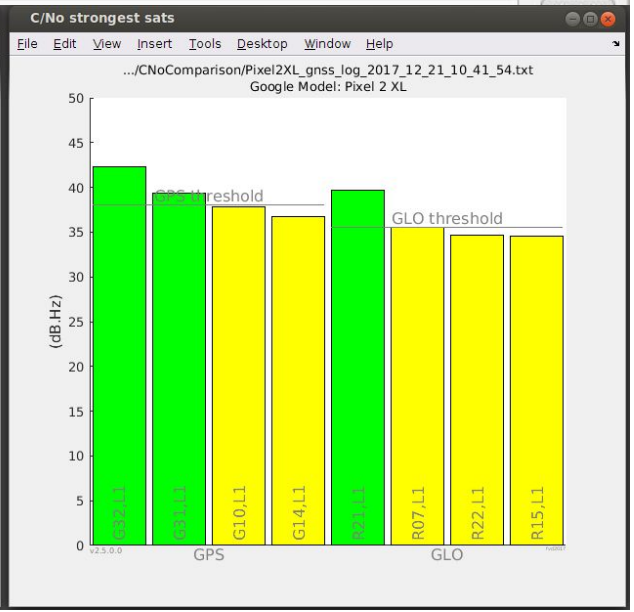


Other useful features of the tools: Receiver C/No comparison









Other useful features of the tools:

Illustrated by hands-on exercises ...

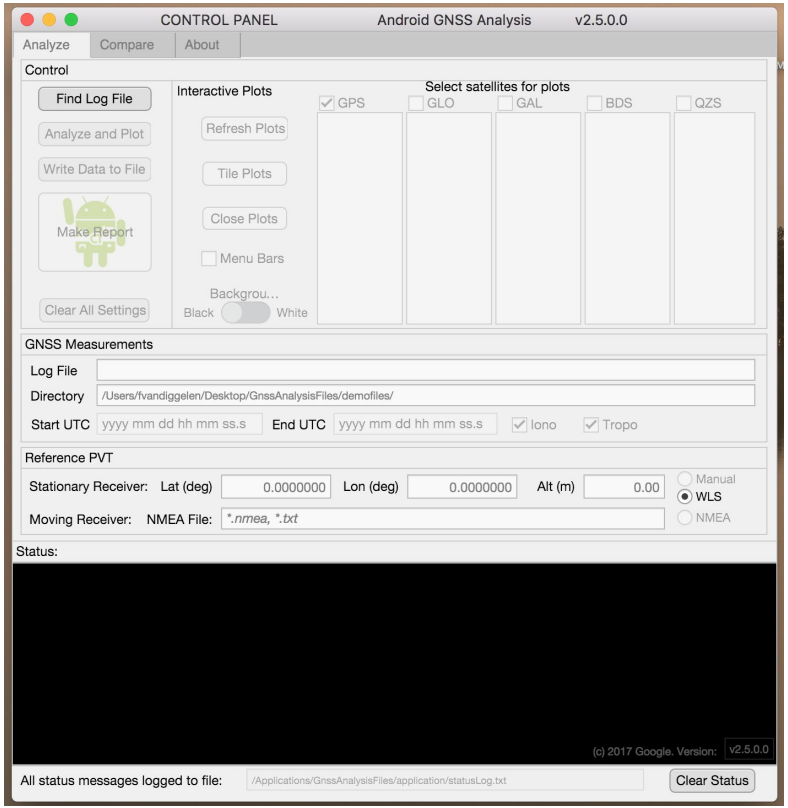
Overview

1. Raw GNSS Measurements
2. Logging Tools
3. How to get Raw Pseudorange
4. Analysis Tools (and smoothed pseudorange)
5. Hands-on Exercises

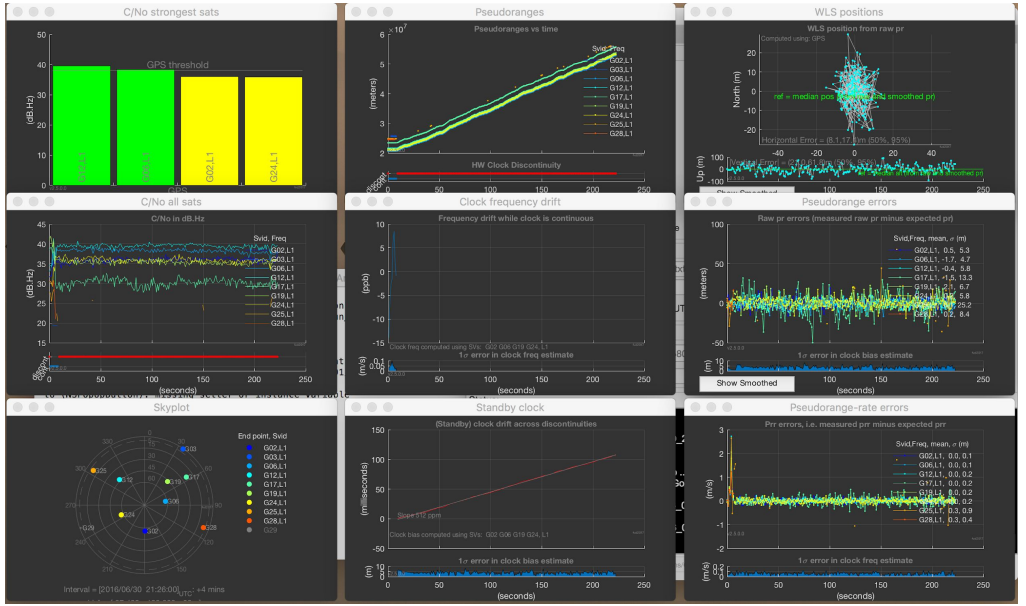
Hands-on exercises

1. .../GnssAnalysisFiles/demofiles/
   - The demo log file you downloaded with the desktop app
   - Learn the basic capabilities of the analysis tools
2. .../GnssAnalysisFiles/driving/
   - GPS dual-frequency log file with ground-truth nmea
   - Analyze reflections in urban canyons
3. .../GnssAnalysisFiles/ionotropodemo-2017/
   .../GnssAnalysisFiles/ionotropodemo-2022/
   - GNSS log file, stationary, at a known position, open sky
   - Analyze iono and tropo errors.

Exercise #1 .../GnssAnalysisFiles/demofiles/

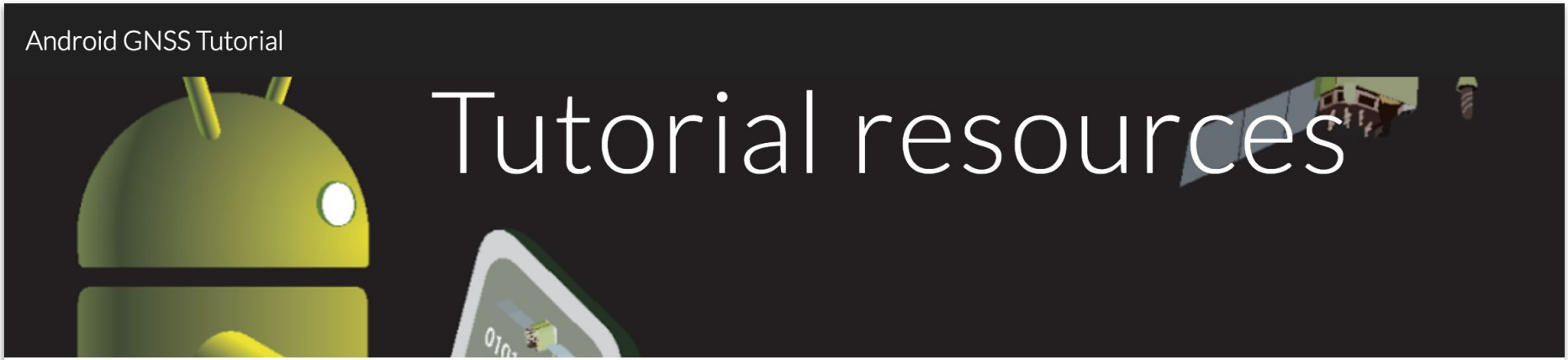


RF Clocks Measurements



Download log files for the following exercises

https://sites.google.com/view/gnsstutorial



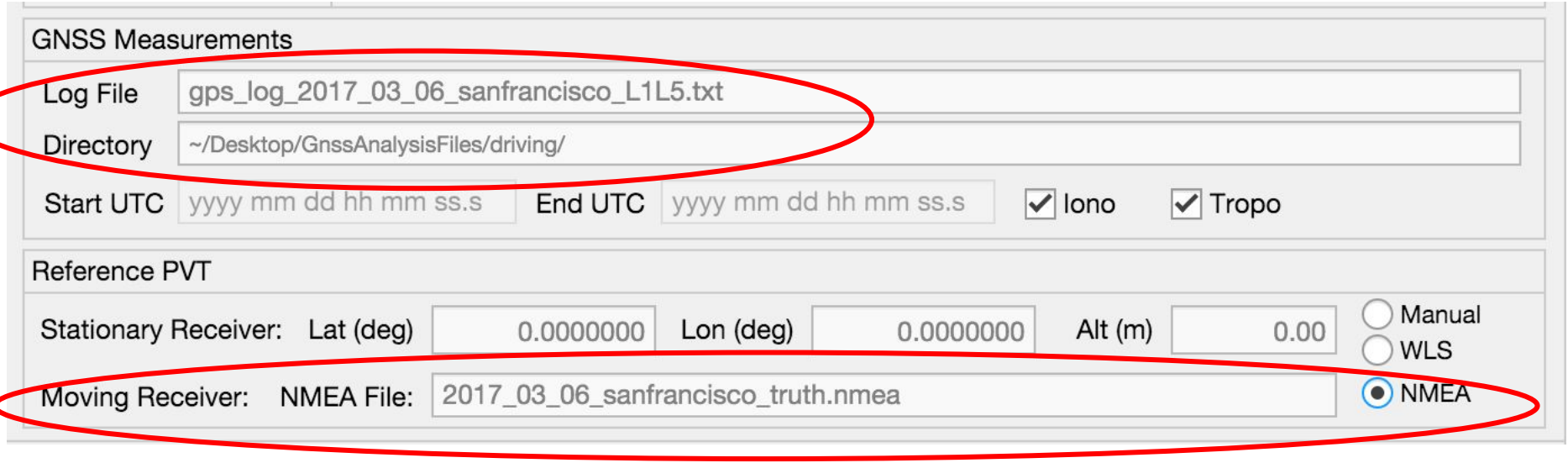
Sample log files to run with GnssAnalysisApp

These zip files have GnssLogger log files with ephemeris for you to process with the GnssAnalysisApp

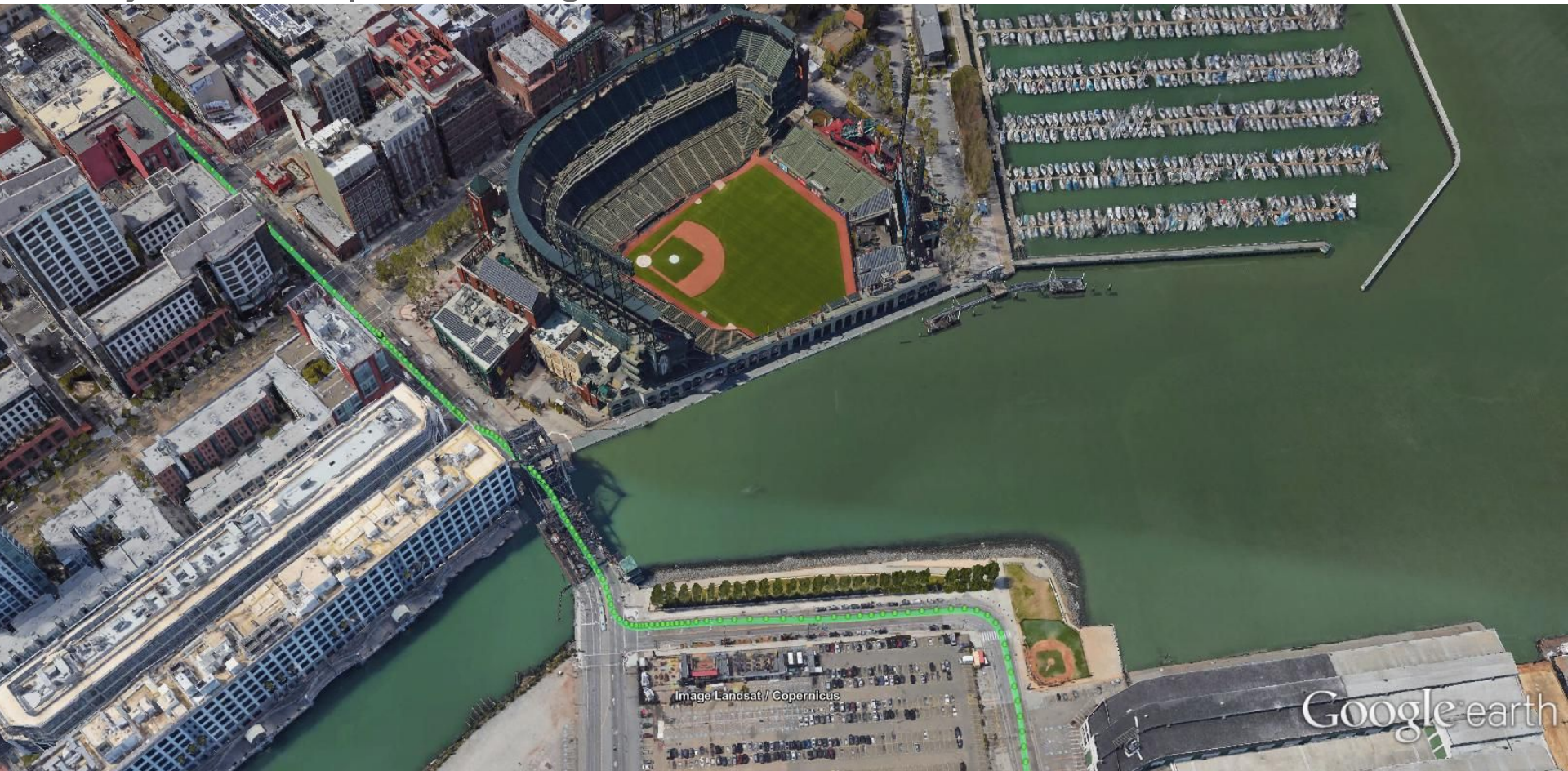
driving (log file, driving, GPS, L1L5, with truth nmea)

ionotropodemo (two log files, GNSS and GPS-only, stationary with true position in readme.txt)

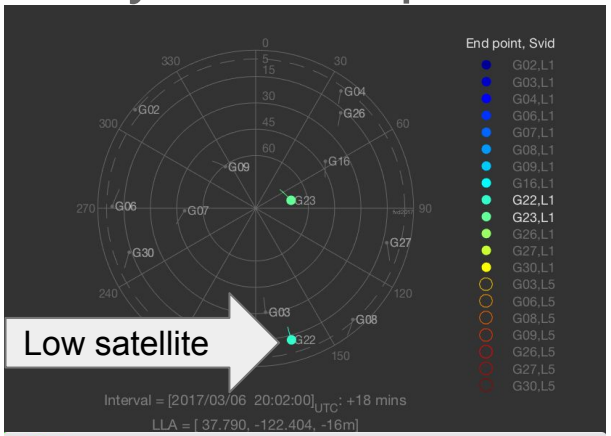
Exercise #2 .../GnssAnalysisFiles/driving/

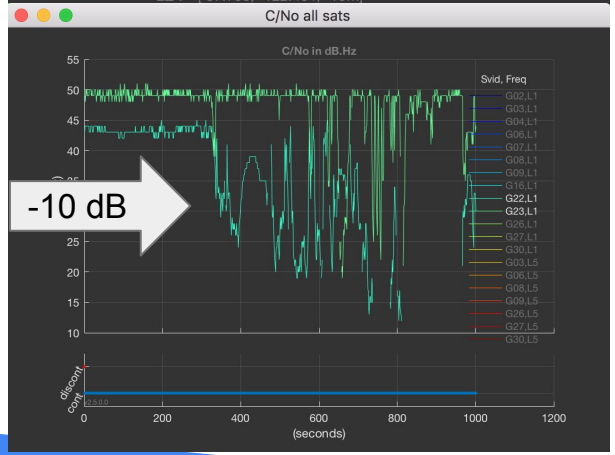


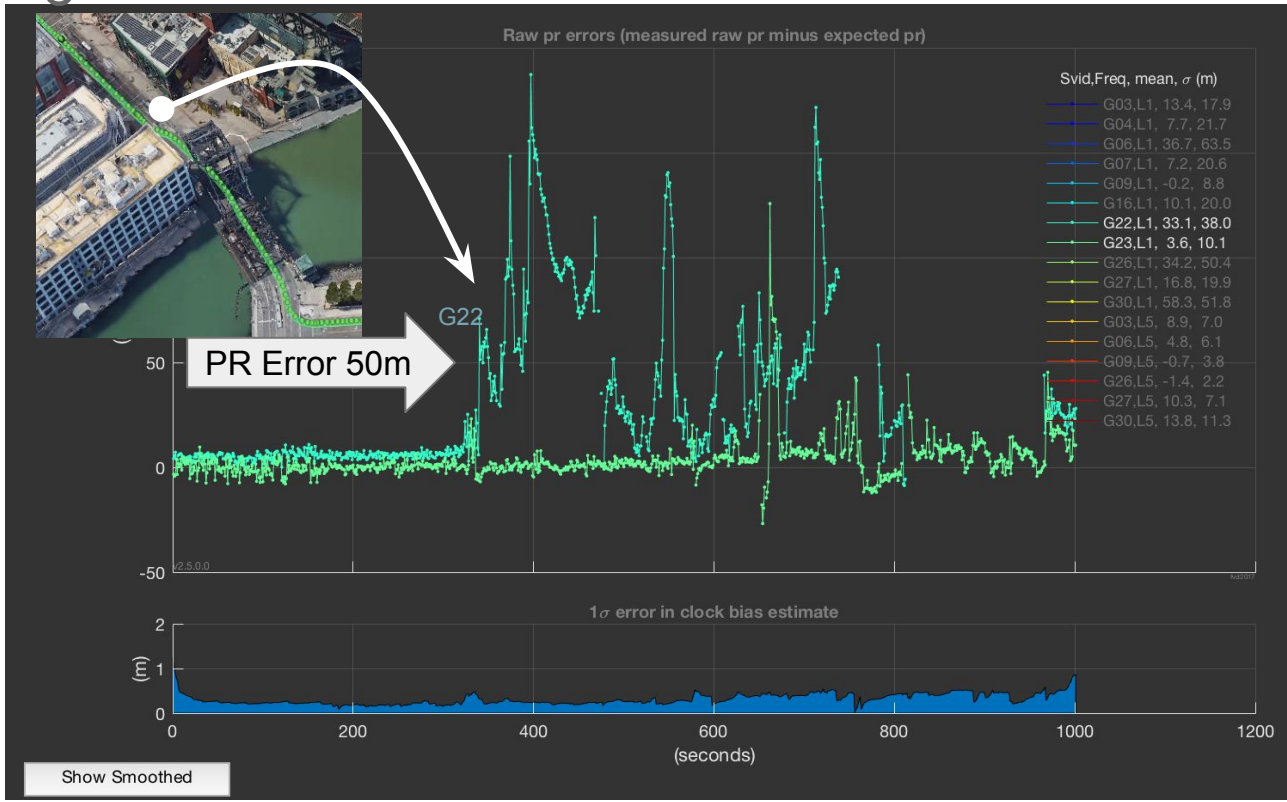
Analysis example, driving into San Francisco:



Analysis example, driving into San Francisco:







What happened with satellite G22?

