GNSS Lab Report

Wi-Failing
Ciriaco Alifano, Gabriele Camisa, Simone Giammusso, Matteo Rizzo

1 INTRODUCTION

The goal of our experiments was to collect GNSS data by means of Google's GNSSLogger Android app and then analyze them to see how different conditions impact signal integrity and positioning accuracy. To do so, we analyzed both raw measurements and the PVT solutions found using the method of WLS [18].

We started with data collection in open-sky conditions and later simulated a cyber-spoofing scenario where we alter pseudoranges and apply artificial time delays using the same MATLAB tools. Ultimately, we studied how biological materials attenuate GNSS signals, by using pork meat as a proxy for human tissue.

2 TOOLS AND METHODS

- **Device:** Samsung Galaxy S23 Ultra smartphone, running Android 15, equipped with the Snapdragon 8 Gen 2 Qualcomm SM8550-AB chipset, which supports dual-frequency, multi-constellation GNSS (GPS, Glonass, NavIC, Beidou, Galileo, QZSS). [10]
- **GNSS data collection:** "GNSSLogger" app, version 3.1.0.4 [7] | "GPSTest" app, version 3.10.5 [9] | "Google Earth" app, version 10.79.0.3. [8]
- **Geoid height calculator:** GeographicLib web app. [12]
- **Data analysis:** MATLAB R2024b [14] | gps-measurement-tools suite [6], developed by Google, enhanced by the NavSAS research group of Polytechnic of Turin [15] | "Location Based Analysis of Visible GPS Satellites" example notebook from MATLAB Satellite Communications Toolbox. [13]

Every log contains 5 minutes of uninterrupted data sampled at [45.047208 N, 7.655716 W, 250 m a.s.l.] (Parco Cavalieri di Vittorio Veneto, Turin, Italy) (fig. 14), isolated approximately 20 meters from nearby obstructions and under fair weather conditions, ensuring minimal multipath interference and optimal reception. All user-space power-saving options were turned off and the device was put in Airplane mode during logging.

3 OPEN-SKY CONDITIONS

3.1 Raw measurements - HW clock de-sync

During our first experiment, which we conducted under all conditions specified in 2, we observed hardware clock discontinuities that directly affected the reliability of pseudorange computations.

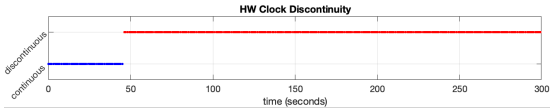

Figure 1: Status of GNSS hardware clock over time with just GNSSLogger

Although no power-saving modes of any kind were active in the mobile device, HW clock discontinuities, indicated by a transition from blue (continuous) to red (discontinuous), were detected at approximately 46 seconds into the logging session and persisted for the duration of the test (fig. 1). This discontinuity indicates that the device's internal clock experienced a sudden and irregular change in its progression. Instead of increasing smoothly over time, the clock may have jumped forward, backward, or altered its rate. This issue is not caused by a loss of synchronization with GNSS time, which is always managed through a clock bias, but rather by a sudden instability in the internal device clock that breaks the assumption of consistent bias over time. As a result, time-dependent computations, such as speed estimation or pseudorange rate, become unreliable from the point of discontinuity onward.

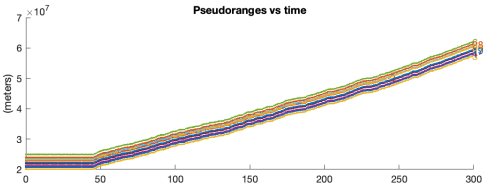

Figure 2: Pseudoranges vs. time under HW clock discontinuity

As shown in Figure 2, after the clock discontinuity occurs, all pseudorange values begin to increase steadily and appear very similar across all satellites. This unusual pattern is caused by an incorrect handling of the clock bias during the computation of the GNSS receive time (*tRxNanos*).

In the script, *tRxNanos* is calculated using the expression:

$$\text{tRxNanos} = \text{TimeNanos} - \text{FullBiasNanos}(1) - \text{weekNumberNanos}$$

where *TimeNanos* is the internal time of the device at the moment of signal reception, *FullBiasNanos* is the correction term that accounts for the difference between the device's clock and true GNSS time, and *weekNumberNanos* adjusts for GPS week rollovers.

