Wireless Security

Prof. Marco Mellia
Dr. Andrea Nardin

Wireless Security Lab

Spoofing Android GNSS measurements

Andrea Nardin

Contents

- Lab briefing: Android GNSS measurements
- Lab Tasks

Contents

Motivation

Examples of mobile applications using location data

Use of COTS devices for Precise Positioning

- Cost/form factor and size effective
- Processing capacity
- Benefits of embedded sensors and communication interfaces
- Going beyond single frequency
  (e.g., Broadcom BCM47755 in Xiaomi Mi 8 Pro)

Availability of GNSS raw measurements from Android 7 onwards

- Opens the door to more advanced GNSS processing techniques
- Raw measurements allow to optimize the multi-GNSS solutions
  [on-chip PVT can be optional]

Release of white paper for using GNSS Measurements on Android Devices

- Bridges the gap between GNSS experts and Android developers

GNSS in COTS devices: pills of history

1999: Benefon Esc

May 2018: Xiaomi Mi 8 Pro

March 2004: TomTom

Google Maps

February 2005: Google Maps

https://www.gpsworld.com/wirelesssmartphone-revolution-9183/ *By Frank van Diggelen, Broadcom Corporation*

1999 — Mobile phone manufacturer Benefon launched the **first commercially-available GPS phone**, a safety phone called the Benefon Esc! The GSM phone was sold mainly in Europe, but many other GPS-enabled mobile phones would follow.

May 2000 — Switch-off of the Selective Availability (SA)

2001 — As GPS receiver technology got much smaller and cheaper, private companies began pumping out **personal GPS products**, like the in-car navigation devices from Tom Tom and Garmin.

2006 — Google patented the SUPL service (US7714779B2) to provide "real-time" long-term-orbit **ephemeris data to mobile Android** devices by overcoming the need of demodulating the whole navigation message (almanac and ephemeris).

2008 — Google Maps was first released for Android and iOS after the introduction of **My Location** feature using **GPS/Assisted GPS** supplemented by wireless network and cell sites

2016 — Google allowed the **access to raw measurements** through a specific Location API available on Android API-Level 24.

2018 — The first **Dual Frequency GNSS smartphone**, the Xiaomi Mi 8 Pro, approached the market with the embedded Broadcom BCM47755 chipset

2020 — 2° Generation Broadcom Chipsets BCM47765 with High-Definition GPS (HDGPS)

! Optional on-chip PVT calculations !

GNSS Inside an Android phone

Xiaomi Mi 8 Pro

BROADCOM
BCM47755

GENERAL FEATURES

- Integrated **multi-frequency** GNSS baseband and RF front end for simultaneous reception of **GPS**, **GLONASS**, **BeiDou** (BDS), **Galileo** (GAL), and SBAS satellite systems
- Support for position batching, geofencing, sensor fusion and sensor navigation

ADC & CLOCK

- Integrated 12-bit, 2-channel ADC
- Timers: One Real-time Clock (RTC) (42 bits, 32.768 kHz)
- Two general-purpose 32-bit microsecond timers
- One 48-bit microsecond counter for better resolution timestamps than the RTC can provide

Behind an Android phone

- The most common way to obtain a position on Android is via a **fused location** provider that combines several sources (GNSS, Wi-Fi or even mobile networks) to improve the accuracy, time to first fix, availability or power consumption

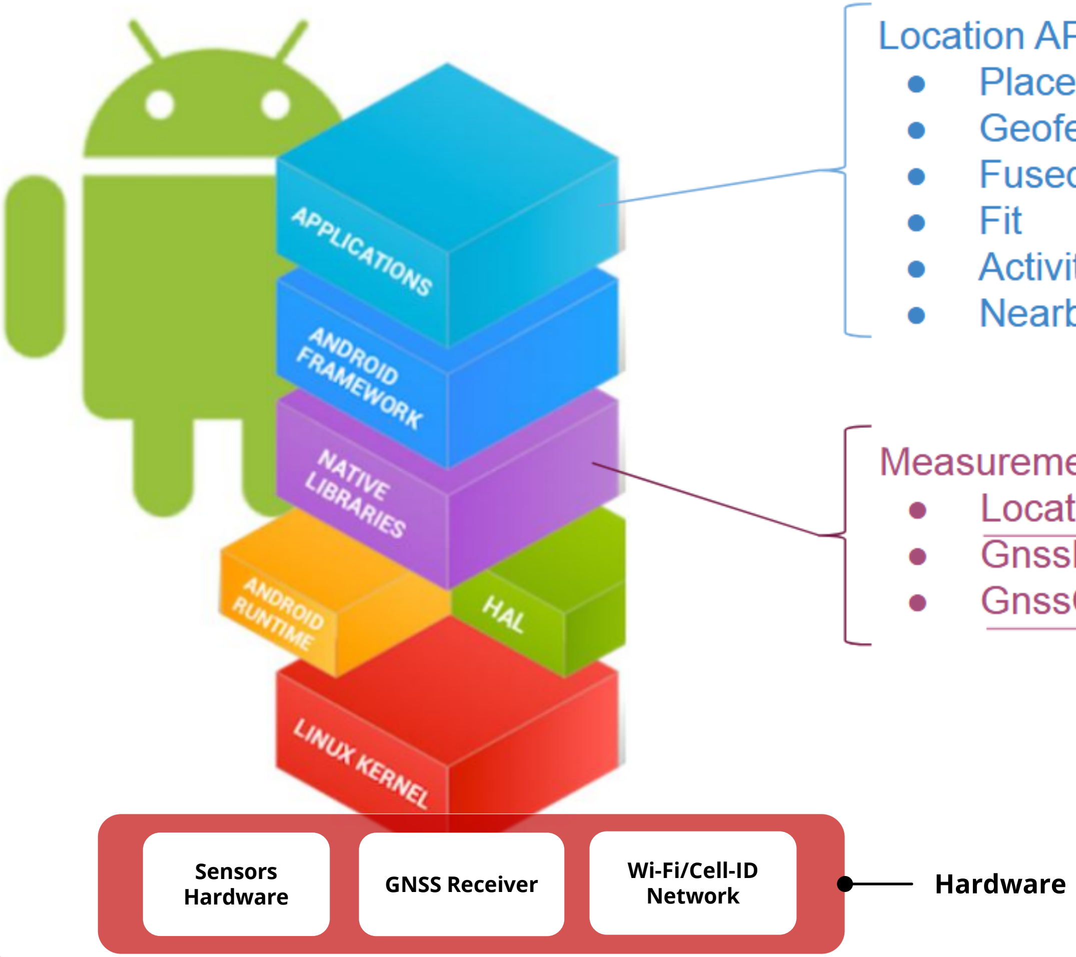
aka *Google Play Services* aka *Google Mobile Service*
Most Android phones have this (not China)