Exercise #3 .../GnssAnalysisFiles/ionotropodemo-2017/

1. Use true position for Reference PVT
2. Select highest satellites to use for clock bias computation
   CustomParam.txt
3. Remove iono and tropo model from analysis

Then error plot will show all errors relative to the highest satellites.

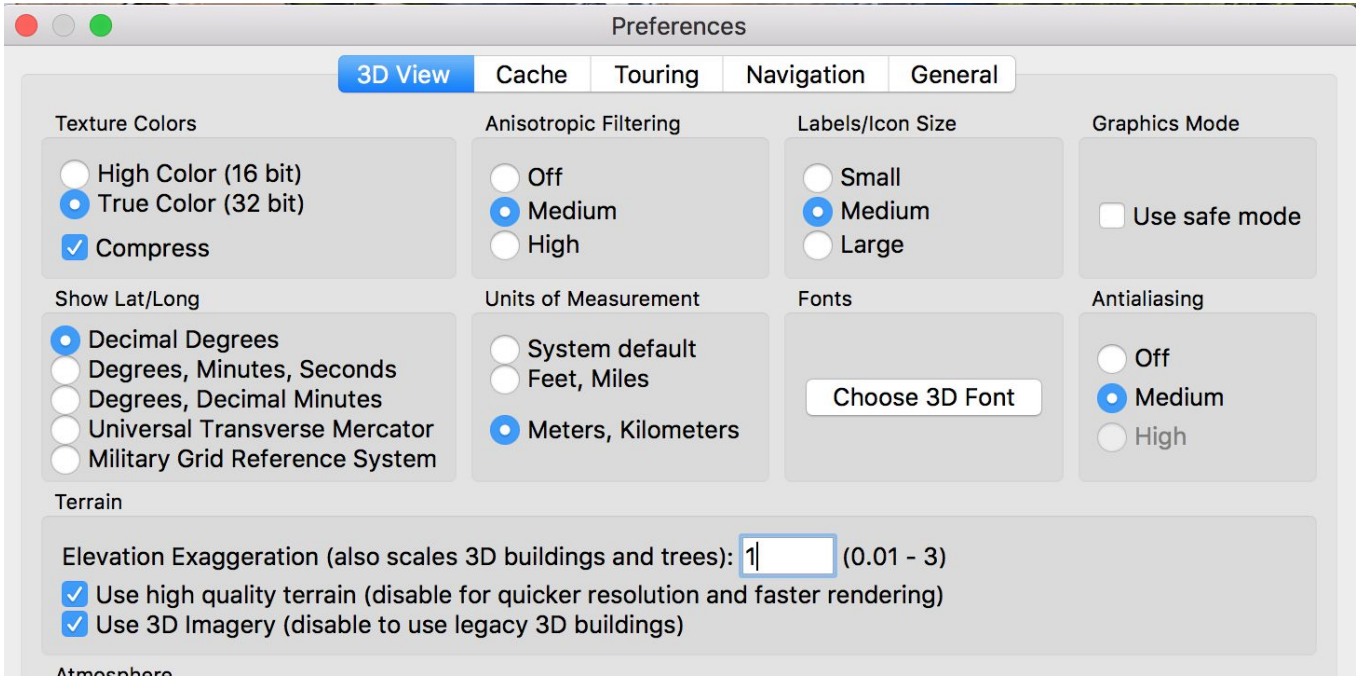
How to get true position from Google Earth (1)

Preferences …

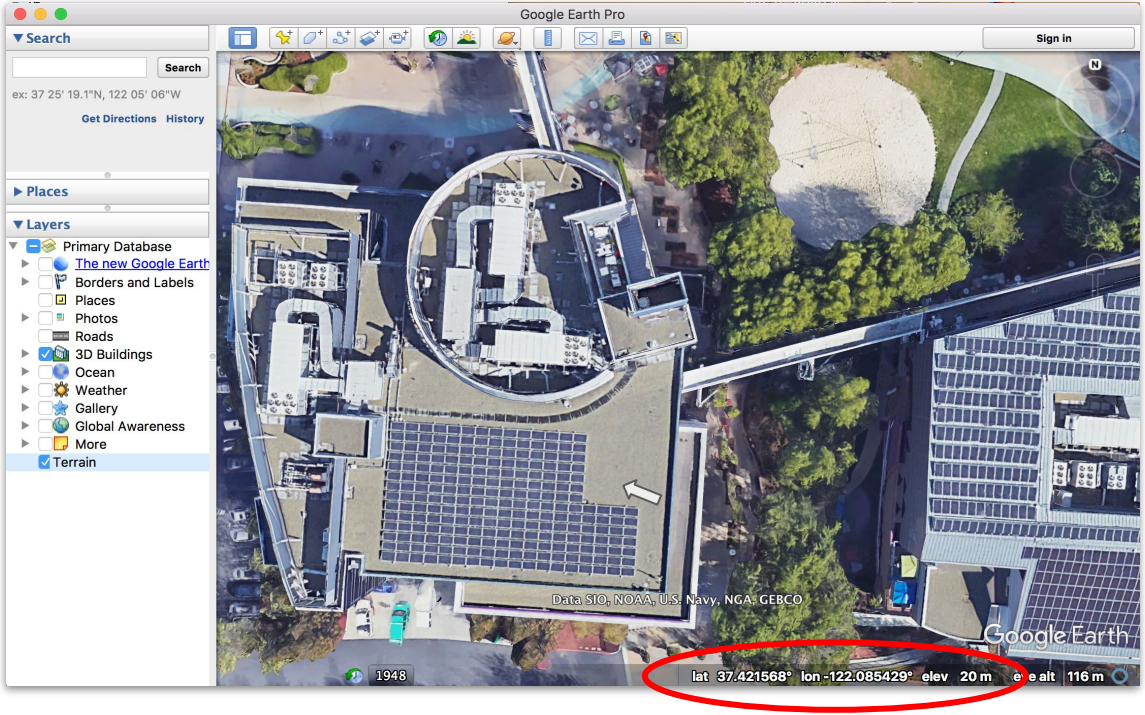
Show Lat/Long
  Decimal Degrees

Units of Measurement
  Meters,Kilometers

Terrain
  Use high quality terrain
  Use 3D imagery



How to get true position from Google Earth (2)



hG = height above Geoid,
from Google Earth 3D Buildings, 20m
hS = height of stand = 1m
dE = -32, Ellipsoid - Geoid

hE = hG+hS+dE = 20+1-32 = -11 m.

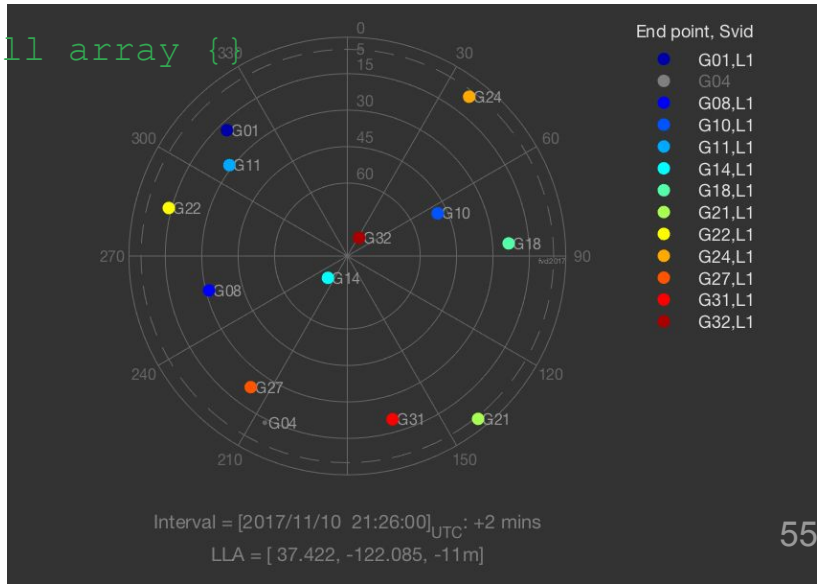
Rooftop true position: 37.421568, -122.085429, -11m

CustomParam.txt

```
%Currently supported:
%param.losSvid = list of svid to use for computing clock (Bc and BcDot)
%template for losSvid.Svid: must have .FreqBand, .Constellation, .Id
GpsL1Svid.Id=0;
GpsL1Svid.Constellation=GnssConstants.GNSS_CONSTELLATION_GPS;
GpsL1Svid.FreqBand=GnssConstants.L1_BAND;  %generic GPS L1 struct
Svid(1)=GpsL1Svid; Svid(1).Id = 32;
%Svid(2)=GpsL1Svid; Svid(2).Id = 14;
param.losSvid.Svids = {Svid};  %pack in a cell array {33}
```

You edit these lines to choose the reference satellite(s) you want.

And place this txt file in the same directory as your log file.



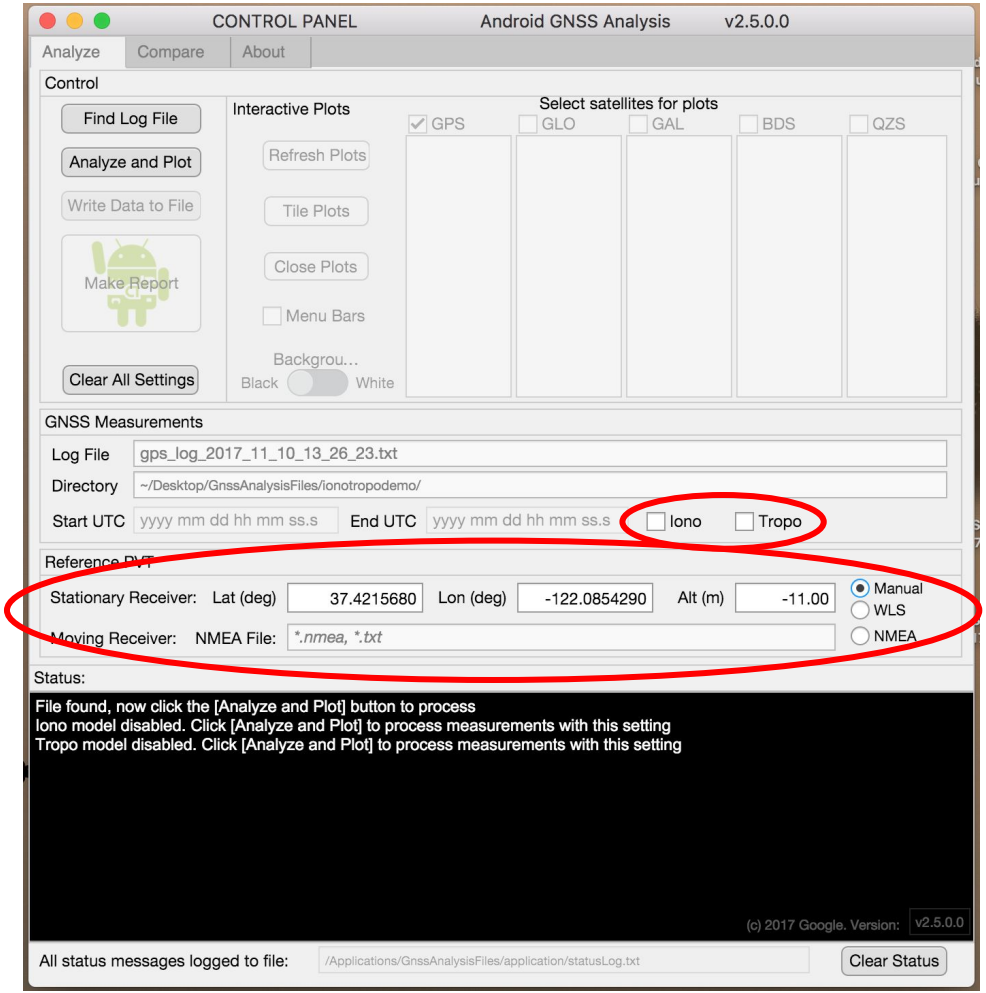
Analyzing, errors:

iono +

tropo +

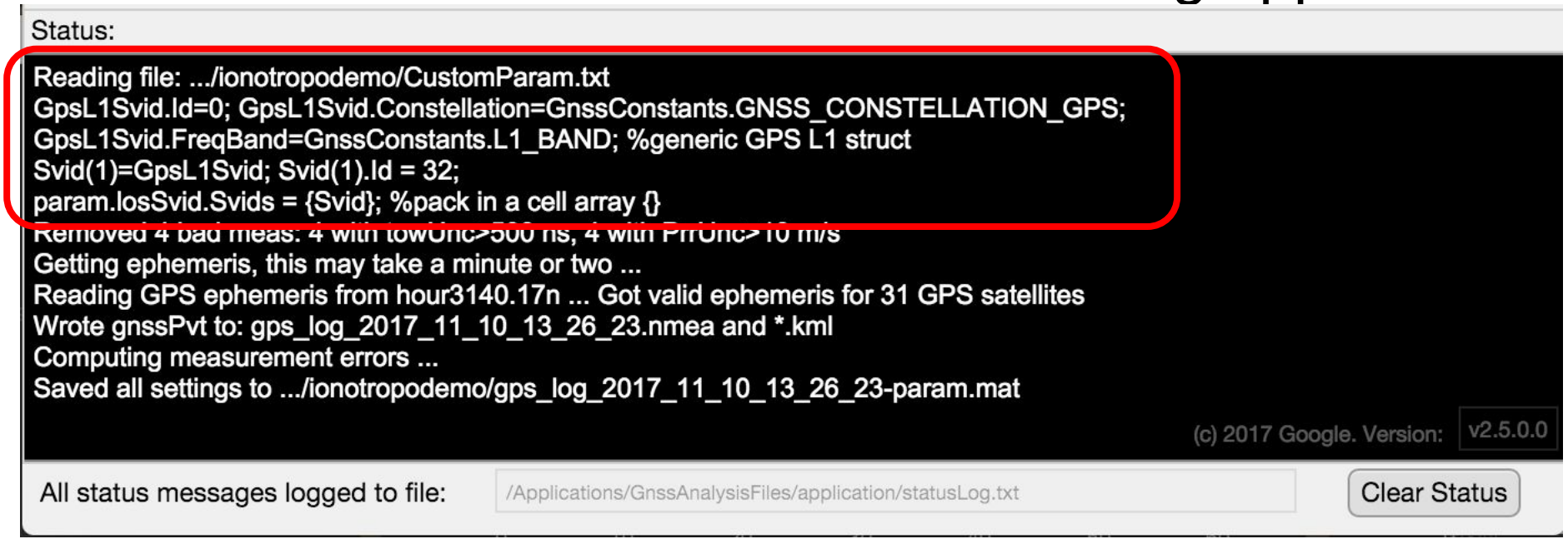
SIS[1]

In this exercise we will see the measured values of iono and tropo errors, by adding and removing the modeled values.

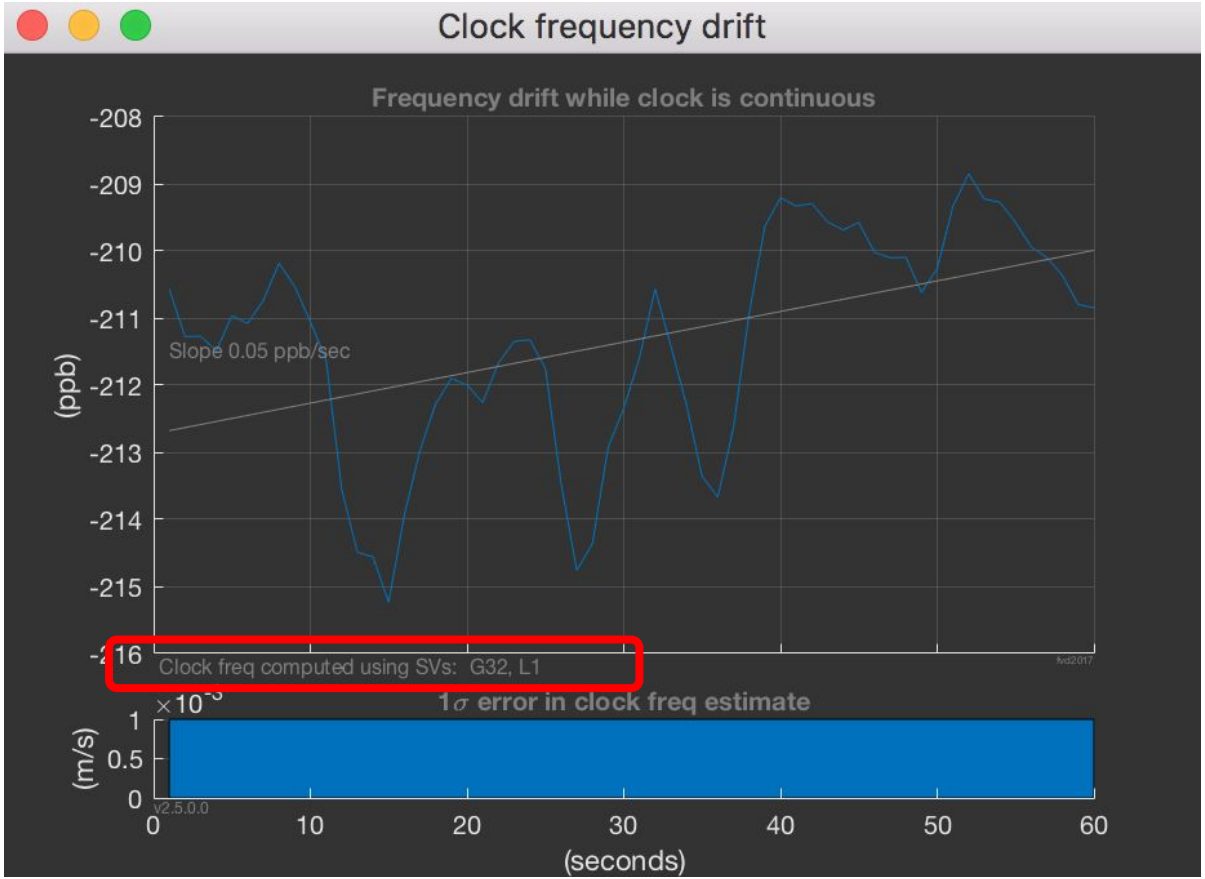


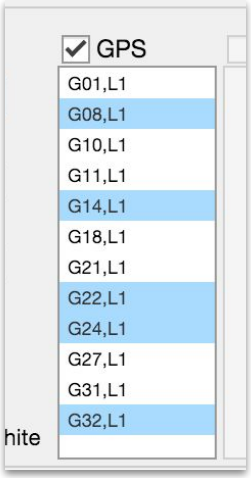
[1]SIS errors = Signal In Space errors = errors after applying broadcast ephemeris and clock values