However, the script applies a constant *FullBiasNanos* value across the entire dataset, using the one extracted from the first epoch (*FullBiasNanos(1)*). In reality, the clock bias is not constant: it can drift over time or change abruptly in the presence of a clock discontinuity. When this happens, *TimeNanos* may suddenly jump, but since the bias is not updated accordingly, it no longer compensates correctly. This causes *tRxNanos* to become inaccurate from that point on, introducing a common shift in the receive time for all satellites. This has a significant effect on the pseudorange calculation, which is defined as:

$$\text{pseudorange} = (tRx - tTx)c$$

where *tRx* is the receiver time computed as 3.1, *tTx* is the transmission time from the satellite and c is the speed of light. When the signals from the satellites arrive at the same time, *tRx* will be identical for all of them. In this case, the differences in the pseudoranges are determined solely by the *tTx*, the timestamp of each satellite's signal transmission. However, even a small error in *tRx*, in the order of microseconds (as showed in 19), can result in a large pseudorange error because it is multiplied by the speed of light. As a result, the pseudoranges may appear almost identical in the graph due to this large error, which grows over time.

Despite these errors in PR computation, the estimated positions remain approximately correct. This is because the weighted least squares solution simultaneously estimates not only the receiver's position and velocity but also the clock bias as an unknown parameter. As a result, even if the initial bias used in the pseudorange computation is outdated or incorrect, the solver adjusts the bias dynamically during the estimation process. However, since the initial PR inputs are distorted by the uncorrected bias, the system starts from less precise measurements, which can slightly degrade the accuracy of the estimated position and velocity (fig. 3).

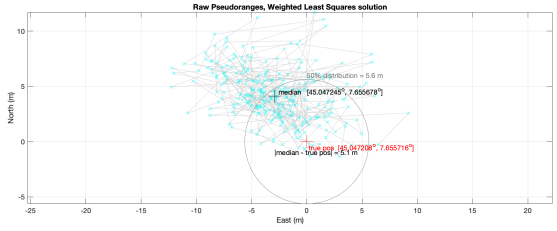

Figure 3: Computed position with discontinuity clock vs true position

3.1.1 Solving the HW clock discontinuity. While the WLS algorithm partially compensates for bias errors, a better solution was found by directly preventing the discontinuities from occurring. In particular, launching the GPSTest [9] app in parallel with GNSSLogger during data collection sessions eliminated HW clock discontinuities (fig. 20).

This behavior suggests that the root cause is not related to external environmental factors, but rather stems from the internal management of GNSS resources on the Android platform, like foreground priority and wake locks, leading to clock resets or timing irregularities. By forcing continuous GNSS activity through GPSTest, this behavior is suppressed entirely, and the hardware clock remains stable throughout the session.

In order to obtain better results, without discontinuity, all subsequent surveys were carried out using the two applications, GNSSLogger and GPSTest, run in parallel.

3.2 Raw measurements - no HW clock de-sync

Having solved the HW clock de-synchronization issue, the logging produced the best samples we had, which we will use as a baseline to analyze worse scenarios. Correlating the logs with the visibility skyplot produced by the MATLAB notebook [13] (fig. 4), we can see how the phone's receiver has been continuously able to receive signals from all visible satellites, except for satellites 20, 26 and 30, which were possibly out of sight because of their low elevation angle of less than 10 degrees above the horizon. In addition, HDOP has been stable at a value of 0.8 (fig. 22), which indicates very good geometrical conditions.

The C/N0 plot (fig. 5) shows how the intensities of signals from visible satellites have no immediate correlation with the elevation angle; in particular, we see how satellites closer to our Zenith and closer in terms of straight-line distance, as shown by the pseudoranges (fig. 23), are not among the strongest signals we received. Finally, some signals were surely more stable in their intensity than others; examples of this behavior are satellite 4 (very stable at more than 50 dB-Hz) and satellite 7 (oscillating in a range between 30 and 45 dB-Hz).