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

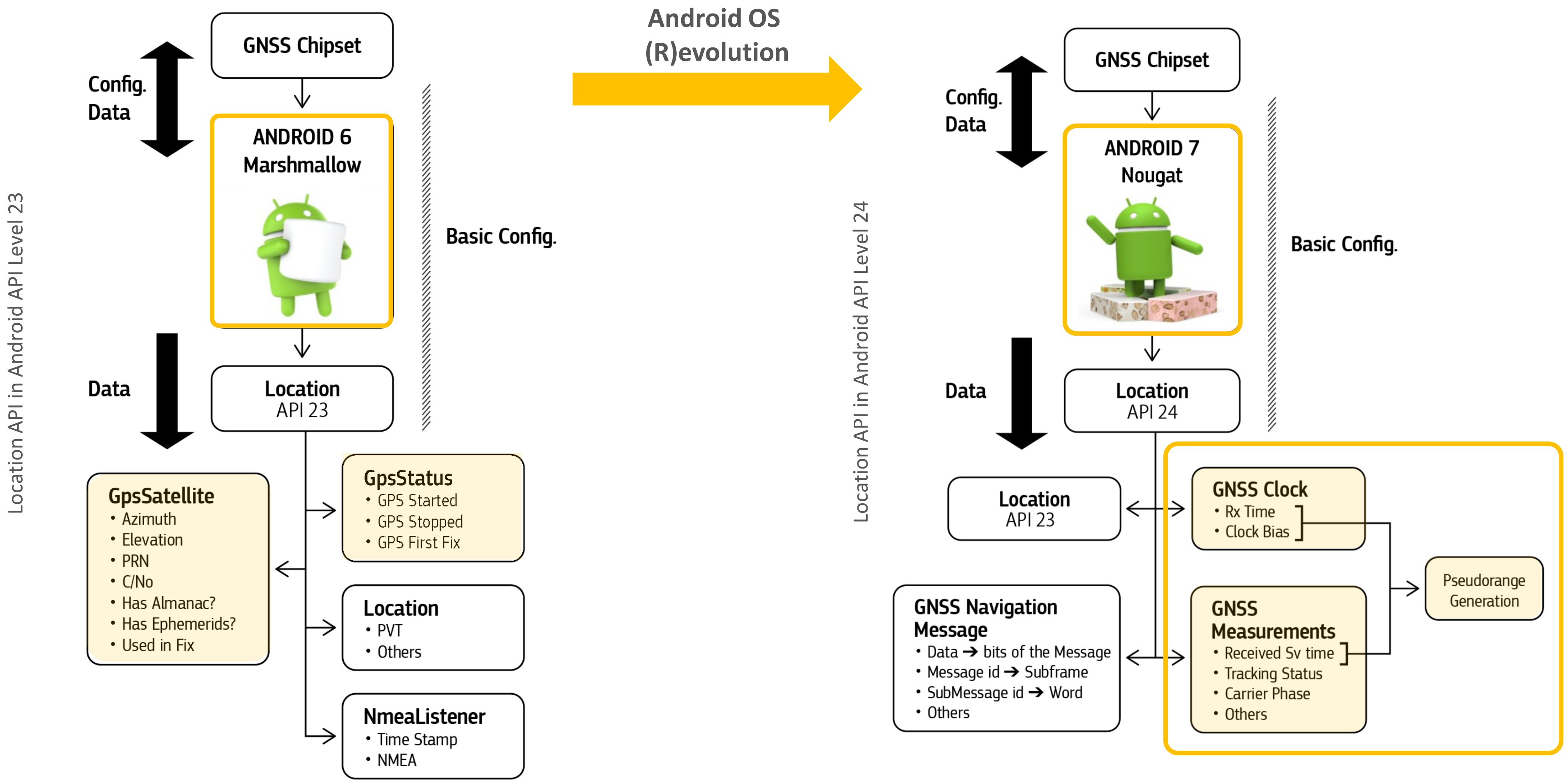
GNSS Raw Measurements
All phones with:
GNSS chips build date ≥ 2016
OS ≥ Android N (Nougat)



F. Van Diggelen, M. Khider, *Mobile Phones for Ionospheric Measurements*, ION Africa Outreach, Regional Workshop on GNSS and Space Weather Rabat, Morocco, May 2022

Access to GNSS Raw Measurements

- Starting with Android 6, developers could access basic GNSS data like **satellite info** and **position solutions** through the Location API.
- From API 24 (Android 7), this expanded to include **raw measurements**, **navigation messages**, and **GNSS clock data**
- Data come from the chipset, but direct access to the chipset is restricted



How to read data

Android 7 Location - Clock and Measurements		
ANDROID CLASS	FIELD	DESCRIPTION
GNSSClock	*TimeNanos*	GNSS receiver's internal hardware clock value in nanoseconds
GNSSClock	*BiasNanos*	Clock's sub-nanosecond bias
GNSSClock	*FullBiasNanos*	Difference between TimeNanos inside the GPS receiver and the true GPS time since 0000Z, 6 January 1980
GNSSClock	*DriftNanosPerSecond*	Clock's drift
GNSSClock	*HardwareClockDiscontinuityCount*	Count of hardware clock discontinuities
GNSSClock	*LeapSecond*	Leap second associated with the clock's time
GNSSMeasurement	*ConstellationType*	Constellation type
GNSSMeasurement	*Svid*	Satellite ID
GNSSMeasurement	*State*	Current state of the GNSS engine
GNSSMeasurement	*ReceivedSvTimeNanos*	Received GNSS satellite time at the measurement time
GNSSMeasurement	*AccumulatedDeltaRangeMeters*	Accumulated delta range since the last channel reset
GNSSMeasurement	*Cn0DbHz*	Carrier-to-noise density
GNSSMeasurement	*TimeOffsetNanos*	Time offset at which the measurement was taken in nanoseconds
GNSSMeasurement	*CarrierCycles*	Number of full carrier cycles between the satellite and the receiver
GNSSMeasurement	*CarrierFrequencyHz*	Carrier frequency at which codes and messages are modulated
GNSSMeasurement	*PseudorangeRatemetersperSecond*	Gets the Pseudorange rate at the timestamp

Fundamental to the determination of pseudorange measurements

Fundamental to the determination of pseudorange measurements

Fundamental to the determination of pseudorange measurements

Table included in the **White paper,** "Using GNSS Raw Measurements on Android Devices"

How to read data

Android 7 Location - Clock and Measurements		
ANDROID CLASS	**FIELD**	**DESCRIPTION**
GNSSClock	*TimeNanos*	GNSS receiver's internal hardware clock value in nanoseconds
GNSSClock	*BiasNanos*	Clock's sub-nanosecond bias
GNSSClock	*FullBiasNanos*	Difference between TimeNanos inside the GPS receiver and the true GPS time since 0000Z, 6 January 1980
GNSSClock	*DriftNanosPerSecond*	Clock's drift
GNSSClock	*HardwareClockDiscontinuityCount*	Count of hardware clock discontinuities
GNSSClock	*LeapSecond*	Leap second associated with the clock's time
GNSSMeasurement	*ConstellationType*	Constellation type
GNSSMeasurement	*Svid*	Satellite ID
GNSSMeasurement	*State*	Current state of the GNSS engine
GNSSMeasurement	*ReceivedSvTimeNanos*	Received GNSS satellite time at the measurement time
GNSSMeasurement	*AccumulatedDeltaRangeMeters*	Accumulated delta range since the last channel reset
GNSSMeasurement	*Cn0DbHz*	Carrier-to-noise density
GNSSMeasurement	*TimeOffsetNanos*	Time offset at which the measurement was taken in nanoseconds
GNSSMeasurement	*CarrierCycles*	Number of full carrier cycles between the satellite and the receiver
GNSSMeasurement	*CarrierFrequencyHz*	Carrier frequency at which codes and messages are modulated
GNSSMeasurement	*PseudorangeRatemetersperSecond*	Gets the Pseudorange rate at the timestamp

Table included in the **White paper,** "Using GNSS Raw Measurements on Android Devices"

getTimeNanos
Local Timestamp attributed to the measurements

For battery saving or other reasons determined by the operating system, clock might have discontinuities!

How to read data

Android 7 Location - Clock and Measurements		
ANDROID CLASS	**FIELD**	**DESCRIPTION**
GNSSClock	*TimeNanos*	GNSS receiver's internal hardware clock value in nanoseconds
GNSSClock	*BiasNanos*	Clock's sub-nanosecond bias
GNSSClock	*FullBiasNanos*	Difference between TimeNanos inside the GPS receiver and the true GPS time since 0000Z, 6 January 1980
GNSSClock	*DriftNanosPerSecond*	Clock's drift
GNSSClock	*HardwareClockDiscontinuityCount*	Count of hardware clock discontinuities
GNSSClock	*LeapSecond*	Leap second associated with the clock's time
GNSSMeasurement	*ConstellationType*	Constellation type
GNSSMeasurement	*Svid*	Satellite ID
GNSSMeasurement	*State*	Current state of the GNSS engine
GNSSMeasurement	*ReceivedSvTimeNanos*	Received GNSS satellite time at the measurement time
GNSSMeasurement	*AccumulatedDeltaRangeMeters*	Accumulated delta range since the last channel reset
GNSSMeasurement	*Cn0DbHz*	Carrier-to-noise density
GNSSMeasurement	*TimeOffsetNanos*	Time offset at which the measurement was taken in nanoseconds
GNSSMeasurement	*CarrierCycles*	Number of full carrier cycles between the satellite and the receiver
GNSSMeasurement	*CarrierFrequencyHz*	Carrier frequency at which codes and messages are modulated
GNSSMeasurement	*PseudorangeRatemetersperSecond*	Gets the Pseudorange rate at the timestamp

Table included in the **White paper,** "Using GNSS Raw Measurements on Android Devices"