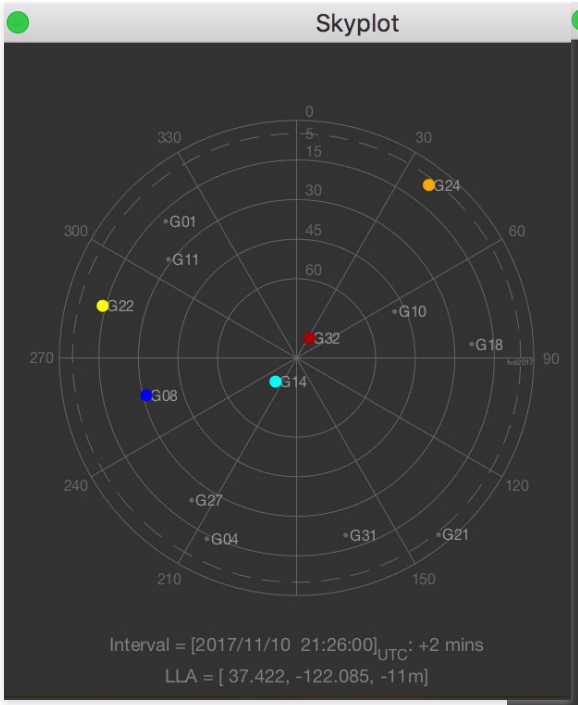
Notice the CustomParam.txt values being applied:

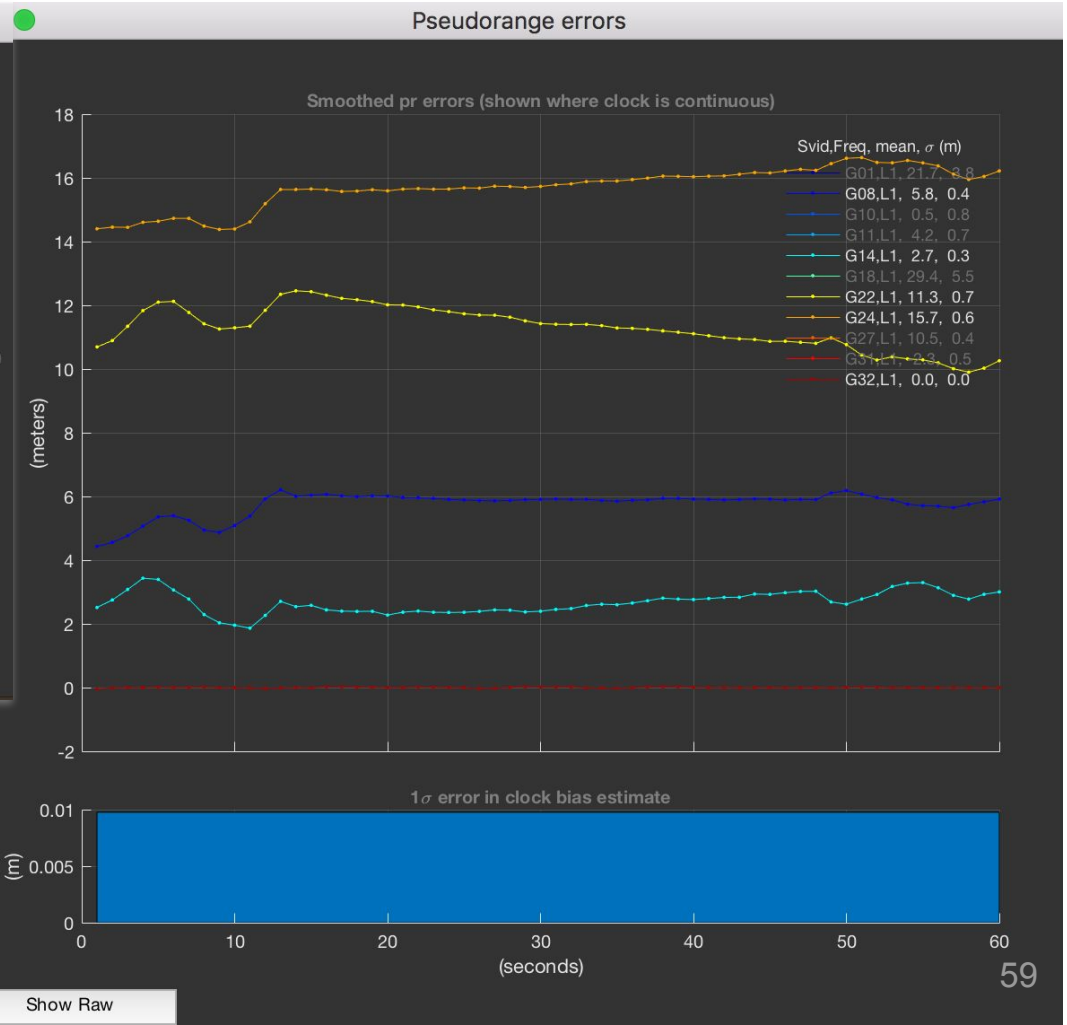


First we select GPS satellites only, using the log file: gps_log_2017_11_10_13_26_23.txt
Then we use satellite ID 32 to estimate the clock.
This will give us all iono and tropo errors *relative* to satellite 32.
For location computation, the *relative* iono and tropo errors are all that matters, because the GNSS 4d nav solution always removes the common error.

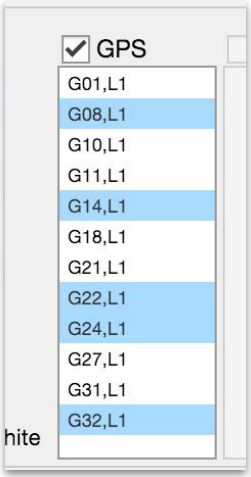

Clock frequency drift
Frequency drift while clock is continuous
-208
-209
-210
-211
-212
-213
-214
-215
(ppb)
Slope 0.05 ppb/sec
Clock freq computed using SVs: G32, L1
1σ error in clock freq estimate
(m/s)
1
0.5
0
v2.5.0.0
0
10
20
30
40
50
60
(seconds)

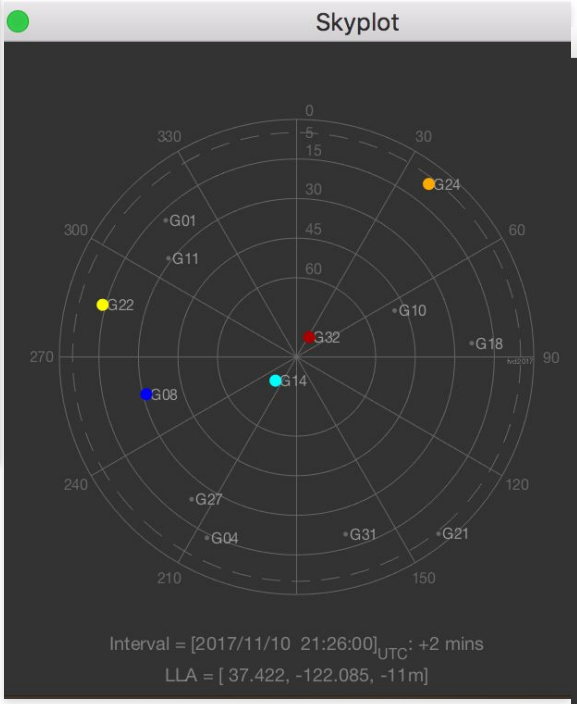


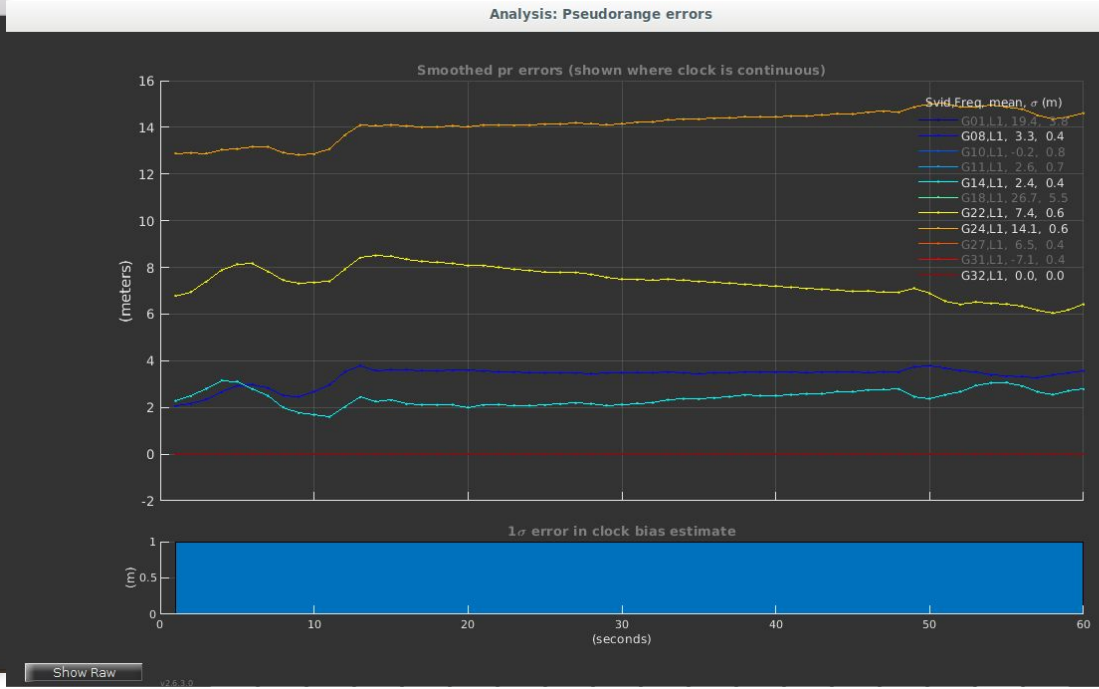




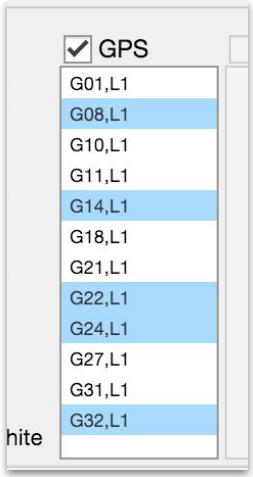
Iono & Tropo Errors

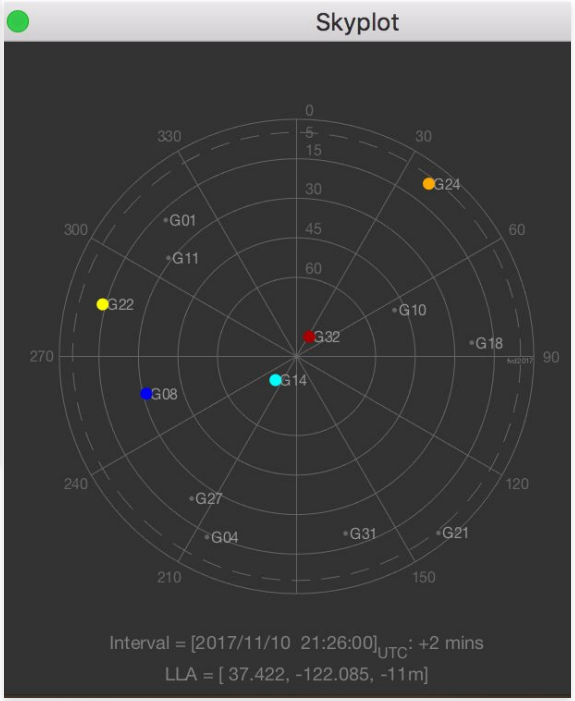


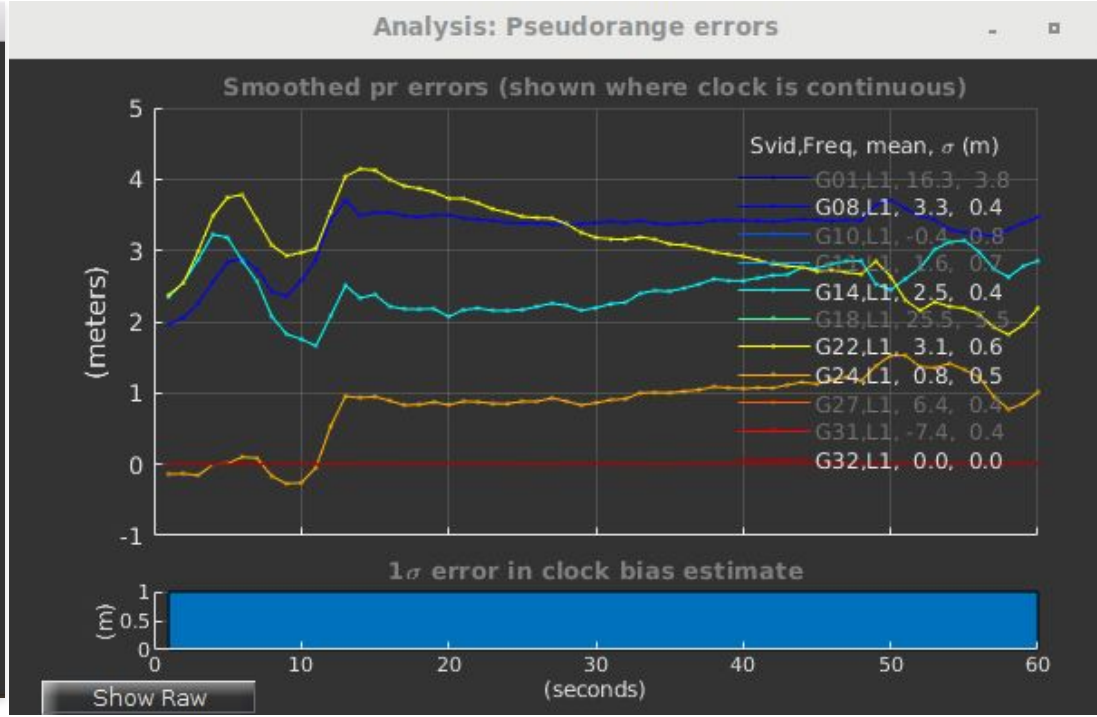




Tropo Error







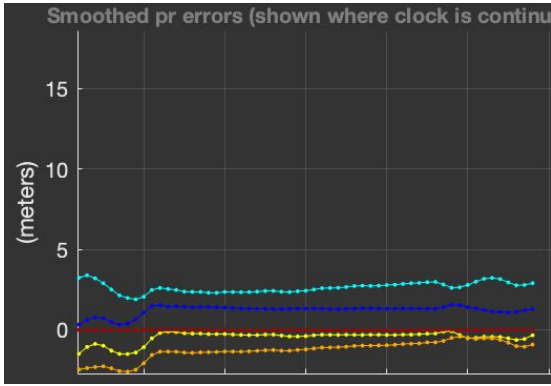
Iono Error

Iono and Tropo Errors, and trend with elevation above horizon

Sat	El	Key
G24	9°	
G22	13°	
G08	32°	
G14	80°	
G32	85°	

Tropo error, Iono modeled

Iono error, Tropo modeled

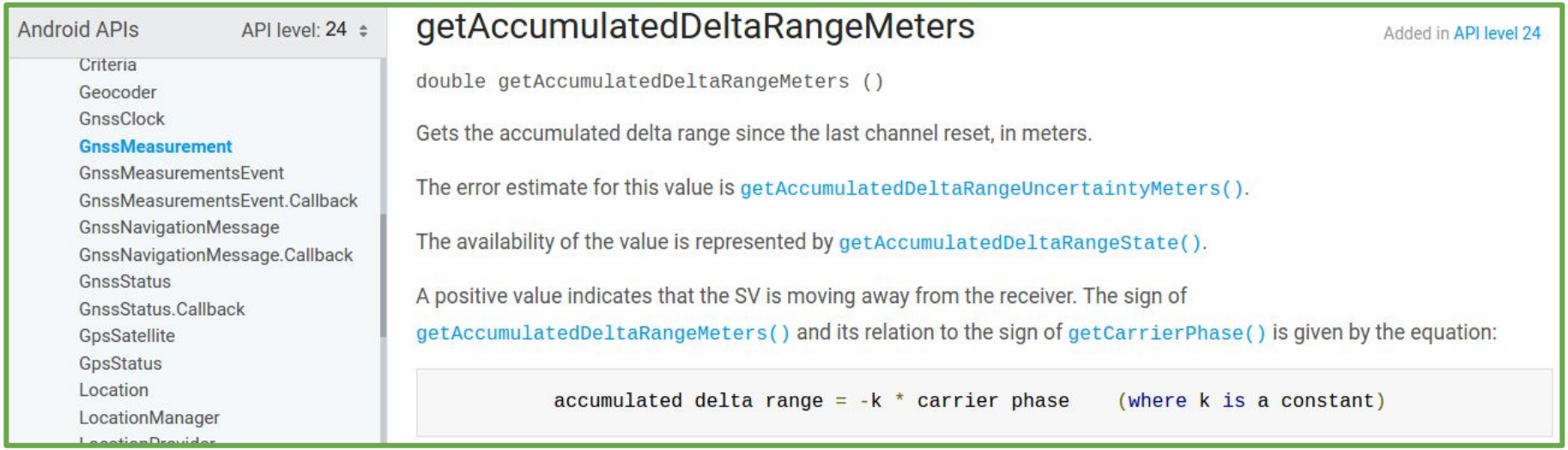


Iono & Tropo modeled

L1, L5 and Carrier Phase

Google

Carrier phase = AccumulatedDeltaRange



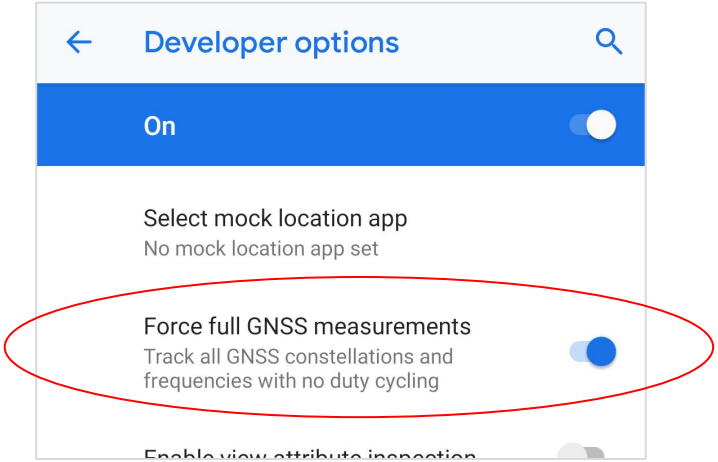
ADR is continuous only when clock is continuous, and there is no duty cycling

Carrier-phase PVT

Enable / Disable Duty Cycling:

In Android P, Google added a Developer option to enable or disable GNSS Duty Cycling

- When selected: The GNSS chipset will not duty cycle and will run at full power - keeping a continuous clock so one can receive continuous carrier phase measurements.
- Look for more information at g.co/GnssTools