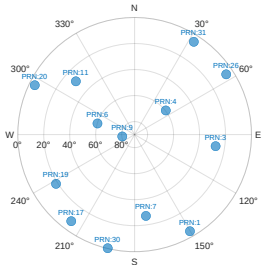

Figure 4: Skyplot of visible satellites at 20:16:32 UTC+2

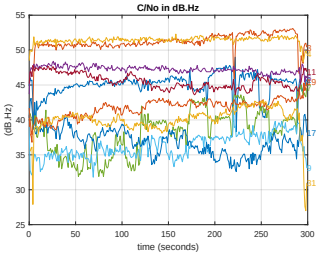

Figure 5: Carrier to noise density ratio, dB-Hz

3.3 Position, Velocity and Time

The open-sky conditions are also reflected in the quality of the results obtained in terms of position, velocity and time estimation. We obtained very stable results; the 50% computed positions were within a circle of radius of 1 m (fig. 15), with an error of 5.4 meters between the median and the real position (on the horizontal plane) (fig. 16).

However, comparing the estimated altitude with the actual one (obtained via Google Earth), we observed a significant discrepancy, not only due to the precision of the GNSS system, but also to the difference in models used to represent the shape of the Earth [4] [3]. In particular, Google Earth uses the EGM96 geoidal model for the planet [17] [5], and therefore refers to an orthometric height, while the script we used relies on the WGS84 ellipsoid model, and outputs an ellipsoid height. The geoid height for the specific location is easily retrieved by using online tools [12].

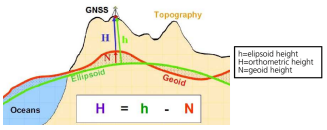

Figure 6: Orthometric and ellipsoid height, taken from [11].

In our case, the median altitude reported by the MATLAB tool (h) was 321 m, the one obtained from Google Earth (H) was 250 m and the geoid height in our location (N) is 48 m, so the error in the altitude estimation is $h - (H + N) = (321 - 250 - 48)$ m $= 23$ m, which aligns with our expectations about the vertical accuracy [2].

4 CYBER-SPOOFING SIMULATION

4.1 Altering pseudoranges

As a first experiment with our cyber-spoofing emulation, we operated on the logs collected in 3. We chose to spoof our location 1 Km away from the real one, at the same altitude, of coordinates [45.0487414 N, 7.6682227 W]. The spoofing process computes, for each satellite in sight, the difference in the pseudorange that would be observed between the true location and the spoofed location; it then derives satellite-dependent time delays, which it adds to each log entry from that satellite. In our setting, spoofing starts at $t = 150$ s, exactly in the middle of the logs collection period.

4.1.1 Effects on raw measurements. We were expecting both a sudden jump in the pseudoranges and a visible spike in the middle of the PRRs (when computed at runtime as a ratio of deltas, rather than being parsed from the logs). However, we observed that plotting PR

amplitudes over time is not enough to visually notice the introduced difference while visualizing the behavior of more curves at once.

Instead, considering only how the PR changes from their initial values, the jump is more evident. In fact, given that the receiver was standing still while collecting GNSS data, the plotted curves will be straight lines with a fairly stable slope while spoofing is not active, due to the natural movement of the SATs. By plotting only the slopes of those curves over time, which are a fair approximation of the PRRs, we see how the SAT-specific deltas added to the pseudoranges translated into a visible spike in each of the curves at $t = 150$ s (fig. 7). This pulse-shaped alterations are far more readable to the eye and at the same time suggest that the absolute modifications over the pseudoranges were rather small.

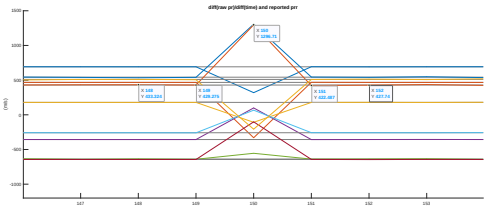

Figure 7: Pseudorange rates computed as ratios of deltas, or slopes of PR change from initial value over time. Tooltips show data about satellite 19.

Let's focus on SAT 19. Its pseudorange was about $2.25 \cdot 10^7$ m during the whole logs collecting operation (fig. 23), and it was one among the most affected satellites by our spoofing. Its PR was also naturally changing due to the satellite moving, with a slope of about 430 m/s; the spoofing introduced a spike of about 1300 m/s at $t = 150$ s.