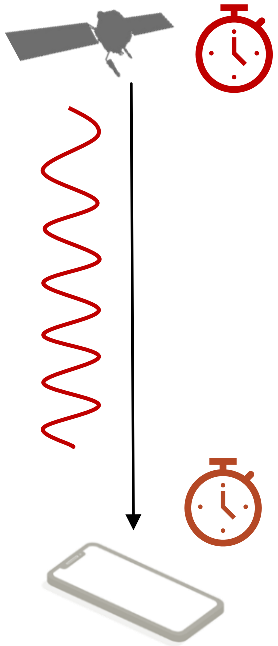
```
getTimeNanos – (getFullBiasNanos+getBiasNanos)
```

Absolute timestamp (GNSS time)

How to determine pseudorange measurements

TRANSMISSION TIME
Transmission time in GNSS time scale

```
TxTime= ReceivedSvTimeNanos
```



RECEPTION TIME
Absolute Timestamp attributed to the measurements

```
Rxtime= getTimeNanos - (getFullBiasNanos+getBiasNanos) -weekNumberNanos
```

How to determine pseudorange measurements

TRANSMISSION TIME
Transmission time in GNSS time scale

```
TxTime = ReceivedSvTimeNanos
```

RECEPTION TIME
Absolute Timestamp attributed to the measurements

```
Rxtime = getTimeNanos – (getFullBiasNanos+getBiasNanos) -weekNumberNanos
```

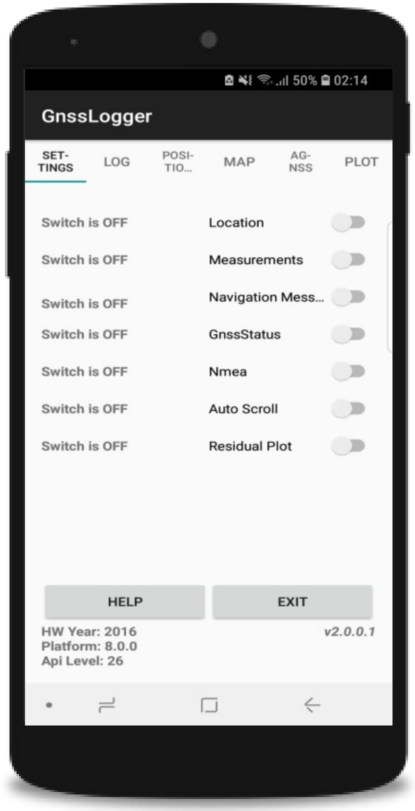
ρ

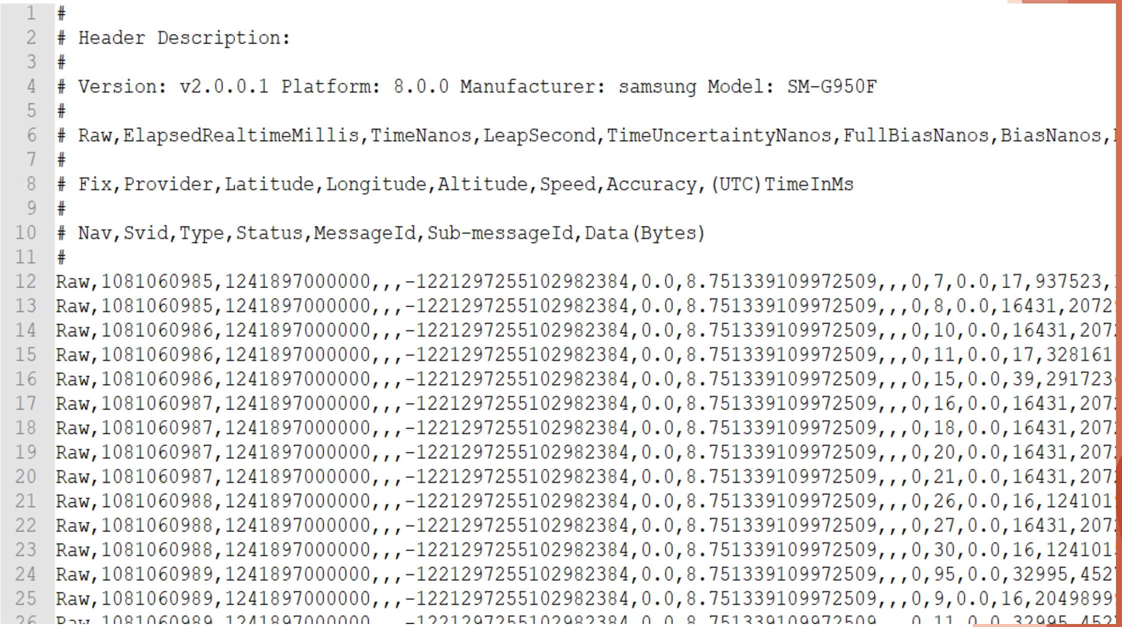
PSEUDORANGE MEASUREMENTS
Geometrical distance between satellite and receiver still affected by local clock bias

```
Raw = (Rxtime- TxTime)*c
```

How to perform an Analysis

1) ACQUIRING & LOG MEASUREMENTS on Android Phones



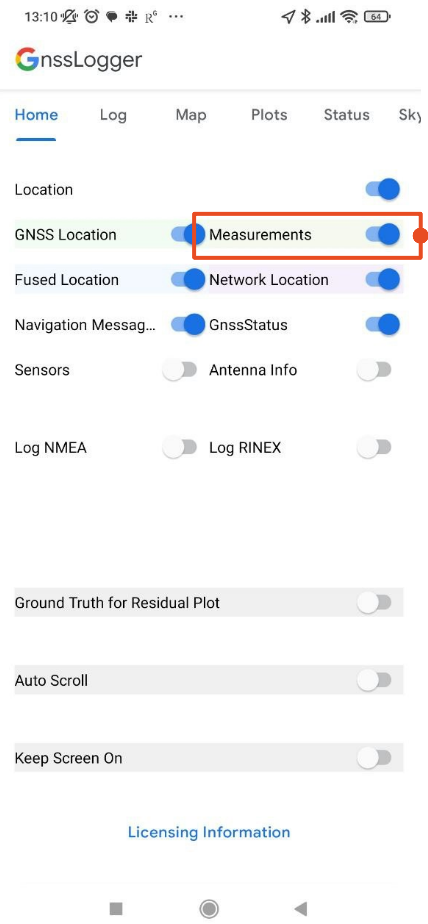


```
#
# Header Description:
#
# Version: v2.0.0.1 Platform: 8.0.0 Manufacturer: samsung Model: SM-G950F
#
# Raw,ElapsedRealtimeMillis,TimeNanos,LeapSecond,TimeUncertaintyNanos,FullBiasNanos,BiasNanos,
#
# Fix,Provider,Latitude,Longitude,Altitude,Speed,Accuracy,(UTC)TimeInMs
#
# Nav,Svid,Type,Status,MessageId,Sub-messageId,Data(Bytes)
#
Raw,1081060985,1241897000000,,,-1221297255102982384,0.0,8.751339109972509,,,0,7,0.0,17,937523,
Raw,1081060985,1241897000000,,,-1221297255102982384,0.0,8.751339109972509,,,0,8,0.0,16431,2072
Raw,1081060986,1241897000000,,,-1221297255102982384,0.0,8.751339109972509,,,0,10,0.0,16431,207
Raw,1081060986,1241897000000,,,-1221297255102982384,0.0,8.751339109972509,,,0,11,0.0,17,328161
Raw,1081060986,1241897000000,,,-1221297255102982384,0.0,8.751339109972509,,,0,15,0.0,39,291723
Raw,1081060987,1241897000000,,,-1221297255102982384,0.0,8.751339109972509,,,0,16,0.0,16431,207
Raw,1081060987,1241897000000,,,-1221297255102982384,0.0,8.751339109972509,,,0,18,0.0,16431,207
Raw,1081060987,1241897000000,,,-1221297255102982384,0.0,8.751339109972509,,,0,20,0.0,16431,207
Raw,1081060987,1241897000000,,,-1221297255102982384,0.0,8.751339109972509,,,0,21,0.0,16431,207
Raw,1081060988,1241897000000,,,-1221297255102982384,0.0,8.751339109972509,,,0,26,0.0,16,124101
Raw,1081060988,1241897000000,,,-1221297255102982384,0.0,8.751339109972509,,,0,27,0.0,16431,207
Raw,1081060988,1241897000000,,,-1221297255102982384,0.0,8.751339109972509,,,0,30,0.0,16,124101
Raw,1081060989,1241897000000,,,-1221297255102982384,0.0,8.751339109972509,,,0,95,0.0,32995,452
Raw,1081060989,1241897000000,,,-1221297255102982384,0.0,8.751339109972509,,,0,9,0.0,16,2049899
```