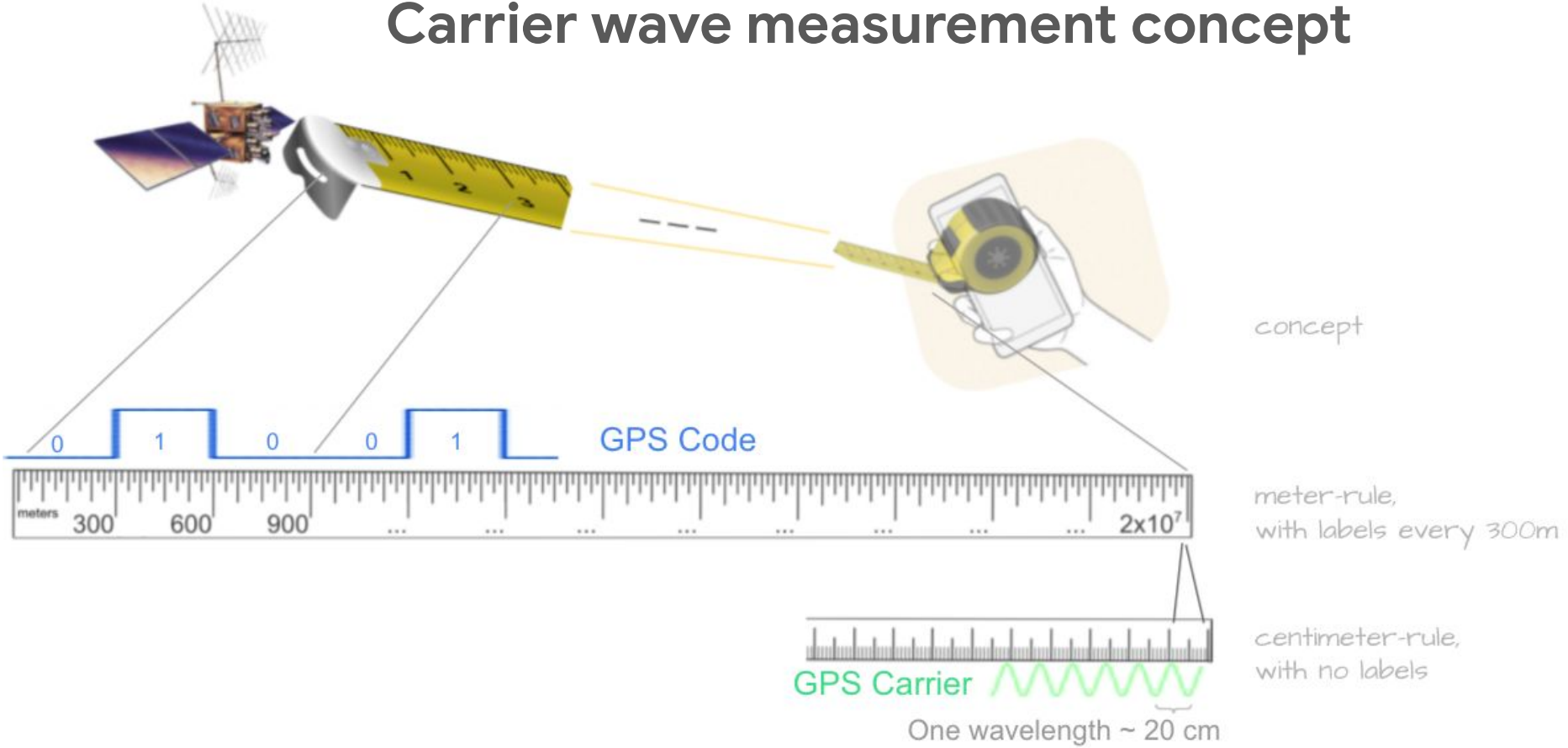
Becoming an Android Developer:

1. Go to the settings menu, and scroll to, or search for "About phone." Tap it.
2. Scroll to the bottom, where you see "Build number."
3. Tap it seven (7) times

Also: some phones disable duty cycling automatically when you request raw measurements

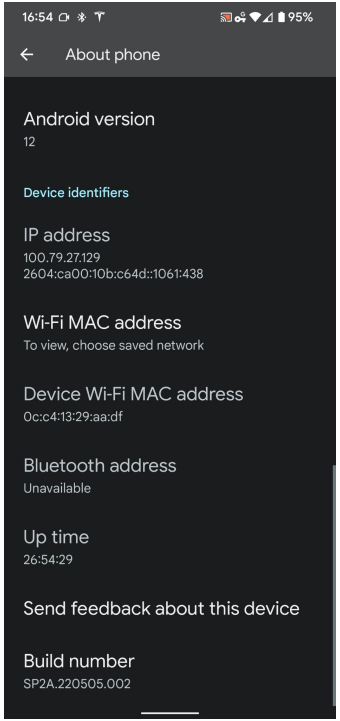

Google

Carrier wave measurement concept



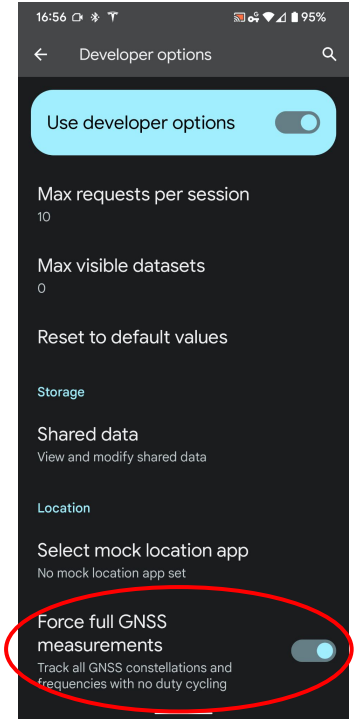
Duty cycling, and carrier phase

Phones duty cycle GNSS to save power, e.g. 1Hz = 0.2s on, 0.8s off, repeat.
Disable the duty cycling to get continuous carrier phase.



←
Enable developer options:

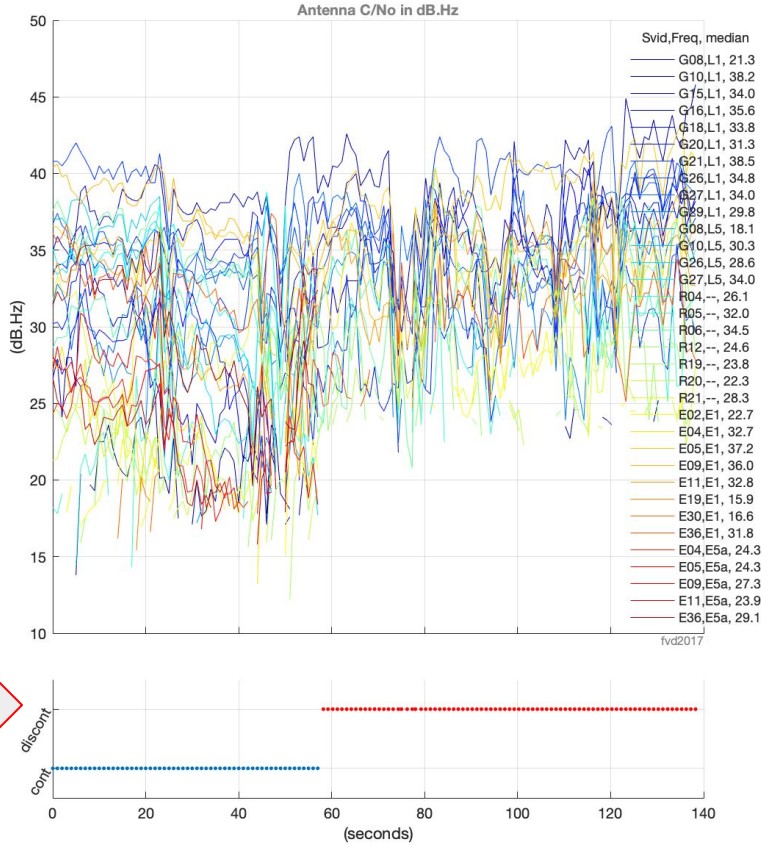
1. Settings,
2. About phone,
3. Tap Build number seven times.



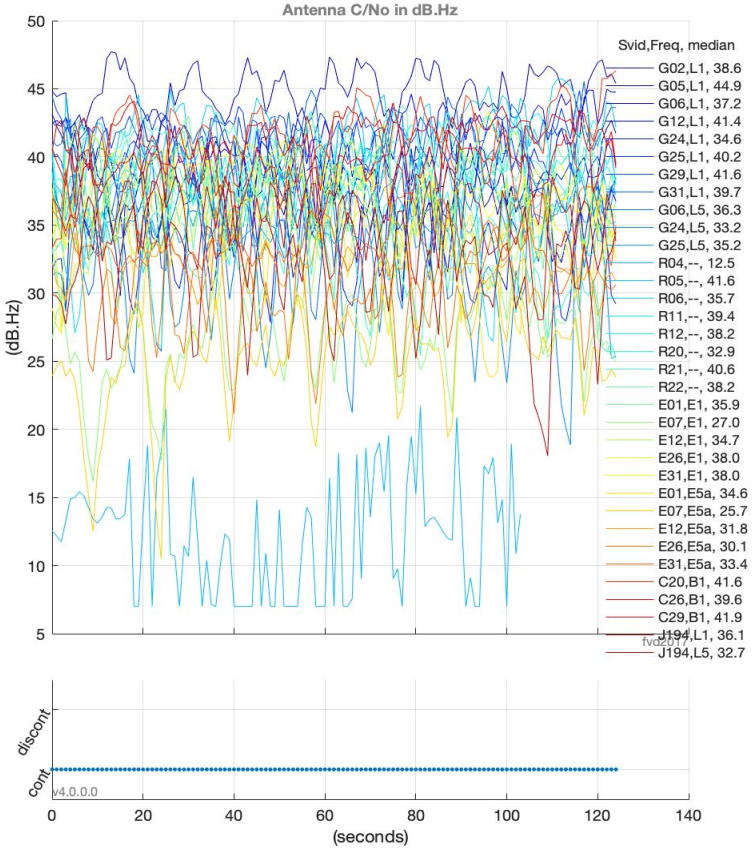
←
Disable duty cycling:

1. Settings,
2. Developer options,
3. Force full GNSS measurements

Checking duty cycling:



Duty cycling



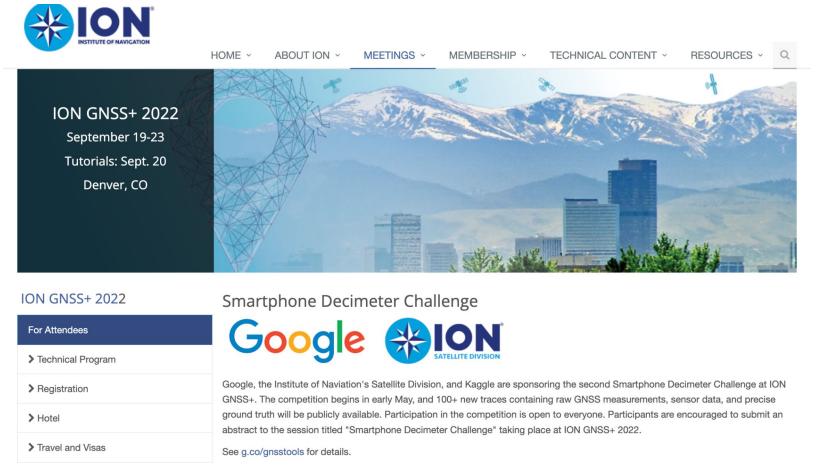
Smartphone Decimeter Challenge, competition

$10,000 cash prizes + free travel, registration, & speaking slots at ION GNSS conference, Sep 2022

- **SDC 2021**
  - 810 teams from 63 countries
  - 121 traces released
  - Phones: Pixels 4/5, Xiaomi Mi8, Samsung S20
    - Features: L1+L5, Carrier Phase ADR.

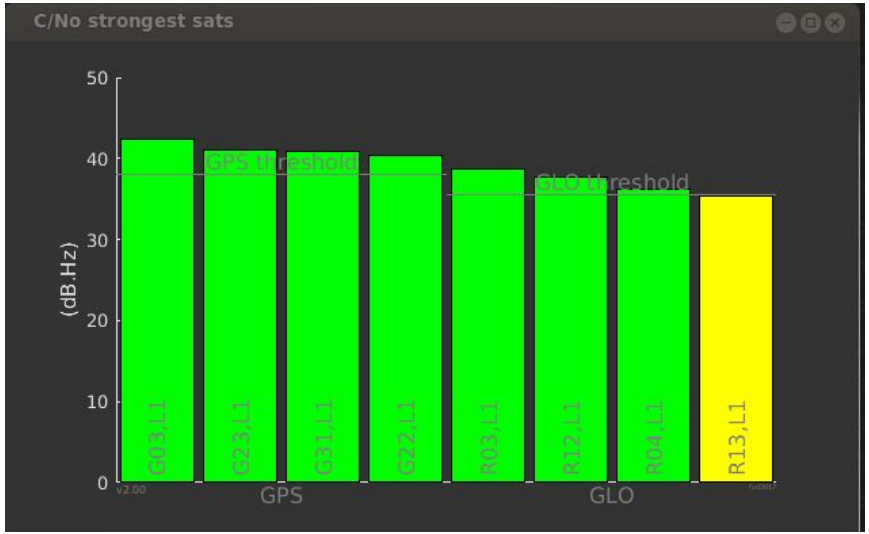
- **SDC 2022**
  - **Time**: first week of May - end of July
  - ~300 traces in US Bay area and Los Angeles
  - New phones included: Pixel 6, Samsung S21.
    - Features: L1+L5, Carrier Phase ADR

https://www.ion.org/gnss/googlecompetition.cfm

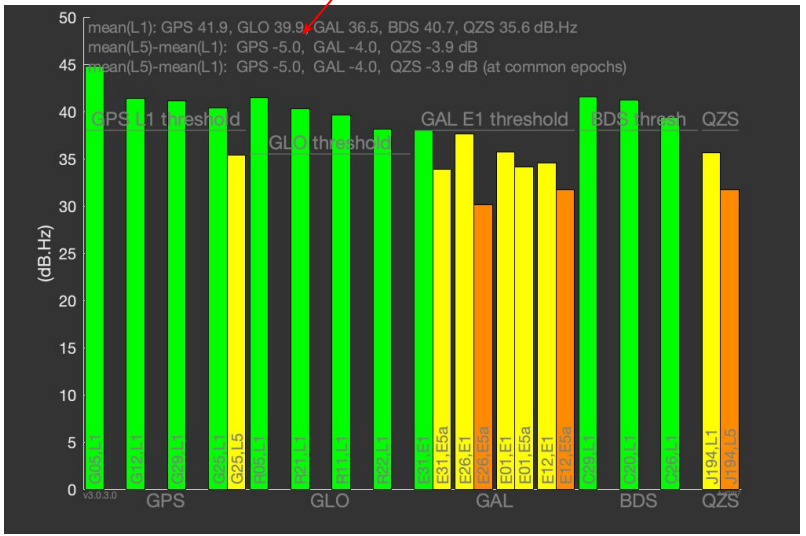


L1L5 C/No Comparison

L1 Only



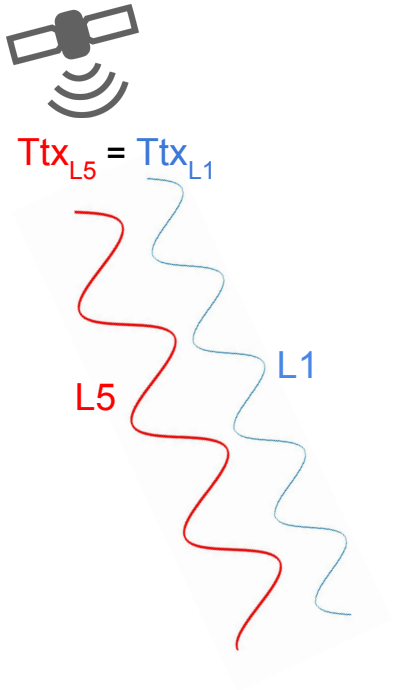
Antenna+filter loss at L5



Minimum C/No with 0 dBi RHCP antenna with 2dB Front End Noise Figure		
	GPS (dB.Hz)	GAL (dB.Hz)
L1/E1	45.5	47
L5/E5a (I + Q)	49	49
L5/E5a (I only)	46	46

From ICDs for GPS and Galileo

Group Delay

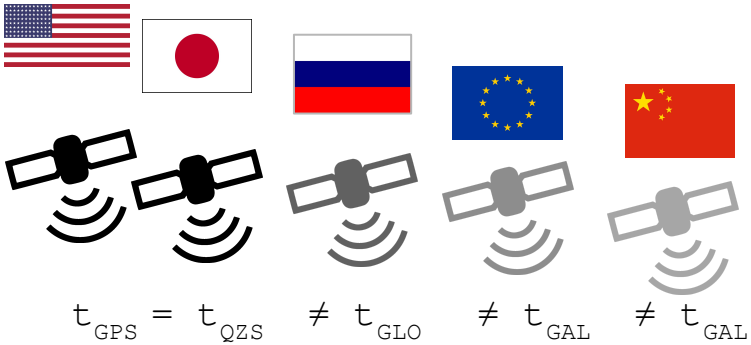


When two signals of different frequency pass through a medium (including: Earth's atmosphere, and the RF chain of the receiver), they experience different time delays relative to each other.

Also, the signal processing in different GNSS chips might result in significant differences in L1 v L5 delays, and this must be measured and compensated for before using L1 and L5 measurements together.