We can conclude that to trick the positioning algorithm into a location that was 1 Km away from the real one, each pseudorange has been altered by at most 0.00004% of the value they had a second before the spoofing started. This really enhances our perspective on how the GNSS receivers rely on the smallest variations in the signal reception times to produce their outputs.

4.1.2 Effects on PVT solution. While the most obvious effect of the spoofing simulation is the sudden change in positioning (fig. 24), the effects on the velocity states surely deserve attention. In fact, it's clear how velocities do not correspond to the ratios between the point-to-point distance and the time between two epochs, otherwise the velocity plots would present a visible spike in the middle, just like the plot of pseudoranges rates (fig. 7).

Instead, velocities are being computed by exploiting the Doppler shift phenomenon [1], because of which the signals from SATs are perceived by the receiver at slightly different frequencies, depending on the relative motion between the SAT and the receiver. Our receiver catched these subtle frequency shifts and autonomously derived PRRs (which here are to be interpreted as relative velocities between the receiver and the specific SAT) that we can find in the log entries (fields *PseudorangeRateMetersPerSecond* and *PseudorangeRateUncertaintyMetersPerSecond*).

The MATLAB tool [6] we used for the analysis has exploited those values, which our spoofing algorithm did not modify in any way, to estimate the velocity of the receiver; in particular, the WLS solver had to compensate the fact that in our new position we still had the previous relative velocity with respect to each of the SATs, which it did by tweaking the velocity estimates in both amplitude, direction and variance. This behavior justifies what we see in fig. 8.

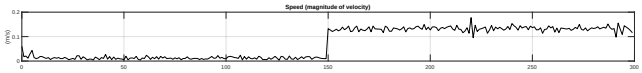

Figure 8: Velocity states over time after spoofing

4.2 Spoofing detection

During the various tests we conducted, we made an attempt to identify methodologies for detecting ongoing cyber-spoofing activities.

4.2.1 Spikes in the runtime-computed pseudorange rates. From the analysis of the pseudorange rates graph, not only we see how the computed PRRs differ from the logged ones (the gray curves), but we also observe distinctive spikes, which occur when the PR value changes abruptly from one epoch to the next (fig. 7). When the spoofed location is far enough, the magnitude of the spike stands out clearly from the background noise, facilitating detection.

Moreover, since our spoofing technique involves adding deltas to all visible satellites starting from a selected timestamp, the spikes present themselves simultaneously, which allows to distinguish the spoofing event even in the case of spikes of lower amplitude.

4.2.2 Inconsistent velocity states. As previously mentioned, the spoofing emulation introduces anomalies in the velocities (4.1.2).

For instance, the sudden increase in speed that the WLS solver introduces when the spoofing starts may not be wide enough to justify the position shift. In our example, the spoofed position was 1 Km away from the real one, but velocity peaked at less than 0.2 m/s, instead of showing a spike reaching 1 Km/s in $t = 150$ s (which would have have been unrealistic nonetheless, in terms of acceleration) (fig. 8).

A more complex cyber-spoofing strategy, which would also alter the pseudorange rates, would probably cancel this behavior, but is also likely to be detected by searching for sudden changes in the PRRs themselves, or by correlating such values to data taken from other sensors the system is equipped with, such as accelerometers.

4.3 Adding time delay

Starting the spoofing attack at $t = 150$ s and using the previously spoofed position, we set the variable *spoof.delay* = 0.003, meaning a 3 ms delay. Essentially, the script now not only will modify the signal reception times by adding SAT-dependent delays, but also simulate that they all were received 3 ms later than they actually were. The change in delay creates a scenario very similar to a meaconing attack. In fact, the only difference here is that we are applying an additional common delay to all the satellite signals, that will be incorporated into the receiver's clock bias. This bias will be estimated and accounted for during the calculation by the WLS solver. The resulting change in the bias due to this delay is evident in figure 9.

Figure 9: Common user clock bias after spoofing with additional delay

Since this delay change only affects the clock bias, the position will still be calculated correctly. However, unlike the simple spoofing attack, where a very small change in pseudoranges could falsify the position by 1 km, we now observe a larger shift in the pseudoranges, which take on much larger values. This behavior, which we discussed in the first HW clock discontinuity scenario, is due to the growing clock bias, now caused by the delay, which wasn't properly corrected in the graph (fig. 10).