2) PROCESSING LOGGED MEASUREMENTS through **Google's MATLAB tool** to inspect data and compute position, velocity, and time (PVT)

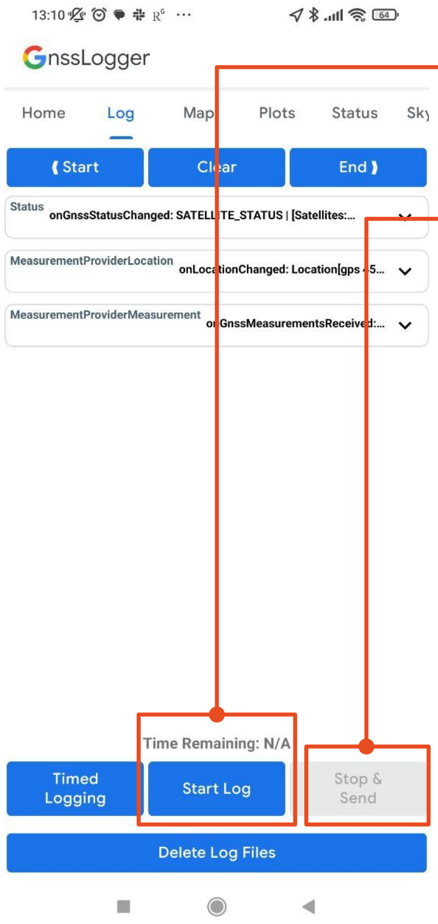
How to perform an Analysis: GNSS Logger App

Home tab



- «Measurements» <u>**must**</u> be switched on
- Other toggles can be on, but they are not essential

Log tab

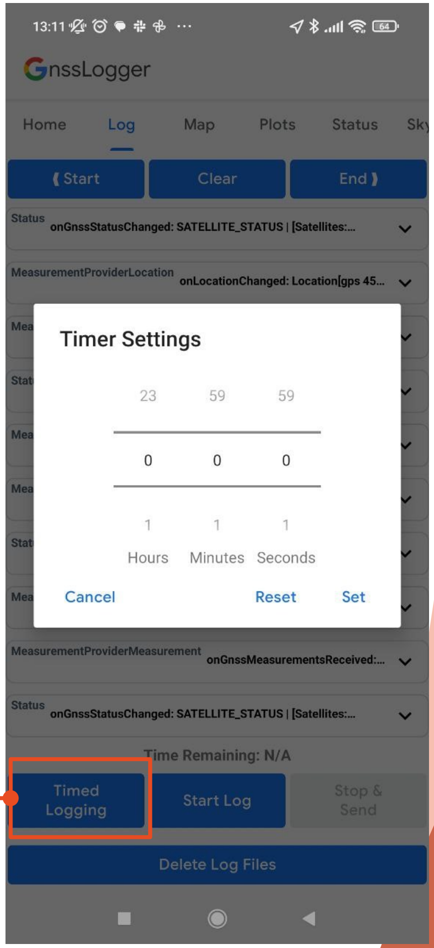


- It starts saving the log data

- Data logging stops. The app asks you were to send the log file (you should manage to get them to your PC)

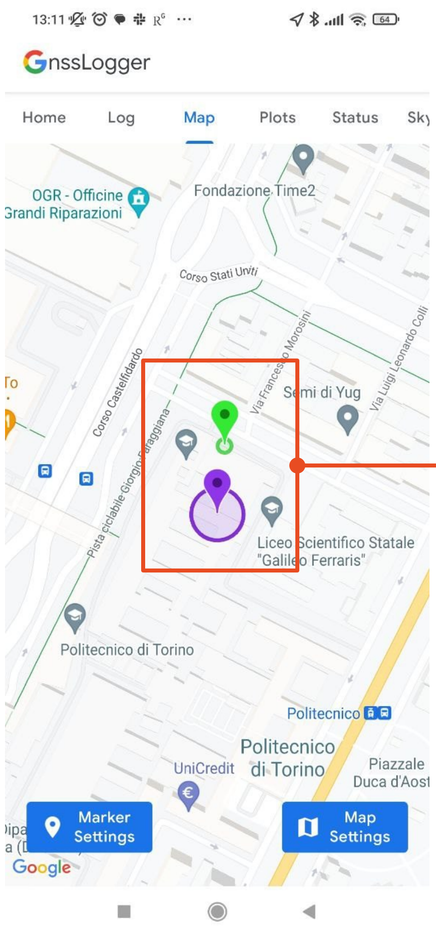
- Timed logging is useful to ensure that multiple data collections have the same length (useful to compare experiments' outputs)

Log tab



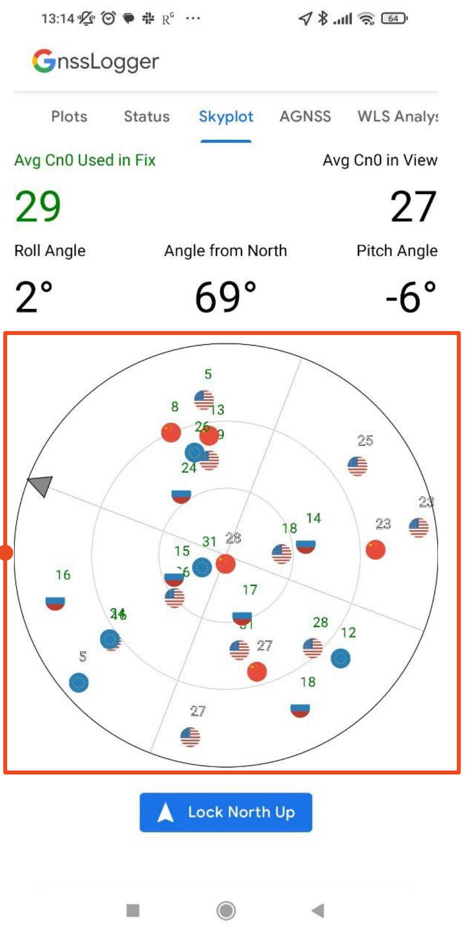
How to perform an Analysis: GNSS Logger App

Map tab



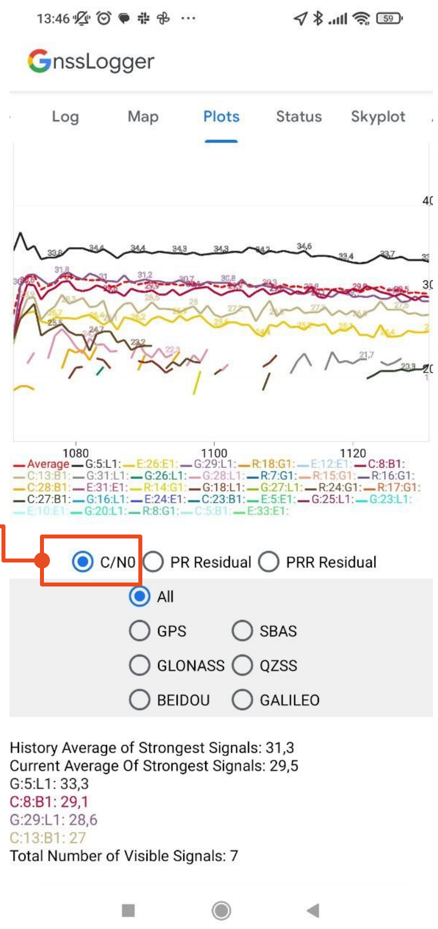
- Positions from different OS location providers are shown (GNSS, network, fused)
- The GNSS position (green) is different from the network location provider position (purple)

Skyplot tab



- A ***Skyplot*** shows the position of satellites in terms of azimuth and elevation
- A satellite at the center of the plot is on top of your head (zenith)
- A satellite on the outer circle is at the horizon
- You can also get **skyplots at a given spacetime location from** https://www.gnssplanning.com/#/settings

Plots tab



- C/N0 plots show you which satellites are visible and which are the strongest signals