Observed Ttx_{L5} ≠ Observed Ttx_{L1}
i.e. $Pseudorange_{L5}$ ≠ $Pseudorange_{L1}$

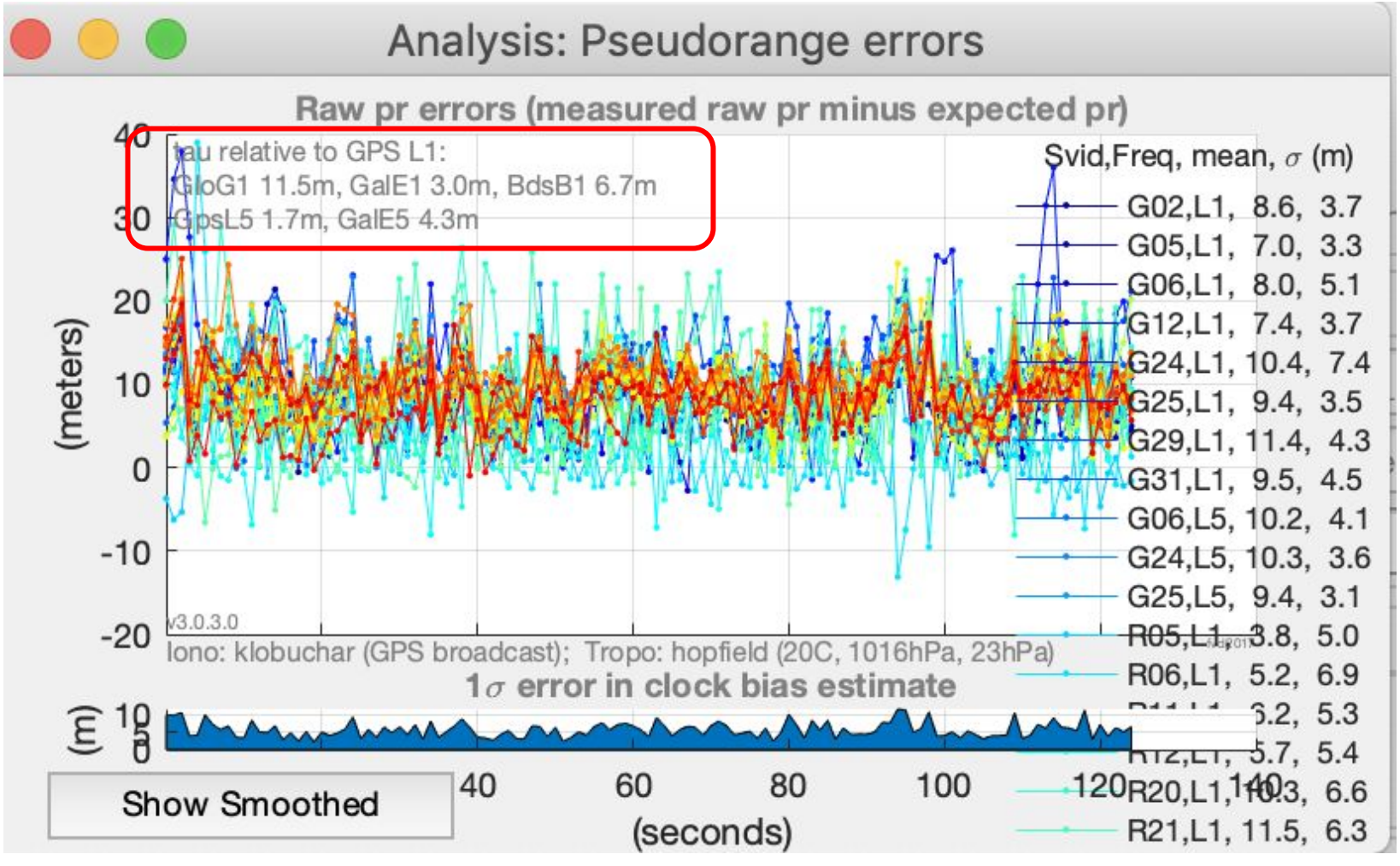
Inter System Bias



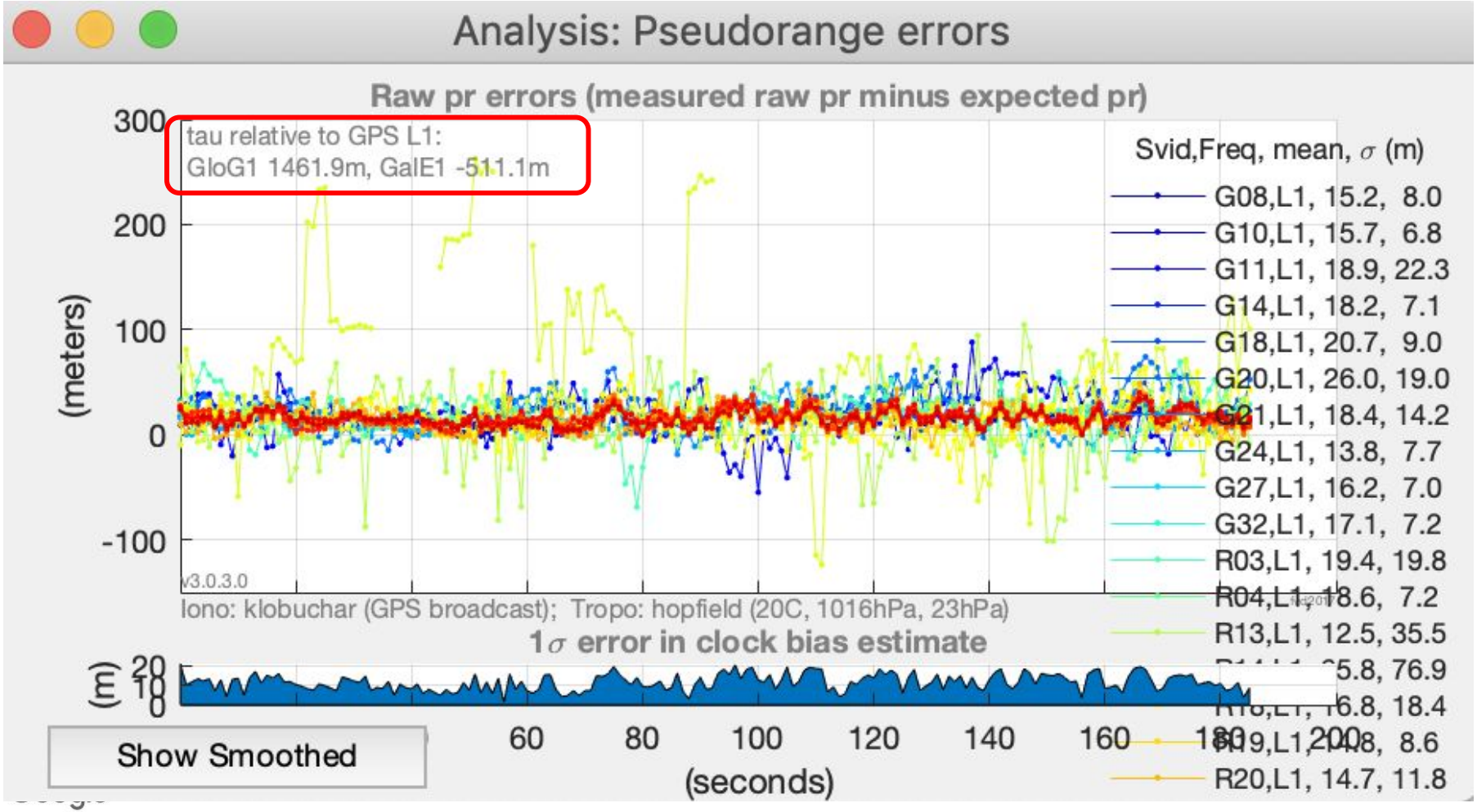
The different GNSSes are synced to different times (except QZS which manages to be synced to GPS)

Moreover, different GPS receivers will treat different signal types differently, resulting in further differences in observed times. These offsets be measured and accounted for before combining pseudoranges from different systems.

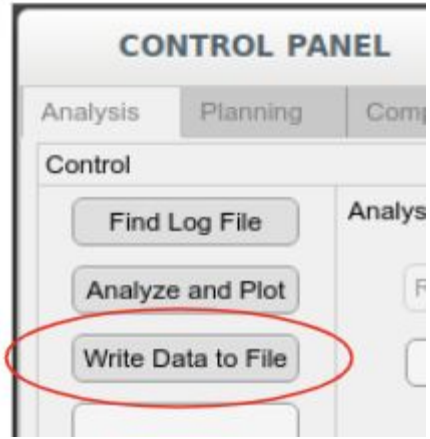
Example 1:


Analysis: Pseudorange errors
Raw pr errors (measured raw pr minus expected pr)
tau relative to GPS L1:
GloG1 11.5m, GalE1 3.0m, BdsB1 6.7m
GpsL5 1.7m, GalE5 4.3m
Svid,Freq, mean, σ (m)
G02,L1, 8.6, 3.7
G05,L1, 7.0, 3.3
G06,L1, 8.0, 5.1
G12,L1, 7.4, 3.7
G24,L1, 10.4, 7.4
G25,L1, 9.4, 3.5
G29,L1, 11.4, 4.3
G31,L1, 9.5, 4.5
G06,L5, 10.2, 4.1
G24,L5, 10.3, 3.6
G25,L5, 9.4, 3.1
R05,L1, 3.8, 5.0
R06,L1, 5.2, 6.9
R20,L1, 10.3, 6.6
R21,L1, 11.5, 6.3
(meters)
v3.0.3.0
Iono: klobuchar (GPS broadcast); Tropo: hopfield (20C, 1016hPa, 23hPa)
1σ error in clock bias estimate
(m)
Show Smoothed
(seconds)

Example 2:


Analysis: Pseudorange errors
Raw pr errors (measured raw pr minus expected pr)
tau relative to GPS L1:
GloG1 1461.9m, GalE1 -511.1m
Svid,Freq, mean, σ (m)
G08,L1, 15.2, 8.0
G10,L1, 15.7, 6.8
G11,L1, 18.9, 22.3
G14,L1, 18.2, 7.1
G18,L1, 20.7, 9.0
G20,L1, 26.0, 19.0
G21,L1, 18.4, 14.2
G24,L1, 13.8, 7.7
G27,L1, 16.2, 7.0
G32,L1, 17.1, 7.2
R03,L1, 19.4, 19.8
R04,L1, 18.6, 7.2
R13,L1, 12.5, 35.5
R20,L1, 14.7, 11.8
(meters)
300
200
100
0
-100
v3.0.3.0
Iono: klobuchar (GPS broadcast); Tropo: hopfield (20C, 1016hPa, 23hPa)
1σ error in clock bias estimate
(m)
20
10
0
60
80
100
120
140
160
180
200
(seconds)
Show Smoothed

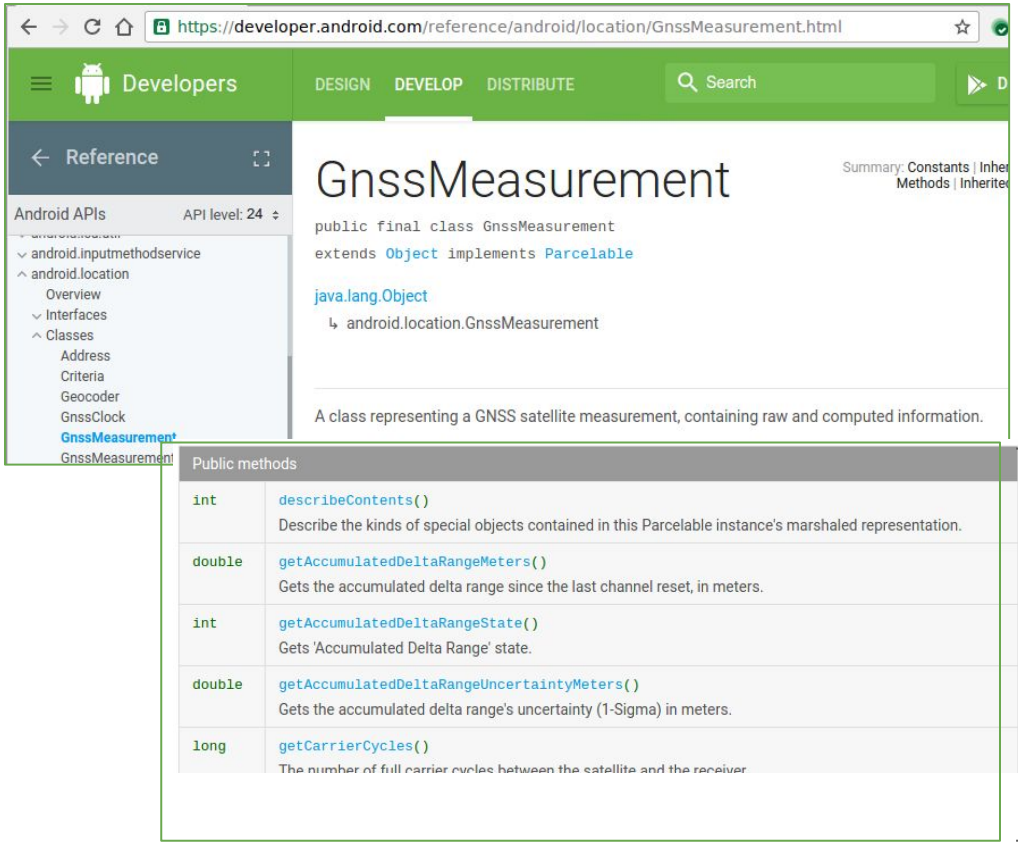
The inter system biases, as measured by the receiver, are saved in the derived data file



gnss_log_2018_08_08_18_56_58_pixel3.derived

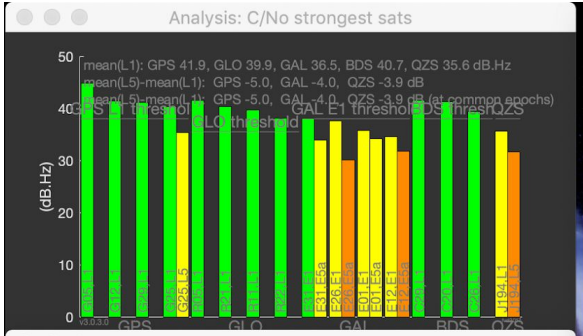
```
# Header Description:
#
# Version: v3.0.3.0
# Start: UTC [2018,08,09,01,56,54.439], GPS [2013, 352632.439]
# Intersystem bias (nanoseconds): TauGloG1 4876.515, TauGalE1 -1704.802 ,
TauBdsB1 NaN, TauQzsJ1 NaN, TauGpsL5 NaN, TauGalE5 NaN, TauBdsB2 NaN, TauQzsJ5
NaN
```

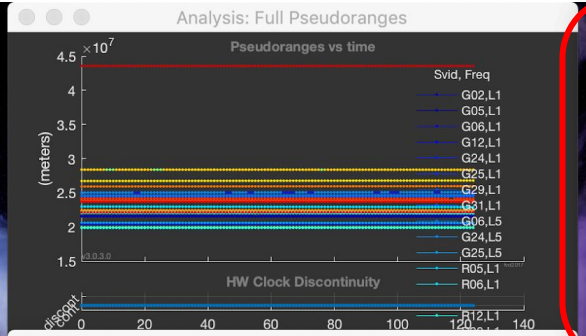
Carrier Phase

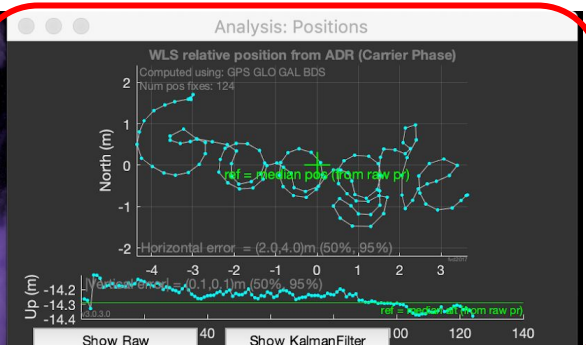


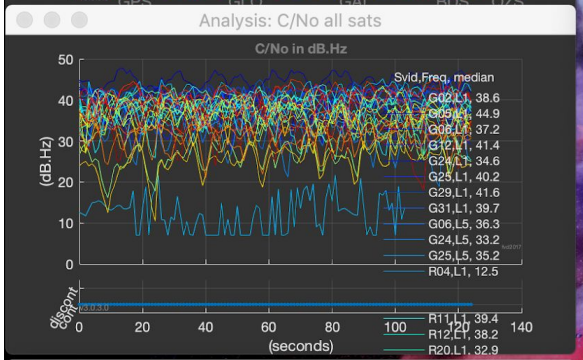
Public methods

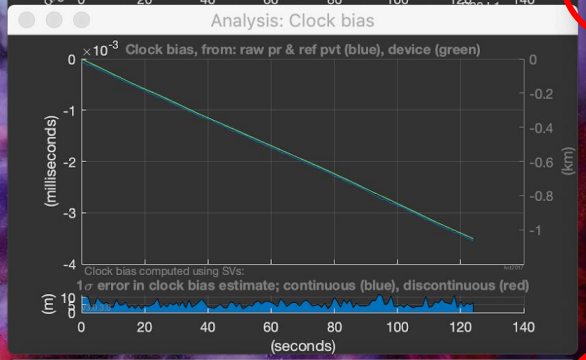
- getAccumulatedDeltaRangeMeters()
- getAccumulatedDeltaRangeState()
- getAccumulatedDeltaRangeUncertaintyMeters()
- getCarrierFrequencyHz()
- getCn0DbHz()
- getConstellationType()
- getMultipathIndicator()
- getPseudorangeRateMetersPerSecond()
- getPseudorangeRateUncertaintyMetersPerSecond()
- getReceivedSvTimeNanos()
- getReceivedSvTimeUncertaintyNanos()
- getSnrInDb()
- getState()
- getSvid()
- getTimeOffsetNanos()
- hasCarrierFrequencyHz()
- hasSnrInDb()

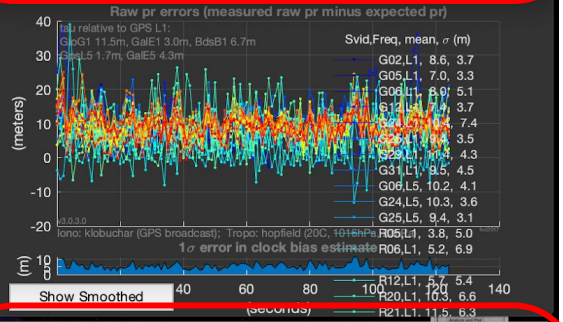

Analysis: C/No strongest sats
mean(L1): GPS 41.9, GLO 39.9, GAL 36.5, BDS 40.7, QZS 35.6 dB.Hz
mean(L5)-mean(L1): GPS -5.0, GAL -4.0, QZS -3.9 dB
mean(L5)-mean(L1): GPS -5.0, GAL -4.0, QZS -3.9 dB (at common epochs)
GPS L1 threshold
GLO threshold
GAL E1 threshold
BDS threshold
QZS threshold
(dB.Hz)
GPS
GLO
GAL
BDS
QZS

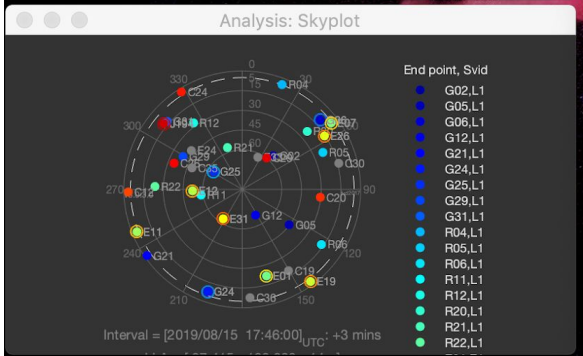

Analysis: Full Pseudoranges
Pseudoranges vs time
Svid, Freq
G02,L1
G05,L1
G06,L1
G12,L1
G24,L1
G25,L1
G29,L1
G31,L1
G06,L5
G24,L5
G25,L5
R05,L1
R06,L1
R12,L1
(meters)
HW Clock Discontinuity
discont

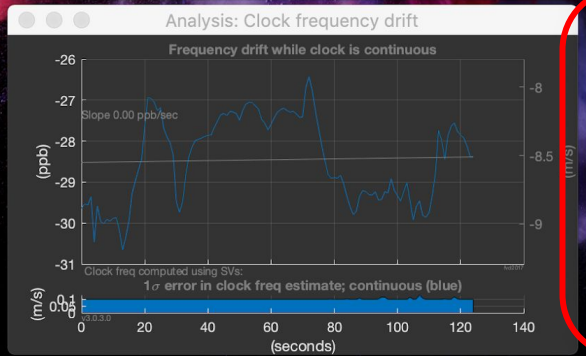

Analysis: Positions
WLS relative position from ADR (Carrier Phase)
Computed using: GPS GLO GAL BDS
Num pos fixes: 124
North (m)
ref = median pos (from raw pr)
Horizontal error = (2.0,4.0)m (50%, 95%)
Vertical error = (0.1,0.1)m (50%, 95%)
Up (m)
ref = receiver antenna (from raw pr)
Show Raw
Show KalmanFilter