Material and references

- **White paper,** "Using GNSS Raw Measurements on Android Devices"
  - https://www.gsc-europa.eu/sites/default/files/sites/all/files/gnss_raw_measurement_web_0.pdf

- **GNSS Analysis MATLAB code (or app) and Android Logger APK can be downloaded from:**
  - **GnssAnalysis[OS].zip**: https://github.com/google/gps-measurement-tools/releases
    Gnsslogger.apk: https://github.com/google/gps-measurement-tools/releases/tag/2.0.0.1

References:

[1] https://www.gpsworld.com/wirelesssmartphone-revolution-9183/

[2] https://developer.android.com/reference/android/location/GnssMeasurement

[3] http://gpsworld.com/google-opens-up-gnss-pseudoranges/

[4] S. Banville, F. Van Diggelen, "Precise GNSS for Everyone: Precise Positioning Using Raw GPS Measurements from Android Smartphones", GPS World 27(11), November 2016. http://gpsworld.com/innovation-precise-positioning-using-raw-gps-measurements-from-android-smartphones/

[5] F. Van Diggelen, "GNSS Measurements Update", GSA Raw Measurements Workshop, Prague, 30 May 2018. https://www.gsa.europa.eu/sites/default/files/expo/frank_van_diggelen_keynote_android_gnss_measurements_update.pdf

Contents

LAB TASKS

TASK 1

Download the modified *Google's GPS measurement tool* from the portal and extract the content of the opensource folder.

TASK 2

Launch the main script and perform the analysis of raw measurements ("*gnssmeas*") and PVT solution ("gpsPvt") using the datasets provided on the portal. Inspect the output plots.
Optional: try some data filters.

TASK 3

Download the *GNSS logger app* from play store and retrieve new data collections (e.g. 5 minutes) using an Android device. Perform raw measurements and PVT analysis. Discuss the output plots with respect to the data collection conditions and compare them (e.g. open sky conditions? Were some line-of-sight signals obstructed? Battery saving mode?)

TASK 4

From the previous step retrieve the coordinates of your estimated position (median). Alternatively, if you can pin your true position on Google Maps, you can extract it from there. Use these coordinates to set a spoofed location not far from your original one.
E.g. `spoof.position = [trueLatutude trueLongitude trueAltitude] + 1e-3;`
What is the effect on the position? More generally, what effects can you notice on the tool outputs?

LAB TASKS

TASK 5

Now try a different spoofed position. For instance, pin a point on Google maps, not too far from the data collection location, extract the Latitude and Longitude coordinates (choose a reasonable altitude if you can't find it) and use them to set a value for `spoof.position`. Did you get what you expect? What did it change? By observing the plots, can you imagine some spoofing detection strategy?

TASK 6

Add a spoofing delay through `spoof.delay` (try in the order of milliseconds) without changing the spoofed position. Does the estimated position change? What changes can you notice? Why? What would be the consequences on a GNSS receiver's performance?

TASK 7

Optional: Repeat the data collection in peculiar and interesting conditions worth to be investigated. For example, near potential interference sources (e.g. broadcasting TV antennas, microwave ovens, phone calls, etc.) and inspect (if any) the effects on the GNSS observables. It would be ideal to perform also a reference experiment in nominal conditions without the interference. However, this is not always achievable.

Remember that you would need similar satellite conditions, e.g. similar experiment location and time (or time+k*24hrs, since the GPS satellites configuration over your head repeats every 24hrs).

TASK 8

Prepare a laboratory report following the report guidelines of the course. In the report, discuss **tasks 3, 5, 6**, and, optionally, **task 7**.

Get and Run GPS-MEASUREMENT-TOOLS

- Download the **enhanced** *Google's GPS measurement tool* **from the portal**

1. Unzip **gps-measurement-tools-master** and go inside **opensource** folder
2. Open main script: **ProcessGnssMeasScript.m** with MATLAB
3. Copy your log file(s) inside **~/opensource/demoFiles** (e.g. ~/opensource/demoFiles/myLogs)
4. Match your folder path by updating dirName (e.g. **~/opensource/demoFiles/myLogs**)

In case of issues in the use of the enhanced Google's gps-measurement-tools-master downloaded from the portal, you can consider using the more stable version available on Google's github (it may need some workaround with some smartphones models and it has no cyberspoofing enabled)

Inside the Main Script

- Main Script: *ProcessGnssMeasScript*
  - Launching the main script is enough to process raw measurements, perform the PVT, and plot the results

- Function to process the raw measurements and compute code Pseudoranges:
  - *[gnssMeas] = ProcessGnssMeas(gnssRaw);*
- Function to perform GNSS PVT:
  - *gpsPvt = GpsWlsPvt(gnssMeas,allGpsEph);*

Inside the Main Script

Input Parameters:

- File Name and Directory Name
- True Coordinate (if known, they are used to compute errors, otherwise leave blank)

```
prFileName = 'pseudoranges_log_2016_06_30_21_26_07.txt'; %with duty cycling, no carrier phase
% prFileName = 'pseudoranges_log_2016_08_22_14_45_50.txt'; %no duty cycling, with carrier phase
% as follows
% 1) copy everything from GitHub google/gps-measurement-tools/ to
%    a local directory on your machine
% 2) change 'dirName = ...' to match the local directory you are using:
dirName = '~/Documents/MATLAB/gpstools/opensource/demoFiles';
% 3) run ProcessGnssMeasScript.m script file
param.llaTrueDegDegM = [];
```

- **Try running the script with the demo log file first!**
- Example of true coordinates in LLA (latitude, longitude, altitude) frame: [45.0621, 7.6633, 295.070] … where is this location?

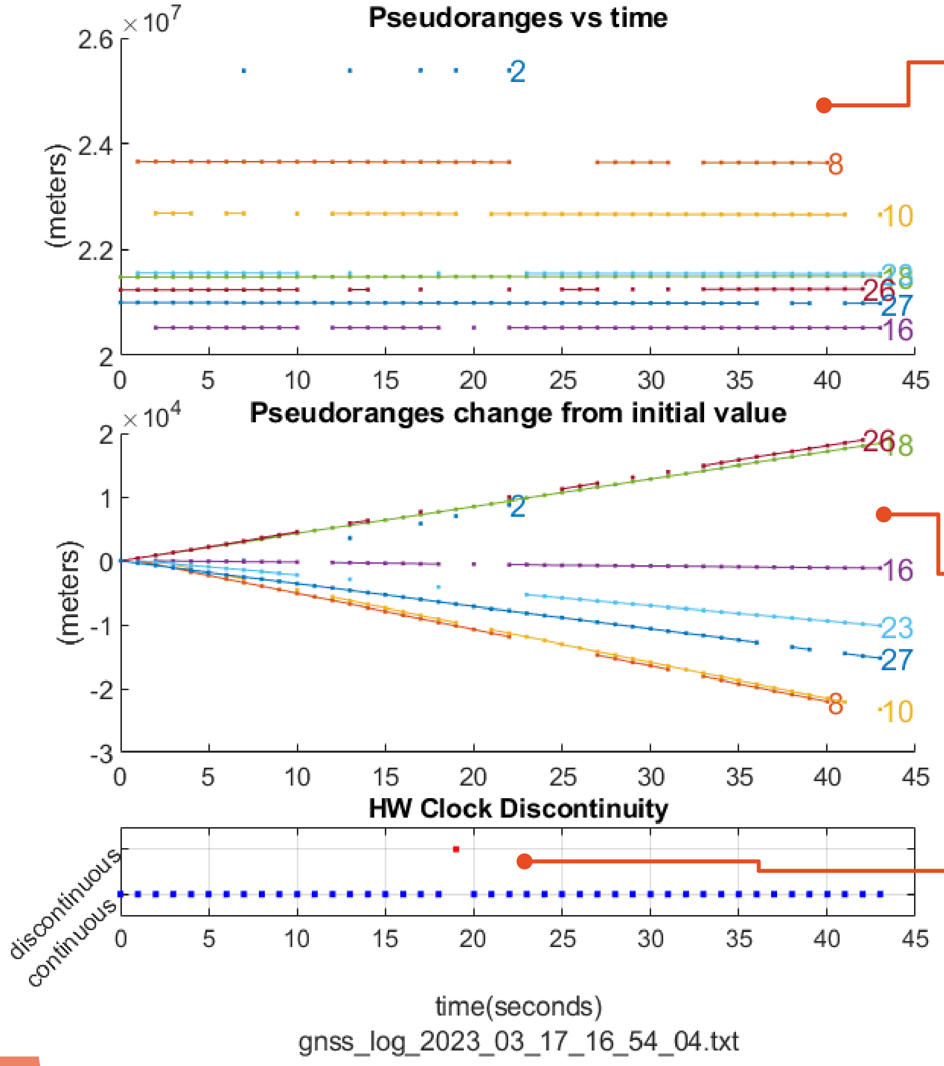
Inside the Main Script

- Spoofing input parameters

```
%% Spoofing settings
spoof.active = 1; % [1: spoofing active, 0: spoofing disabled]
spoof.delay = 40.212e-3; % [s] additional delay introduced by the spoofer [s]
spoof.t_start = 15; % [s] start spoofing time
spoof.position = [45.063454, 7.679441, 347.48]; % spoofed position
```

- The effect of a spoofing attack is emulated acting on the measurements extracted from your data collection (see README for details).
- Setting the `spoof.delay` emulates re-broadcasting after a delay + relative distance between spoofer and victim.
- Setting the spoofer position emulates re-broadcasting from `spoof.position` or, equivalently, the desired counterfeit location induced
- The net result is like a cyberspoofing attack

How to perform an Analysis: MATLAB analysis tool

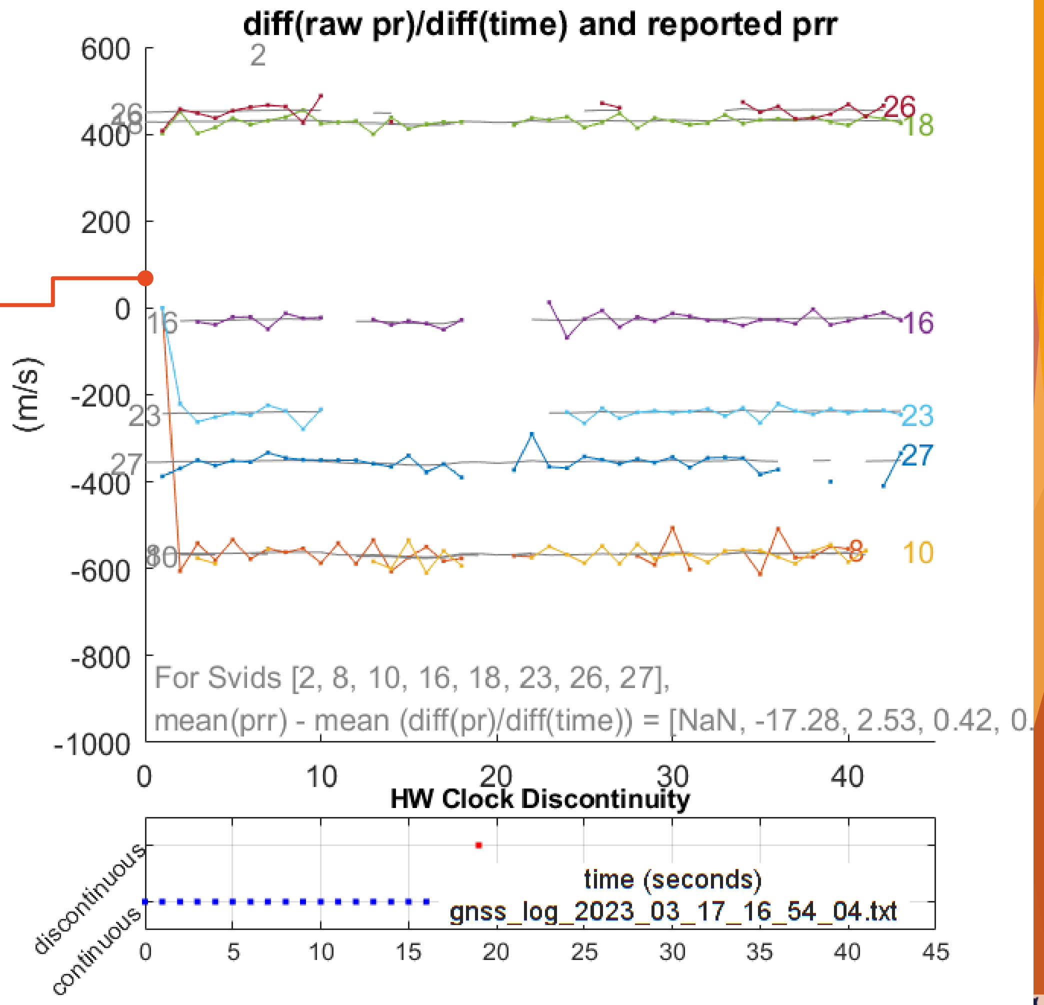


- Pseudorange (PR) measurements for each tracked satellite
- Variations over time are hard to see in this plot

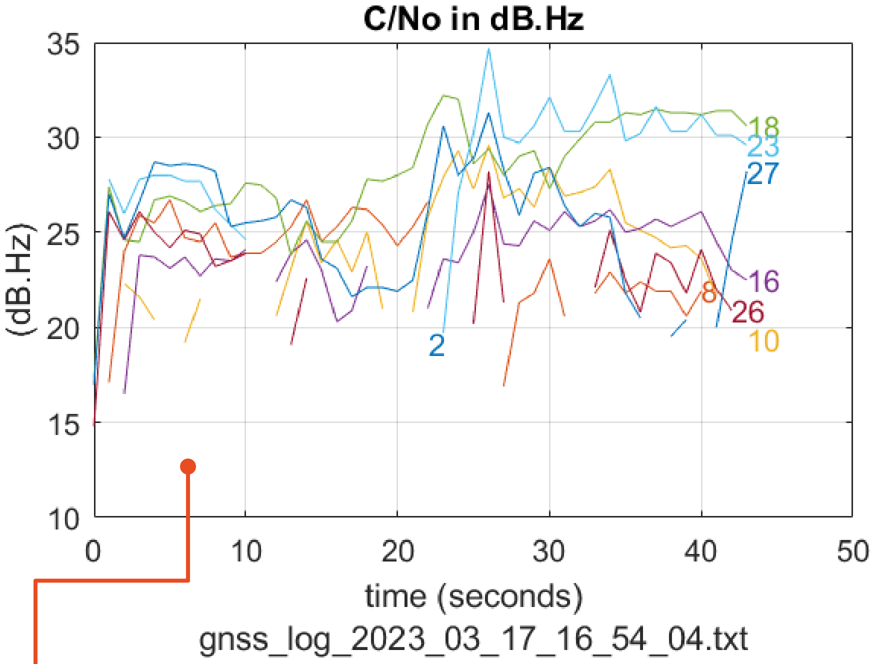
- Derivatives can highlight sudden PR «jumps»
- ***How are the two lines computed? What is their difference?***

- Visualization of PR variations over time
- A decreasing PR means the sat is approaching the receiver and vice versa

- User clock is sometimes turned off for battery management
- This will cause a growing user clock bias w.r.t. the GNSS time
- ***Can clock discontinuity affect the PR measurements?***