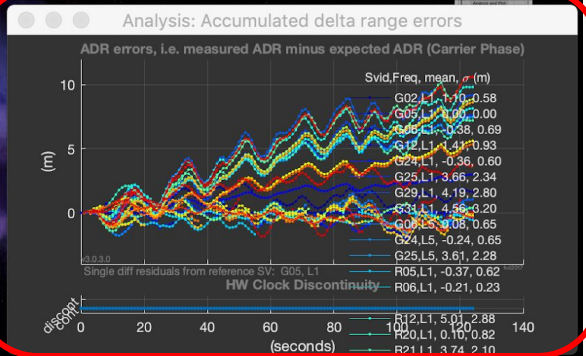

Analysis: C/No all sats
C/No in dB.Hz
Svid,Freq, median
G02,L1, 38.6
G05,L1, 44.9
G06,L1, 37.2
G12,L1, 41.4
G24,L1, 34.6
G25,L1, 40.2
G29,L1, 41.6
G31,L1, 39.7
G06,L5, 36.3
G24,L5, 33.2
G25,L5, 35.2
R04,L1, 12.5
R11,L1, 39.4
R12,L1, 38.2
R20,L1, 32.9
(dB.Hz)
(seconds)
discont


Analysis: Clock bias
Clock bias, from: raw pr & ref pvt (blue), device (green)
(milliseconds)
(km)
Clock bias computed using SVs:
1σ error in clock bias estimate; continuous (blue), discontinuous (red)
(m)
(seconds)

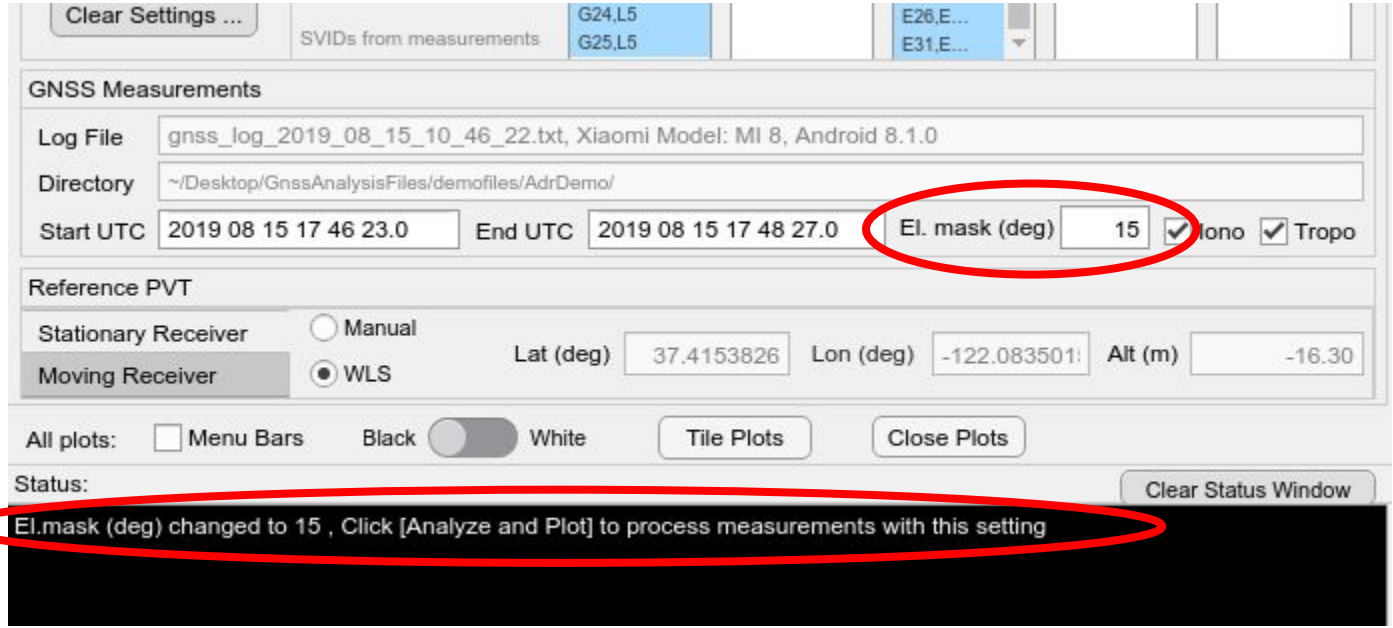

Raw pr errors (measured raw pr minus expected raw pr)
tau relative to GPS L1:
GloG1 11.5m, GalE1 3.0m, BdsB1 6.7m
GalL5 1.7m, GalE5 4.3m
Svid,Freq, mean, σ (m)
G02,L1, 8.6, 3.7
G05,L1, 7.0, 3.3
G06,L1, 8.0, 5.1
G12,L1, 7.0, 3.7
G24,L1, 7.4, 7.4
G25,L1, 8.4, 3.5
G29,L1, 8.4, 4.3
G31,L1, 9.5, 4.5
G06,L5, 10.2, 4.1
G24,L5, 10.3, 3.6
G25,L5, 9.4, 3.1
R05,L1, 3.8, 5.0
R06,L1, 5.2, 6.9
R12,L1, 7.0, 5.4
R20,L1, 10.3, 6.6
R21,L1, 11.5, 6.3
Iono: klobuchar (GPS broadcast); Tropo: hopfield (20C, 101hPa, 50%)
1σ error in clock bias estimate
(m)
(seconds)
Show Smoothed


Analysis: Skyplot
End point, Svid
G02,L1
G05,L1
G06,L1
G12,L1
G21,L1
G24,L1
G25,L1
G29,L1
G31,L1
R04,L1
R05,L1
R06,L1
R11,L1
R12,L1
R20,L1
R21,L1
R22,L1
Interval = [2019/08/15 17:46:00]UTC: +3 mins


Analysis: Clock frequency drift
Frequency drift while clock is continuous
Slope 0.00 ppb/sec
(ppb)
(m/s)
Clock freq computed using SVs:
1σ error in clock freq estimate; continuous (blue)
(seconds)

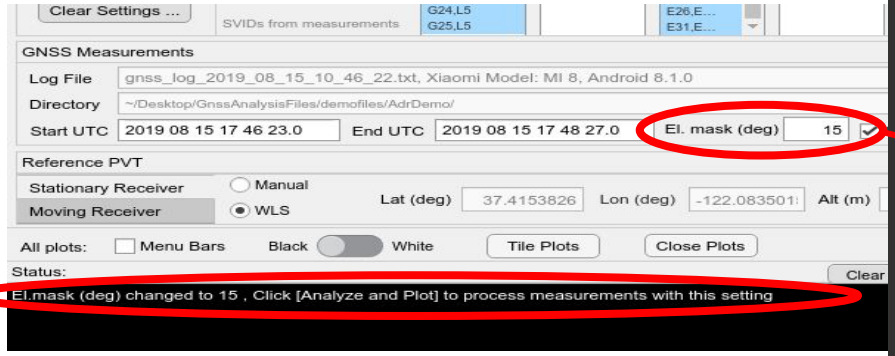

Analysis: Accumulated delta range errors
ADR errors, i.e. measured ADR minus expected ADR (Carrier Phase)
Svid,Freq, mean, σ (m)
G02,L1, 1.10, 0.58
G05,L1, 0.00, 0.00
G06,L1, -0.38, 0.69
G12,L1, 1.41, 0.93
G24,L1, -0.36, 0.60
G25,L1, 3.66, 2.34
G29,L1, 4.19, 2.80
G31,L1, 4.56, 3.20
G06,L5, 0.08, 0.65
G24,L5, -0.24, 0.65
G25,L5, 3.61, 2.28
R05,L1, -0.37, 0.62
R06,L1, -0.21, 0.23
R12,L1, 5.01, 2.88
R20,L1, 0.10, 0.82
R21,L1, 3.74, 2.10
(m)
Single diff residuals from reference SV: G05, L1
HW Clock Discontinuity
discont
(seconds)

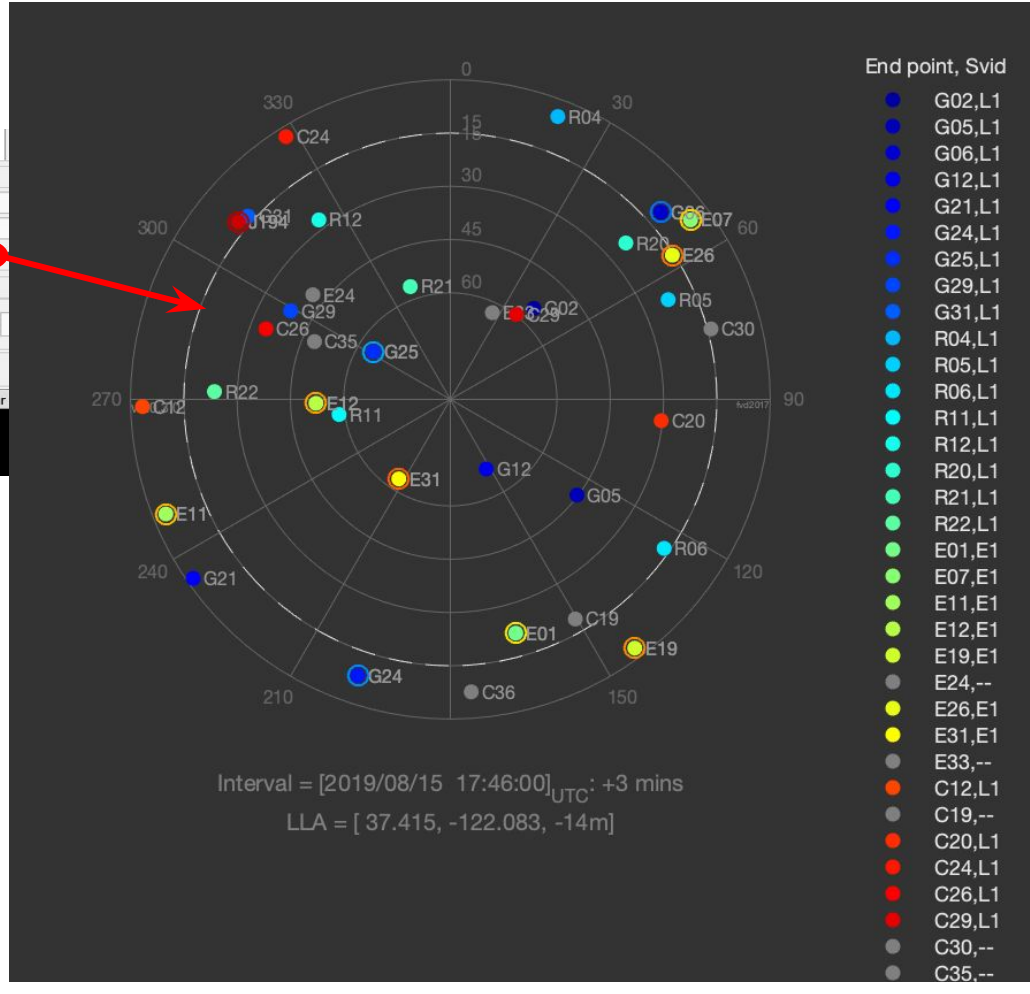
Elevation mask angle



“El. mask (deg) changed to 15. Click [Analyze and Plot] to process measurements with this setting”

Elevation mask angle





Satellites below the mask angle are not used.

Google

Custom Data Filter

Custom filtering of measurements by creating a script file, **CustomDataFilter.m**

You may filter on any of the "# Raw" header values in the GnssLogger log file:

```
# Raw,ElapsedRealtimeMillis,TimeNanos,LeapSecond,TimeUncertaintyNanos,...
  FullBiasNanos,BiasNanos,BiasUncertaintyNanos,DriftNanosPerSecond,...
  DriftUncertaintyNanosPerSecond,HardwareClockDiscontinuityCount,Svid,...
  TimeOffsetNanos,State,ReceivedSvTimeNanos,ReceivedSvTimeUncertaintyNanos,...
 Cn0DbHz,PseudorangeRateMetersPerSecond,PseudorangeRateUncertaintyMetersPerSecond,...
AccumulatedDeltaRangeState,AccumulatedDeltaRangeMeters,...
  AccumulatedDeltaRangeUncertaintyMeters,CarrierFrequencyHz,...
  MultipathIndicator,SnrInDb,ConstellationType,AgcDb
```

Further details in the new User Manual at https://g.co/gnsstools

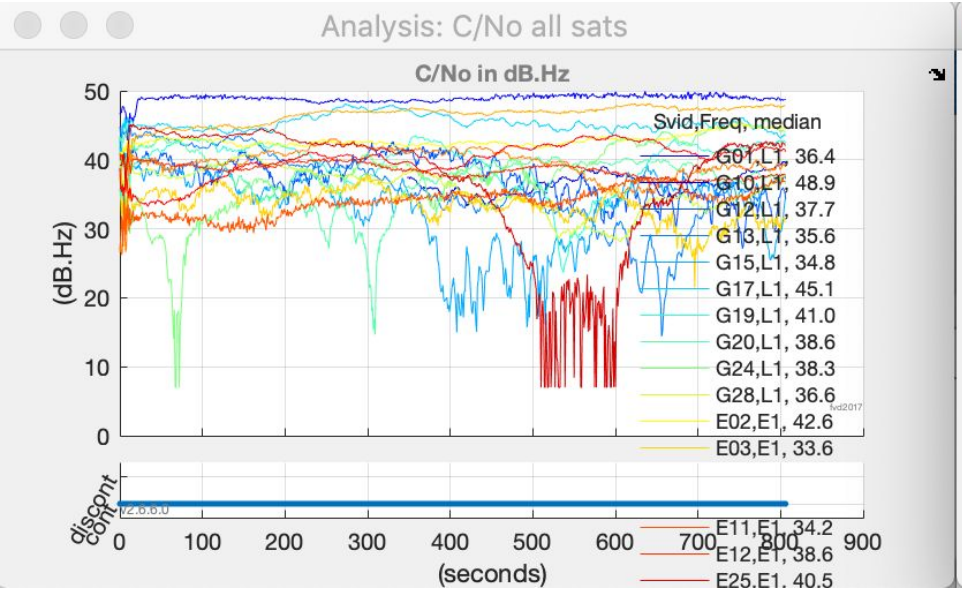
CustomDataFilter.m, example 1:

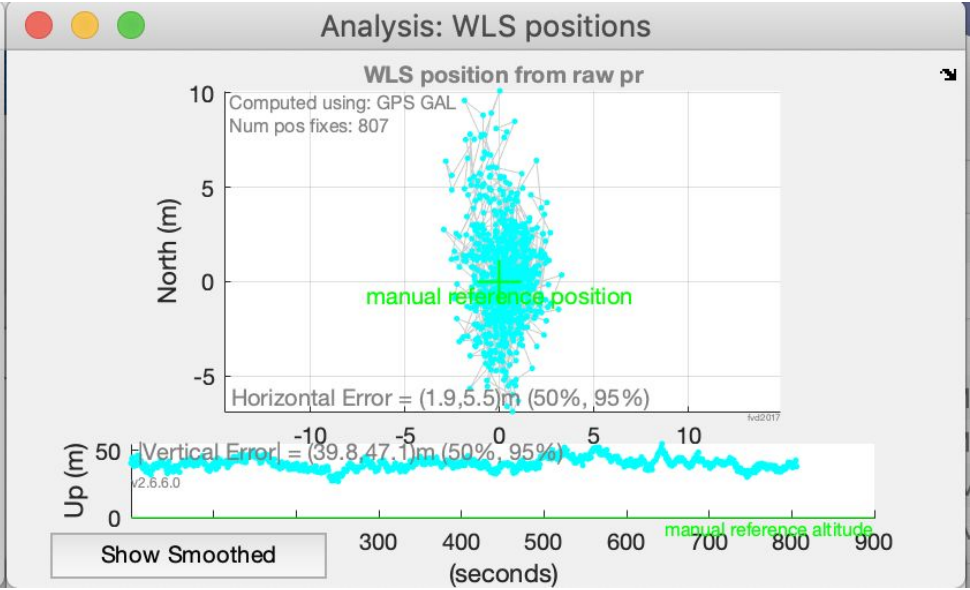
```
dataFilter={};

%Example of how to select satellites from GPS and GAL and :
dataFilter{end+1,1} = 'ConstellationType';
dataFilter{end,2} = '(ConstellationType)==1 | (ConstellationType)==6';
%Constellation types as specified in Android Raw Measurements API

dataFilter{end+1,1} = 'CarrierFrequencyHz';
dataFilter{end,2}   = 'CarrierFrequencyHz>1500e6'; %L1 only
```

GPS+GAL, L1: 1.9, 5.5 m (50%, 95%)





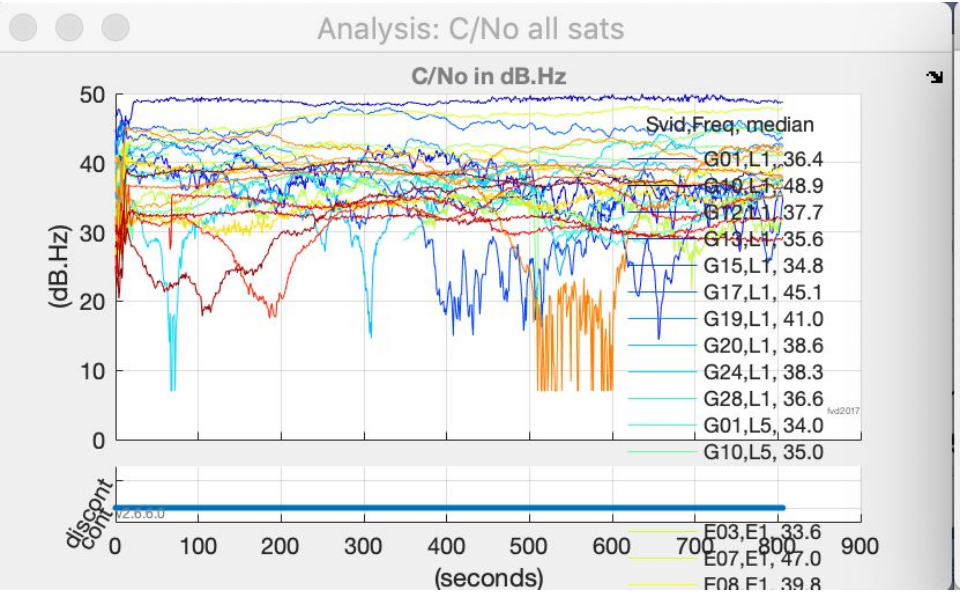
CustomDataFilter.m, example 2:

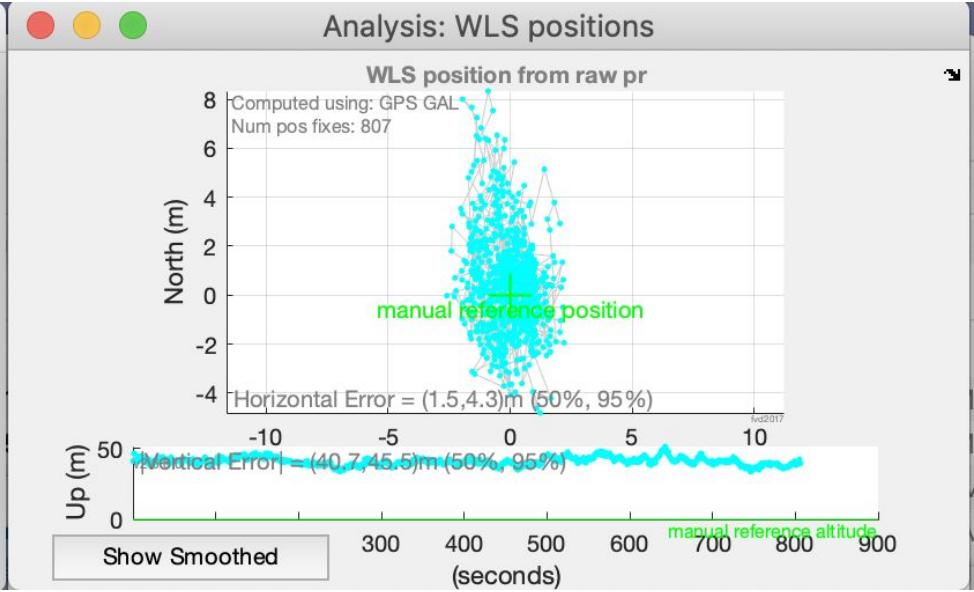
```
dataFilter={};

%Example of how to select satellites from GPS and GAL and :
dataFilter{end+1,1} = 'ConstellationType';
dataFilter{end,2} = '(ConstellationType)==1 | (ConstellationType)==6';
%Constellation types as specified in Android Raw Measurements API
```