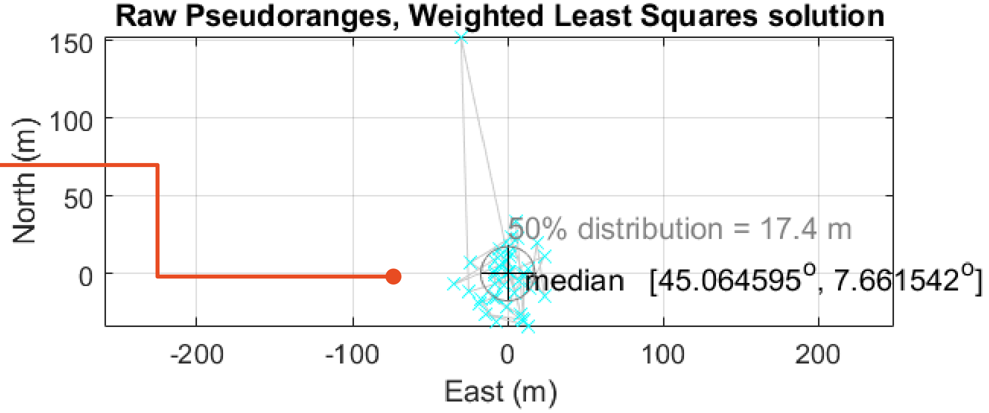


How to perform an Analysis: MATLAB analysis tool



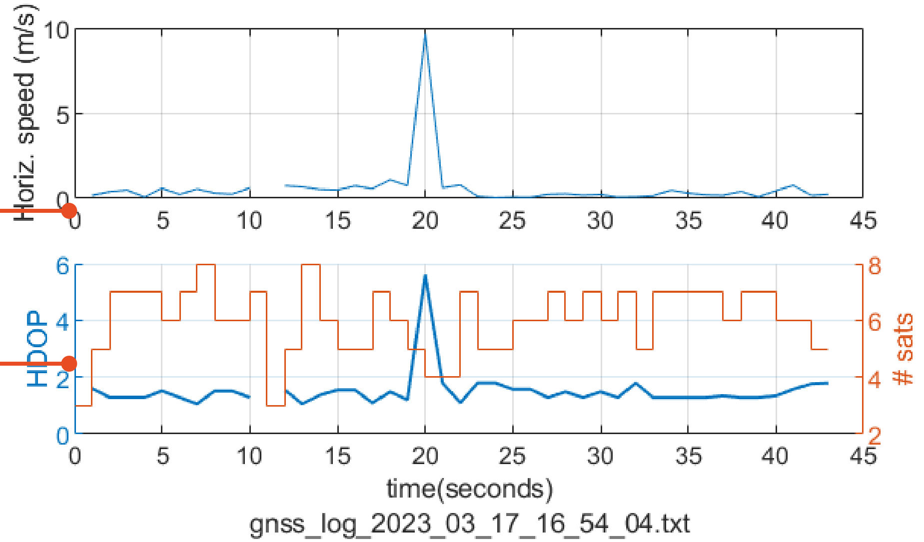
- CN0 values tell you what are the strongest (and weakest) signals and therefore the potentially more (and less) accurate measurements

- If `true location` has been set, it is shown alongside the median
- Outliers can be identified, but not easily mapped to time
- The circle contains the best 50% of the estimates (the larger the circle, the less precise the estimation)
- ***What happens to the median if you are moving?***

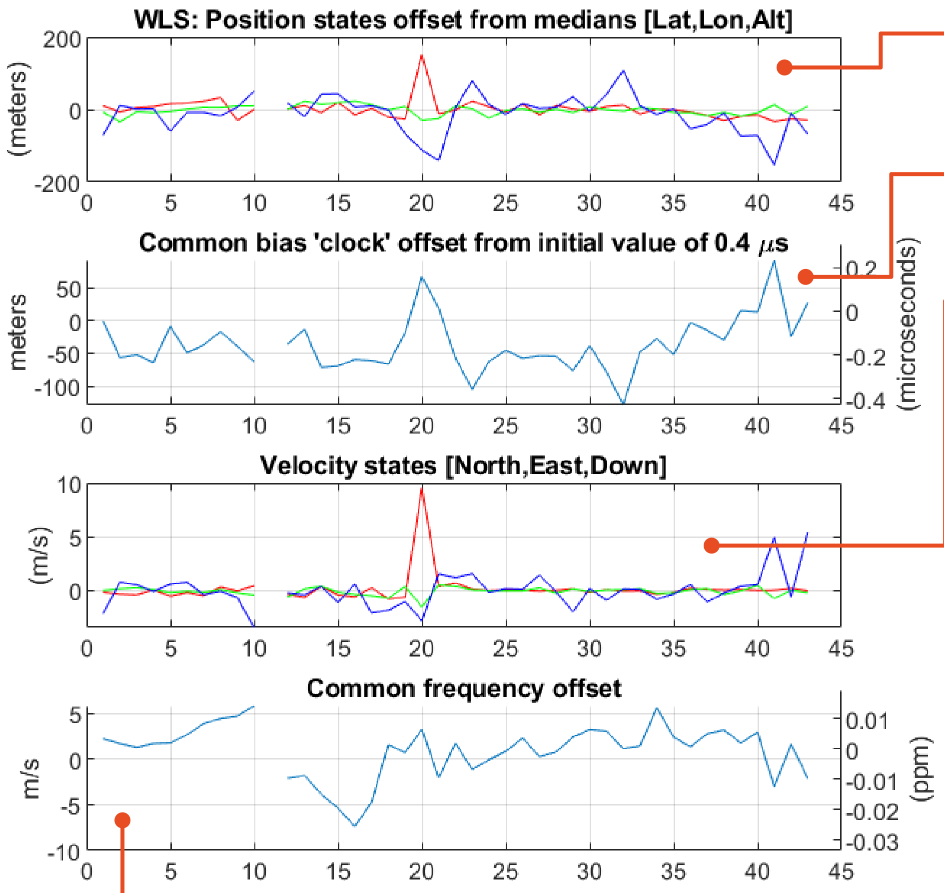


- Horizontal means over a plane on the Earth surface
- Even a static user might have oscillations (it's an estimate!)
- ***How is the speed calculated?***

- Horizontal dilution of precision (HDOP) is a measure of how satellite geometrical distribution around the user impacts the estimated position
- The smaller, the better. A large HDOP can cause large PVT estimation errors
- HDOP is closely related to the number of tracked satellites: with less satellites in view, it is more likely to experience bad geometrical conditions



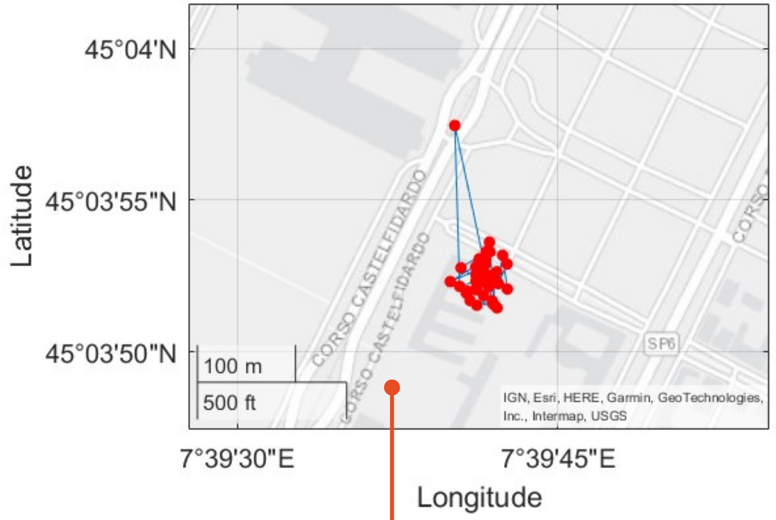
How to perform an Analysis: MATLAB analysis tool



- If the median is a good estimate of the true position, these curves are an indicator of the estimation errors
- They are useful to observe errors through time and correlate them to timed events

- The user clock bias is an important indicator of biases that are common to all the pseudoranges…
- Represented in meters (using the speed of light)

- Velocity estimates are shown w.r.t. a more convenient reference frame (NED)



- This is the user clock bias drift

- *Geoplot* visualization conveniently shows the estimated position over a map

Inside the Main Script

- Data Filter:
  - *dataFilter = SetDataFilter;*
  - Use some default filters by uncommenting them.
  - Warning: Google's MATLAB tool functionalities for constellations other than GPS are limited (better to avoid this filter)
  - *Example: Exclude specific satellites*

```
dataFilter{end+1,1} = 'Svid';
dataFilter{end,2}   = 'Svid ~= 4';
```

  - Write your own custom filter.
  - *Example: C/N0*

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

Inside the Main Script

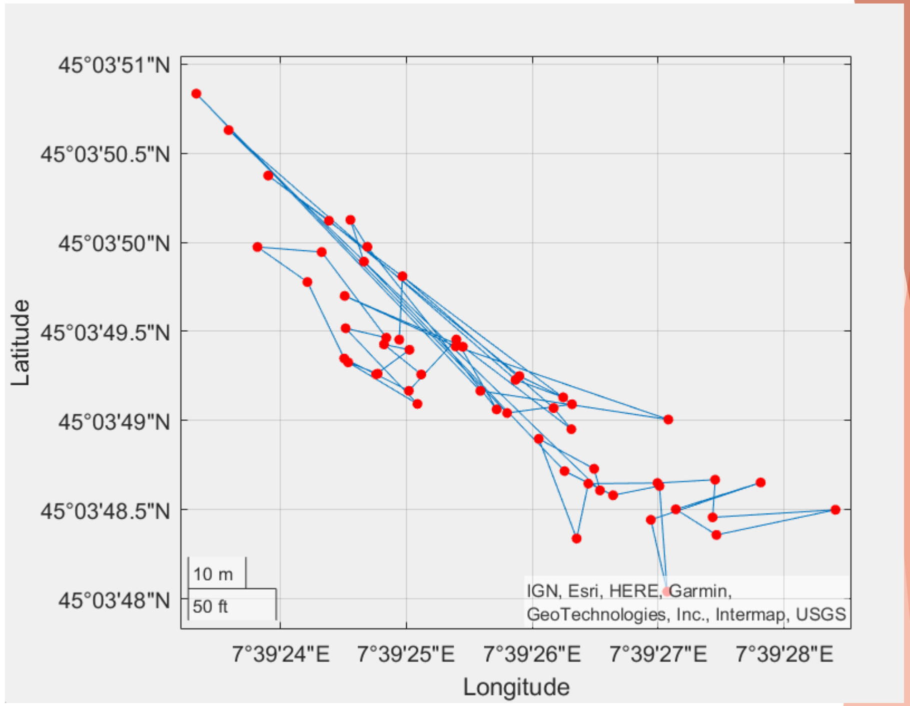
- Acquiring the Ephemeris:

```
%% Get online ephemeris from Nasa ftp, first compute UTC Time from gnssRaw:
fctSeconds = 1e-3*double(gnssRaw.allRxMillis(end));
utcTime = Gps2Utc([],fctSeconds);
allGpsEph = GetNasaHourlyEphemeris(utcTime,dirName);
if isempty(allGpsEph), return, end
```

After running the script for the first time, unzip the downloaded ephemeris in “demoFiles” folder, if automatic unzip does not work

Report Guidelines

- Abide by the rules outlined in the course intro slides
- Keep it short (no more than 4 pages). Select only the **meaningful plots** that are relevant with respect to your comments (focus on your own data collection)
- In the report it's crucial to provide a clear and concise overview of the experiment's **purpose, methods, results, and conclusions**. However, we are not interested in a mere repetition of the theoretical concepts from classes, rather we are interested into an insightful analysis of the results.
- If you make qualitative assessments, they must be substantiated.
    - For instance, avoid commenting a plot like the one in the figure claiming that the estimate is «not accurate». Rather, measure its dispersion (e.g. variance, standard deviation, root mean square) and/or compare it with other data collections



Report Guidelines cont'd

- Prepare a report of a maximum 4 pages (plus applendix) using the LaTeX conference-style format for ACM CCSthat you can find on www.overleaf.com.
  - Overleaf is a free online editing system for LaTeX. It allows collaborative editing, with a simple interface and visual editor based on LaTeX, the most used markup system to describe scientific documents.
  - LaTeX is a markup system in the early 1980s by Leslie Lamport who extended the original TeXfirst released in 1978 by Donald Knuth. Since then, it has been the de facto standard for producing scientific documents. It lets you focus on the content and not on the pagination. There are plenty of LaTeX tutorials, e.g., the Overleaf help pages.
- You need to create a free account on Overleaf, then create a new document by looking for the Sample ACM CCS template.
- Next, modify the document format to
  - use the "review" option: `\documentclass[sigconf, review]{acmart}`
  - change authors, title, etc.
  - remove the copyright part to save space:
    - `\setcopyright{none} %remove the (c) section`
    - `\acmConference{Wireless Security Report}{Torino}{2024} % set a possibly significant conference name`
    - `\acmPrice{} % leave the rest empty. These will still appear.`
    - `\acmISBN{}`
    - `\acmDOI{}`
- You can share the document using the Share with link option so that anyone with the link can collaborate to edit the file.

Report Guidelines cont'd

Report preparation: Your lab report should follow a scientific paper structure and be written in a clear, concise, and technical style. The report should demonstrate your understanding of GNSS signal collection and the effects of spoofing (<u>without repeating the theoretical lecture</u>), as well as your ability to analyze and interpret data.

You can organize your report according to the following outline (<u>feel free to modify</u> it):

- 1. Introduction
  - Purpose of the experiment: Briefly explain what the lab is about and why it's important.
  - Objectives: Clearly state the goals of the experiment (e.g., to collect raw GNSS data using an Android device, and to analyze the effects of spoofing on positioning or observations quality).
- 2. Methods
  - Devices and tools used: List and describe the Android devices, apps (e.g., GNSS Logger), spoofing tools, and any post-processing software.
  - Data collection procedure:
    - How and where the GNSS data was collected (e.g., open sky, indoors, under spoofing conditions).
    - Settings used during data logging (e.g., battery saving enabled, logging duration).
  - Spoofing setup: Describe how spoofing was introduced and its settings/characteristics
  - Data analysis: Explain how you processed or visualized the data (e.g., pseudorange analysis, position error, statistical analysis).
- 3. Results and Discussion
  - Data presentation:
    - Include plots, tables, and summaries of key results. Try to follow lab tasks and report outputs (plots, tables, etc) when relevant for the discussion.
    - Highlight differences between spoofed and unspoofed data.
  - Interpretation:
    - Discuss what the data shows about the quality of GNSS measurements and PVT under different conditions.
    - If spoofing was performed, explain its effect on position estimates or signal metrics.
    - Mention any anomalies, unexpected results, or challenges encountered.
  - Limitations:
    - Discuss any sources of error or limitations in your setup or data interpretation.
- 4. Conclusion
  - Summary: Recap the main findings in concise sentences.
  - Insights gained: Reflect on what the experiment reveals about GNSS systems and spoofing vulnerabilities.
  - Future work: Briefly suggest potential improvements or extensions for this kind of experiment.
- Formatting Notes:
  - Include figures and tables with appropriate captions and references in the text.
  - Cite any tools, apps, or relevant literature you used or referred to.
  - Use technical terminology appropriately but define any concepts that are not commonly known (or reference proper sources)

Inside the MATLAB Code: Glossary

```
allRxMilliseconds:     Milliseconds at which the measurements are dumped (loca time);
gnssMeas.FctSeconds:   Fetch Time in Seconds (local time);
N:        Number of fetch seconds (measurements records and PVT if everything goes well;
gnssMeas.ClkDCount:    zeros(N,1);
gnssMeas.HwDscDelS:    zeros(N,1);
gnssMeas.Svid:     all the sv ids found in gnssRaw
M:        Number of unique Svid
gnssMeas.AzDeg:    Satellites Azimuth Degrees
gnssMeas.ElDeg:    Satellites Elevation Degree
gnssMeas.tRxSeconds:   Time of reception, seconds of GPS week
gnssMeas.tTxSeconds:   Time of transmission, seconds of GPS week
gnssMeas.PrM: PseudoRange Measurements;
gnssMeas.PrSigmaM: Pseudorange Standard Deviation;
gnssMeas.DelPrM:   DeltaPseuodrangeMeasurements;
gnssMeas.PrrMps:   Pseudorange Rate;
gnssMeas.PrrSigmaMps:  Pseudorange Rate Standard Deviation;
gnssMeas.AdrM:     Accumulated Delta Range Measurements;
gnssMeas.AdrSigmaM:    Standard Deviation of the Accumulated Delta Range Measurements;
gnssMeas.AdrState: Accumulated Delta Range State;
gnssMeas.Cn0DbHz:  C/No;
```

LAB troubleshooting

- In case of issues with **ephemeris extraction** (i.e. getting satellite positions from internet), we recommend a manual extraction of the ephemeris data retrieved from NASA's CCDIS service, matching the date and time of your data collection.
- Files logged through the GNSSLogger App v3.0.0.1 might break the code depending on the model of your device, chipset, or API version
- **Old logs** available in dataset_a and dataset_b might require old-format ephemeris that cannot be automatically decompressed. In case you want to use them, please, extract them manually after the download
- *GeoPlot* and other functions could be not available for old or basic MATLAB installations. MATLAB will warn you about the **toolbox needed**, and you can add it to your installation a posteriori, using your Mathworks license. Geoplot is part of the *Mapping Toolbox.*

- In general, try to apply your engineering skills to fix the issue first. If you cannot solve it after reasonable effort, the instructor is here to help.