GPS+GAL, L1: 1.9, 5.5 m (50%, 95%)

GPS+GAL, L1L5: 1.5, 4.3 m





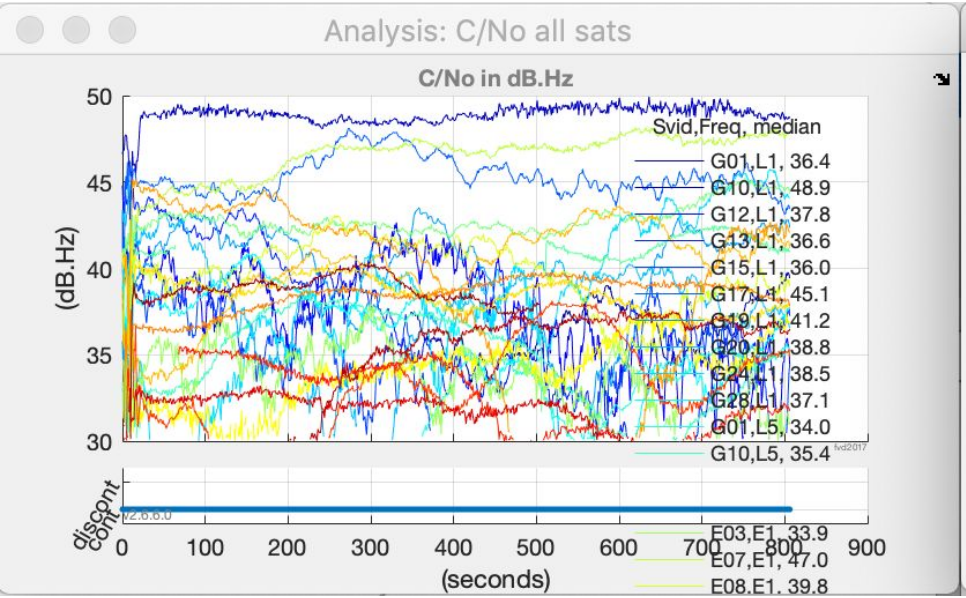
CustomDataFilter.m, example 3:

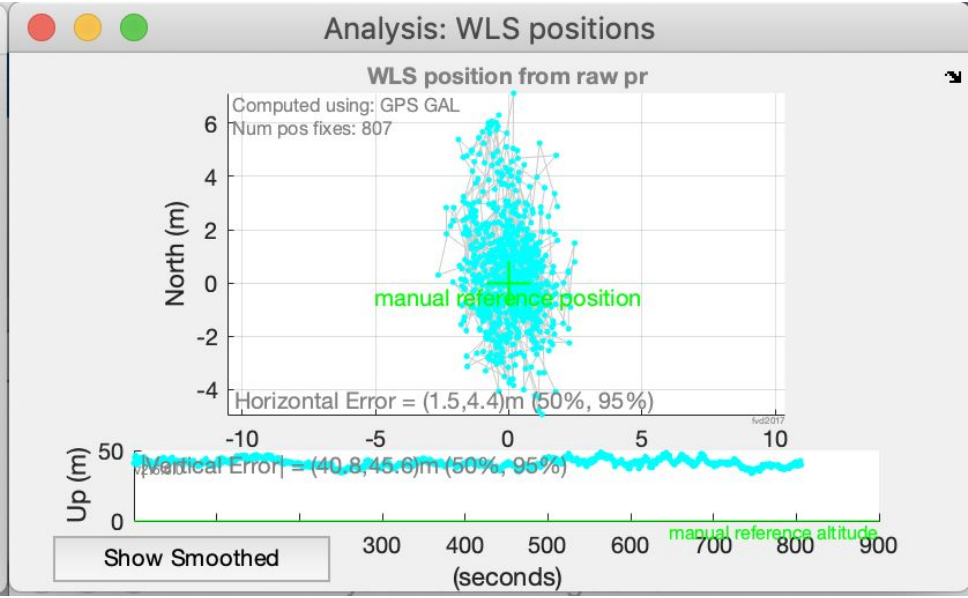
```
dataFilter={};

%Example of how to select satellites from GPS and GAL and :
dataFilter{end+1,1} = 'ConstellationType';
dataFilter{end,2} = '(ConstellationType)==1 | (ConstellationType)==6';
%Constellation types as specified in Android Raw Measurements API

dataFilter{end+1,1} = 'Cn0DbHz';
dataFilter{end,2}   = 'Cn0DbHz>30';
```

GPS+GAL, L1: 1.9, 5.5 m (50%, 95%)
GPS+GAL, L1L5: 1.5, 4.3 m
G+G, L1L5, >30dBHz: 1.5, 4.4 m





the end. Thank You!

extra slides follow …

GNSS Frequency Spectrum: Highlighting L1,L5

Shaded spectra (L2, E6, GLONASS CDMA, IRNSS)
Red = Military/Regulated use only

Shaded and red signals are not commonly found in consumer GNSS.

Green and Blue signals are available in new smartphones.

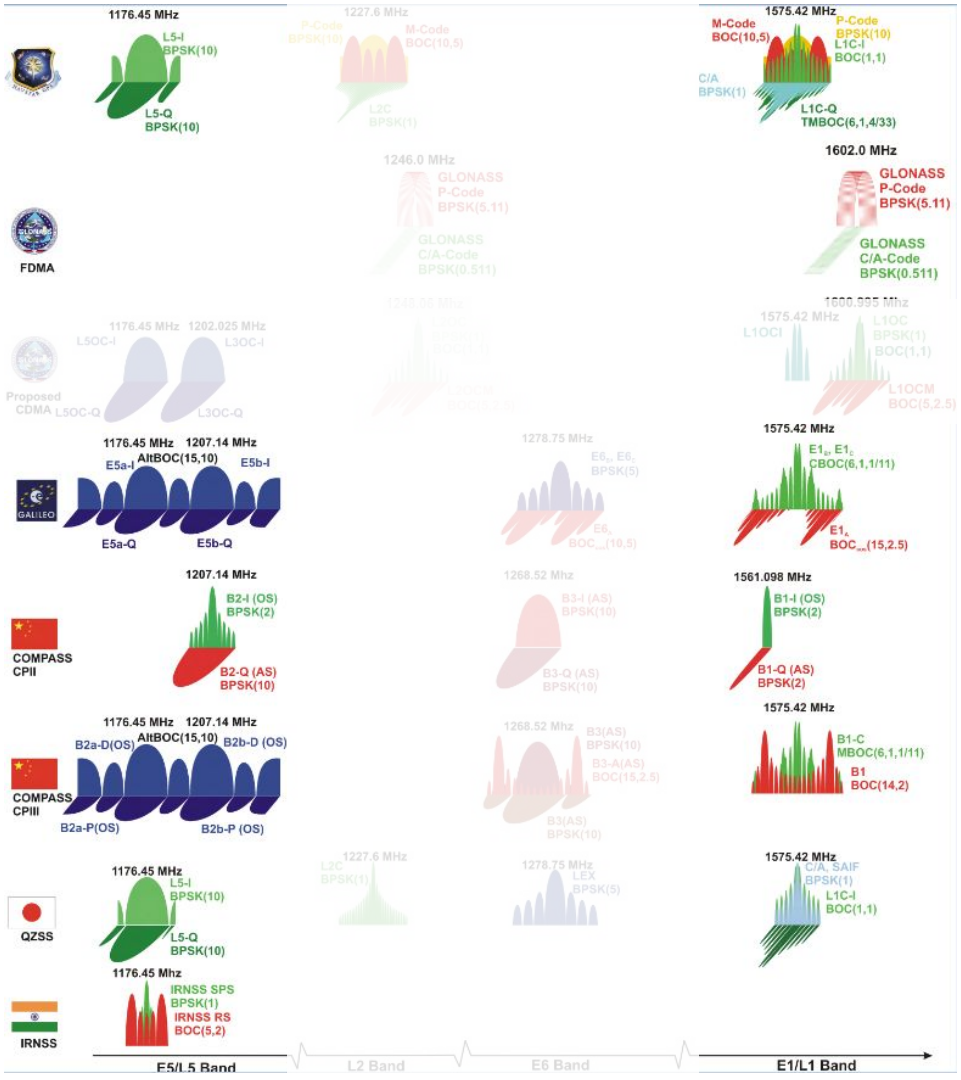
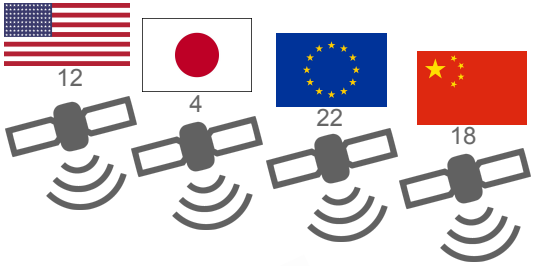
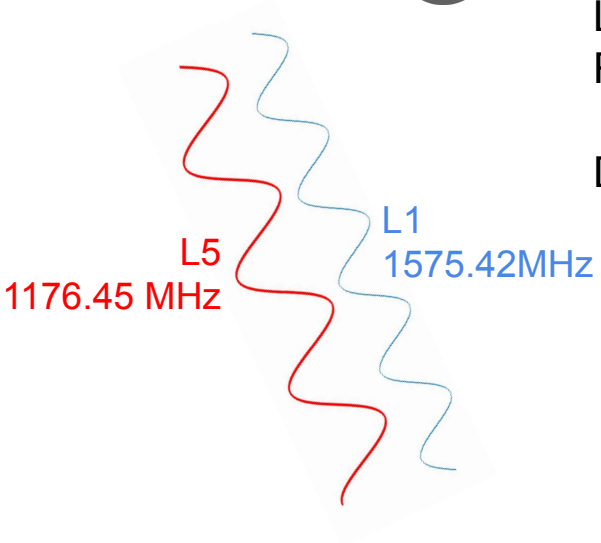


Image source: Source: Stefan Wallner, https://gssc.esa.int/navipedia/index.php/GNSS_signal





September 2019

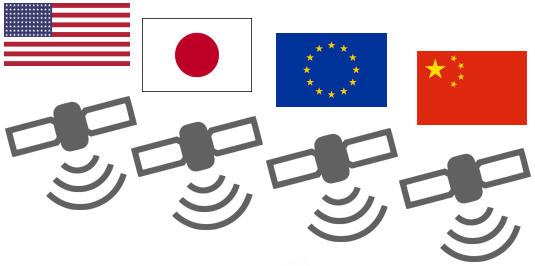
L1,L5 signals from 56 operational satellites

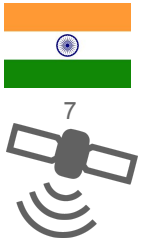
Raw measurements, including carrier phase, from smartphones

Details will be covered in GNSS 102 class, 3:30pm

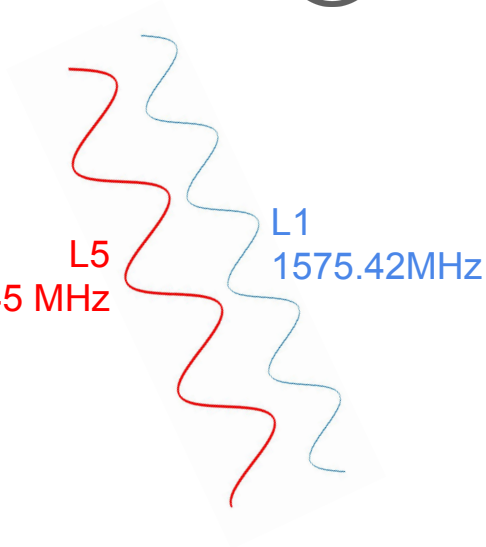
"L1" 1575.42MHz: GPS, QZSS L1, Galileo E1, BeiDou B1C
"L5" 1176.45MHz: GPS, QZSS L5, Galileo E5a, BeiDou B2a

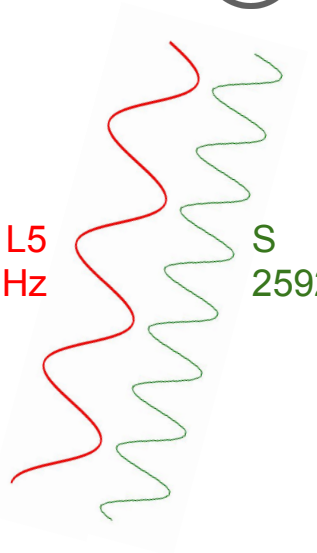
Sources:
https://www.navcen.uscg.gov/?Do=constellationStatus
http://qzss.go.jp/en/technical/satellites/index.html#QZSS
https://www.gsc-europa.eu/system-status/Constellation-Information
http://www.csno-tarc.cn/system/constellation





NAVIC (previously IRNSS) transmits on L5 and S bands

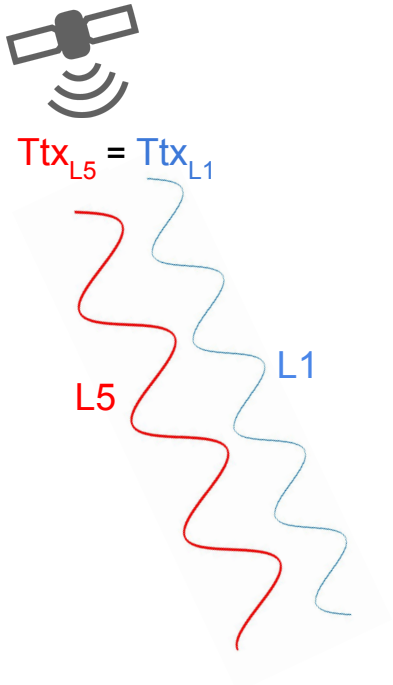




"L1" 1575.42MHz: GPS, QZSS L1, Galileo E1, BeiDou B1C
"L5" 1176.45MHz: GPS, QZSS L5, Galileo E5a, BeiDou B2a

Sources:
https://www.navcen.uscg.gov/?Do=constellationStatus
http://qzss.go.jp/en/technical/satellites/index.html#QZSS
https://www.gsc-europa.eu/system-status/Constellation-Information
http://www.csno-tarc.cn/system/constellation

Group Delay



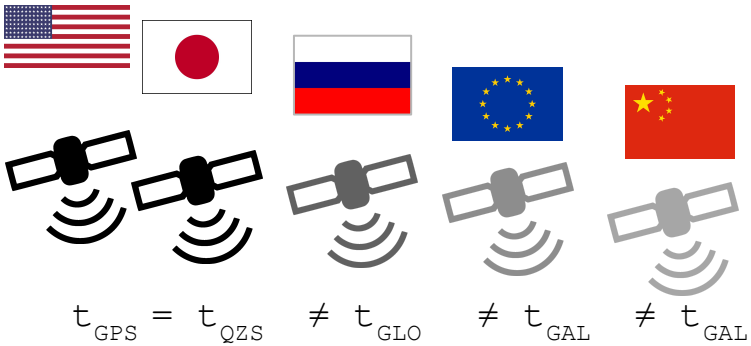
When two signals of different frequency pass through a medium (including: Earth's atmosphere, and the RF chain of the receiver), they experience different time delays relative to each other.

Also, the signal processing in different GNSS chips might result in significant differences in L1 v L5 delays, and this must be measured and compensated for before using L1 and L5 measurements together.

Observed Ttx_{L5} ≠ Observed Ttx_{L1}

i.e. $Pseudorange_{L5}$ ≠ $Pseudorange_{L1}$

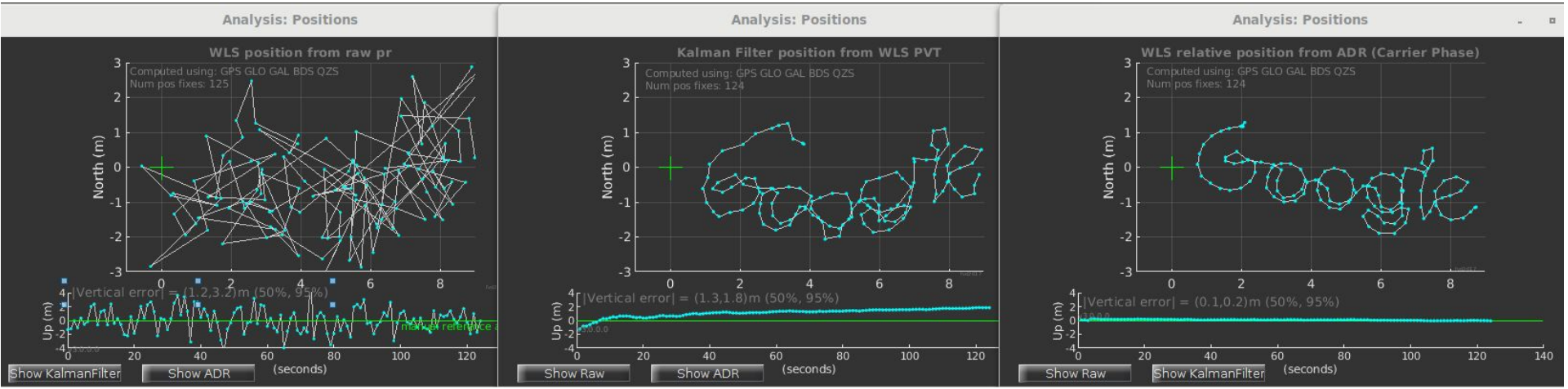
Inter System Bias



The different GNSSes are synced to different times (except QZS which manages to be synced to GPS)

Moreover, different GPS receivers will treat different signal types differently, resulting in further differences in observed times. These offsets be measured and accounted for before combining pseudoranges from different systems.

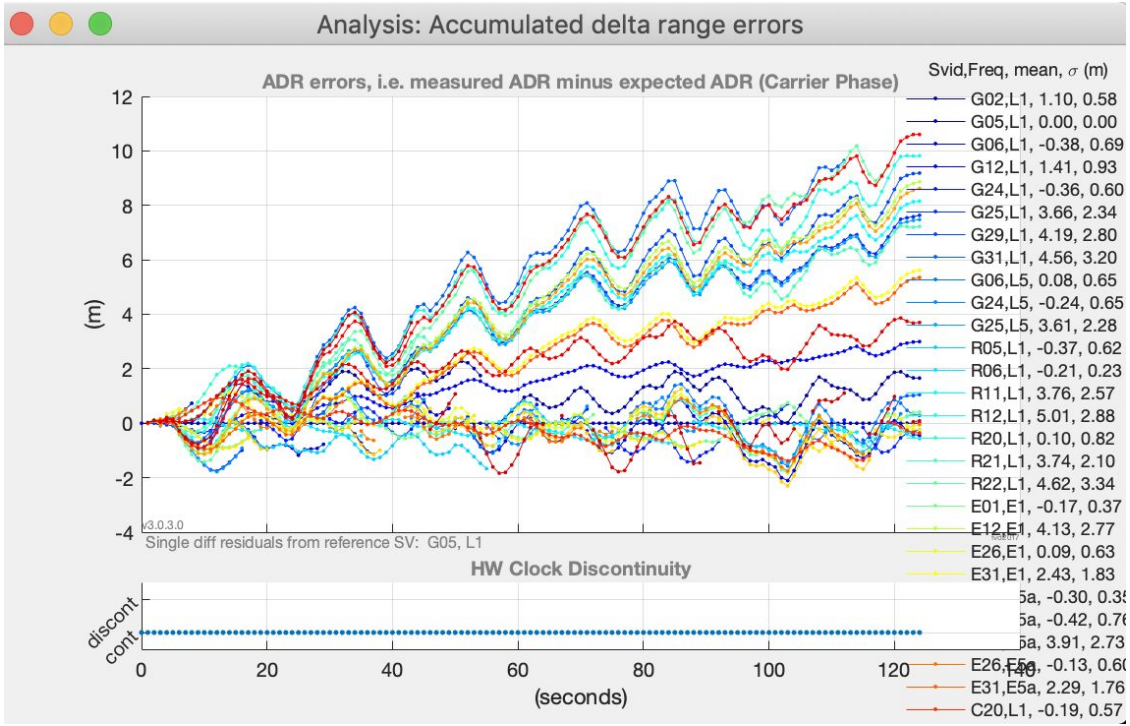
Raw pseudorange, Kalman Filter, ADR

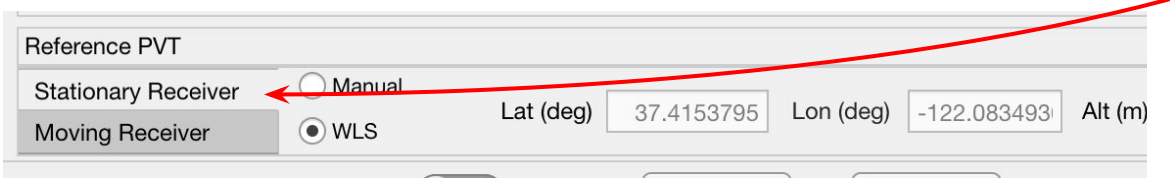


Comparison of positions from the same data set, left to right:

- Raw Pseudorange, processed with Weighted Least Squares (weighted by reported measurement uncertainty).
- Kalman Filter, using WLS position, and WLS velocity (velocity computed from Doppler measurements)
  Not shown: WLS position from smoothed pseudorange, is similar to Kalman Filter position.
- WLS position from ADR (Carrier Phase).

Conclusion: you should expect to get noisy positions from *Raw* Pseudoranges.



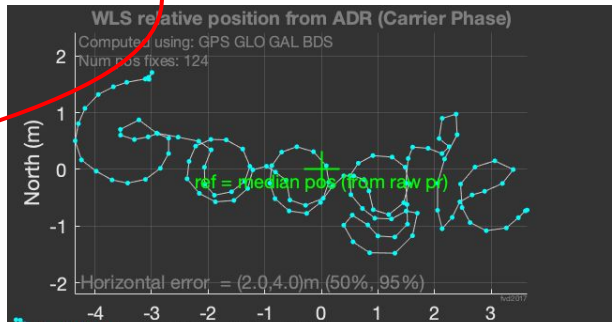


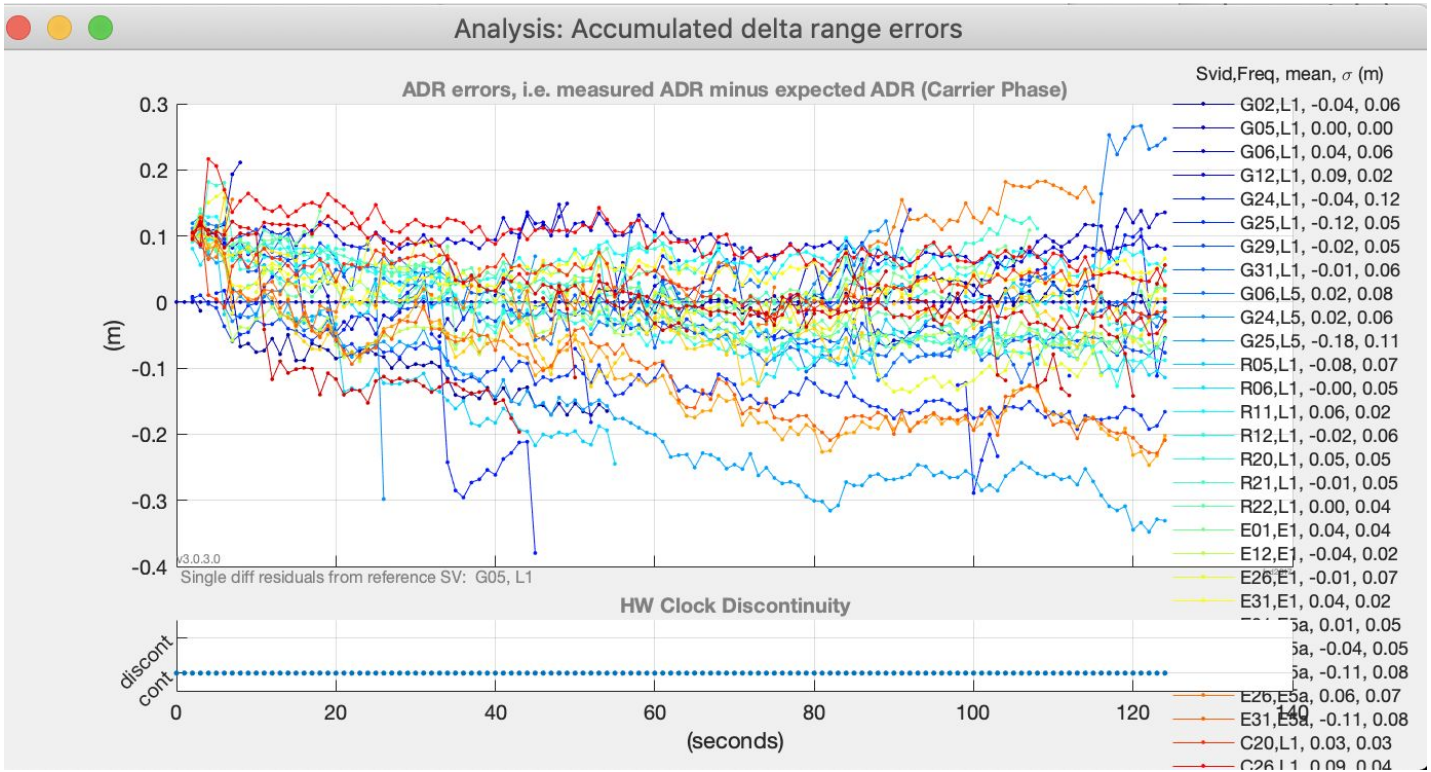
Residual errors, single-differenced from reference SV G05 L1.

Using Reference PVT specified by user, in this case “Stationary Receiver, WLS median position”.

QUESTION: Why are these residual errors growing over time?

Hint:





Reference PVT

Stationary Receiver ○ Device

Moving Receiver ◉ NMEA NMEA File: gnss_log_2019_08_15_10_46_22_Adr.nmea

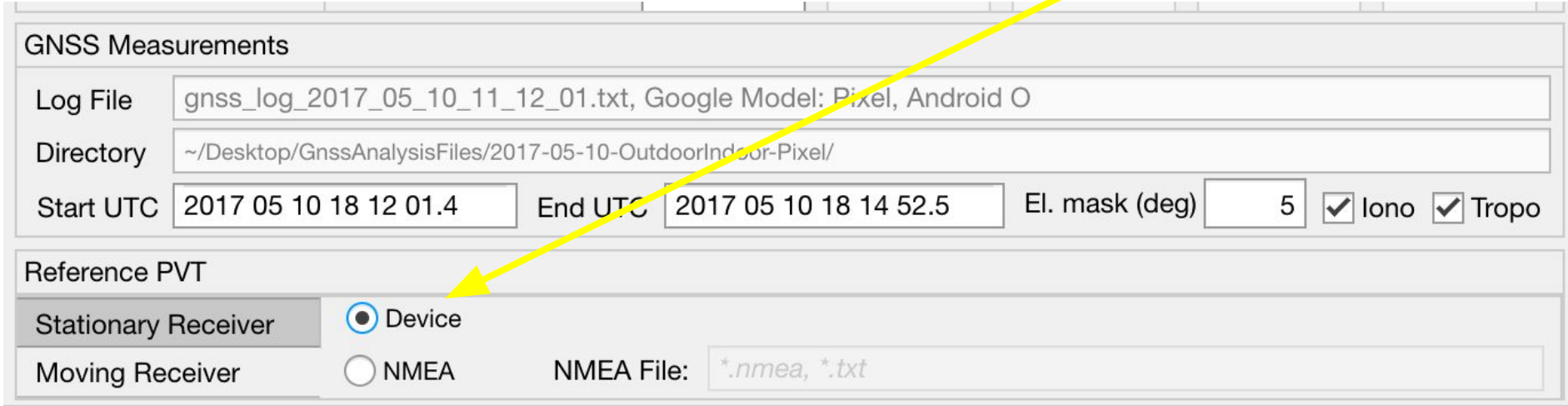
Residual errors, single-differenced from reference SV G05 L1.

Using Reference PVT: nmea file from previously computed ADR position

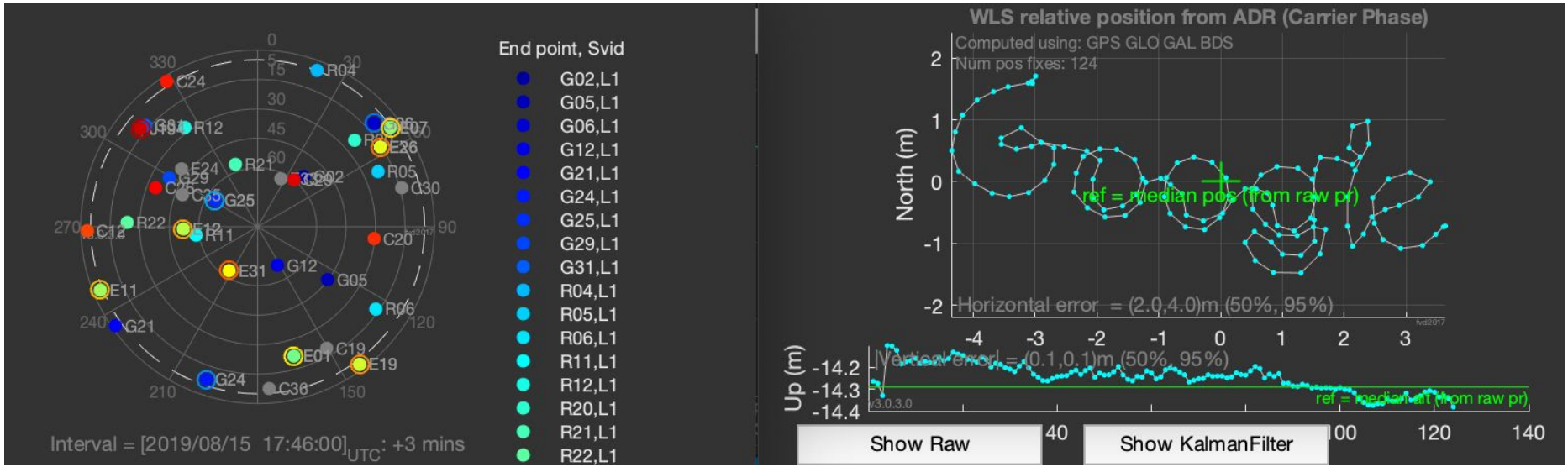
Device position for reference PVT

Measurement Error Plots depend on the reference PVT (Position, Velocity, Time)

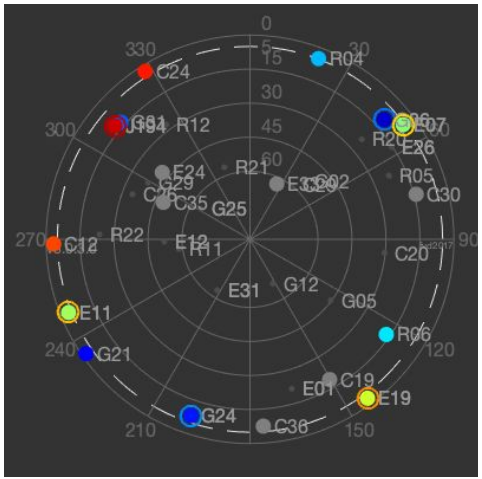
- Stationary
  - You enter it Manually, or ...
  - App will work it out for itself (median WLS)
- Moving
  - You provide an NMEA file with reference PVT, or ...
  - App will read the device GNSS position from the GnssLogger log file



Select satellites for position



Select satellites for position



Analysis Plots

Refresh SVID Plots

(hide deselected SVIDs)

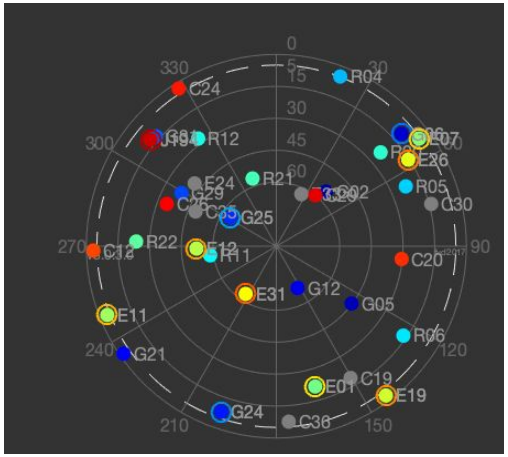
Refresh Positions

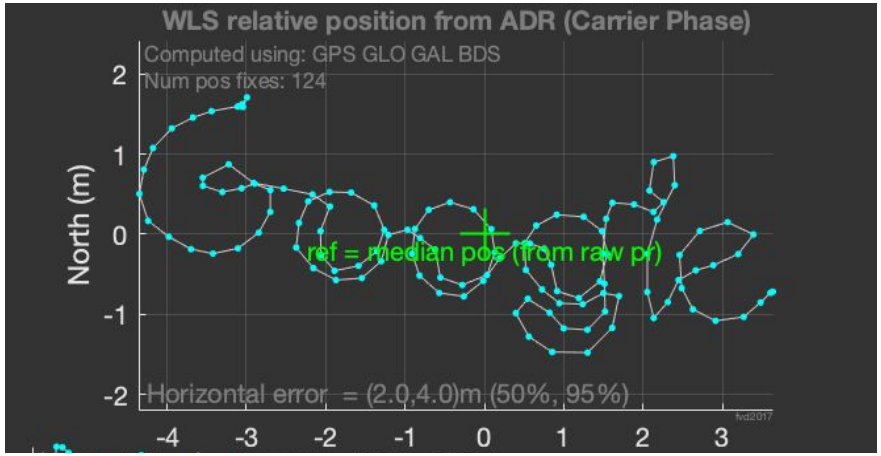
(WLS, Kalman, ADR)

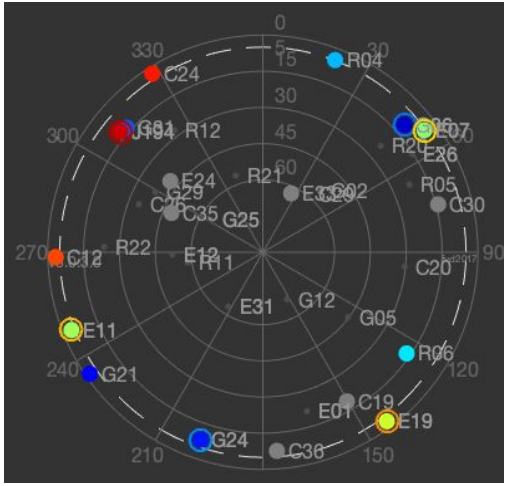
SVIDs from measurements

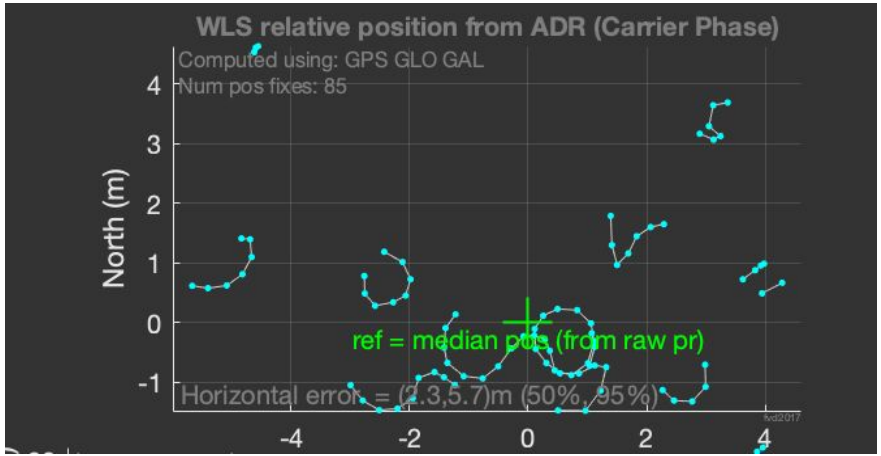
GPS	GLO	GAL	BDS	QZS
G02,L1	R04,L1	E12,E1	C12,L1	J194,L1
G05,L1	R05,L1	E19,E1	C20,L1	J194,L5
G06,L1	R06,L1	E26,E1	C24,L1	
G12,L1	R11,L1	E31,E1	C26,L1	
G21,L1	R12,L1	E01,E5a	C29,L1	
G24,L1	R20,L1	E07,E5a		
G25,L1	R21,L1	E11,E5a		
G29,L1	R22,L1	E12,E5a		
G31,L1		E19,E5a		
G06,L5		E26,E5a		
G24,L5		E31,E5a		

Suppose you wanted to remove all satellites *above* 30 degrees elevation. What would the position look like?


0
30
60
90
120
150
180
210
240
270
300
330
C24
R04
R12
E24
R21
G29
C35
G25
R22
E12
R11
C20
E31
G12
G05
E11
R06
G21
C19
E01
E19
G24
C36


WLS relative position from ADR (Carrier Phase)
Computed using: GPS GLO GAL BDS
Num pos fixes: 124
North (m)
ref = median pos (from raw pr)
Horizontal error = (2.0,4.0)m (50%, 95%)
fvd2017


0
30
60
90
120
150
180
210
240
270
300
330
C24
R04
R12
E24
R21
G29
C35
G25
R22
E12
R11
C20
E31
G12
G05
E11
R06
G21
C19
E01
E19
G24
C36


WLS relative position from ADR (Carrier Phase)
Computed using: GPS GLO GAL
Num pos fixes: 85
North (m)
ref = median pos (from raw pr)
Horizontal error = (2.3,5.7)m (50%, 95%)
fvd2017