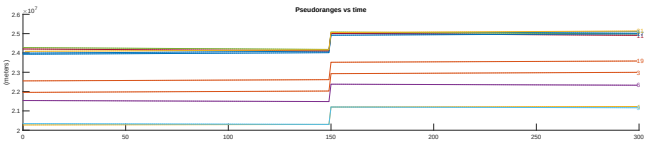

Figure 10: Pseudoranges vs. time, after spoofing with additional delay

4.4 Interference scenario

4.4.1 Can subdermal GPS microchips enable human tracking? The feasibility of receiving GNSS signals through human skin is severely limited due to the signal's extremely weak power and the radio-frequency (RF) attenuation properties, especially in the L-band, of biological tissue.

GNSS signals are very weak and require a line-of-sight to satellites. Human tissue, with its high water and organic content, strongly attenuates these signals such that subdermal reception is extremely unlikely without an external antenna.

4.4.2 Using minced pork to simulate human tissues for GNSS signal attenuation. We employed minced pork as a biological analog to human tissues to investigate the attenuation of GNSS signals in subdermal environments. Porcine meat is widely recognized in biomedical research as a suitable surrogate for human tissues due to its comparable physiological and dielectric characteristics [16]. In addition, it offers a homogeneous medium that facilitates consistent and reproducible measurements. However, while it simplifies experimental conditions, this model lacks the stratified structure of natural skin, which may influence the attenuation characteristics compared to intact tissues.

4.4.3 Scenario analysis. For this experiment, we replicated the same conditions previously described in 3, maintaining a continuous clock signal without discontinuities. The measurements were repeated the following day at the same time (20:16:32 UTC+2), this time surrounding the device [10] with a 2.5 cm thick layer of minced pork to simulate subdermal tissue.

To evaluate the effect of biological interference on GNSS signal quality, we compared the Horizontal Dilution of Precision (HDOP) and the number of tracked satellites across two scenarios. In the interference scenario (11), HDOP values fluctuate noticeably, ranging approximately from 0.9 to 1.4, and the number of SATs varies frequently between 6 and 9. These fluctuations indicate unstable satellite geometry and occasional signal degradation, likely due to the RF attenuation properties of the biological tissue. The increased HDOP and reduced satellite availability correlate with potential deterioration in positional accuracy, consistent with expected GNSS performance under obstructed conditions. In contrast, in open-sky conditions, the GNSS receiver can maintain high positioning accuracy with minimal dilution effects, as already discussed in (3).

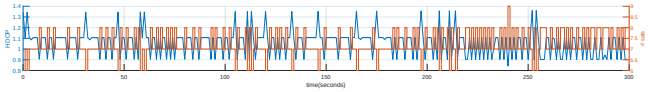

Figure 11: HDOP and SATs number in the interference scenario

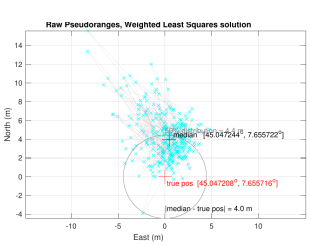

Figure 12: Positioning in the interference scenario

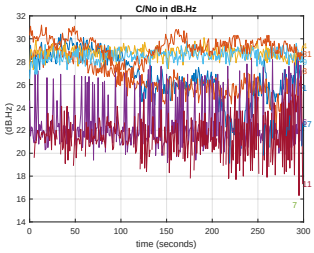

Figure 13: C/N0, dB-Hz in the interference scenario

Figures 5 and 13 show the Carrier-to-Noise Density Ratio (C/N_0) measurements under the two conditions, a key indicator of GNSS signal strength and quality, with higher values correlating to better measurement reliability and accuracy. In the interference scenario, most satellite signals exhibit significant attenuation, with C/N_0 values ranging predominantly between 20–30 dB-Hz. The lower and more fluctuating values suggest substantial signal degradation caused by the dielectric absorption and scattering effects of biological tissue. In contrast, in the non-interference scenario, C/N_0 values are consistently higher and less variable, mostly within the 35–50 dB-Hz range, indicating robust and stable signal reception.

The observed differences between the two conditions provide clear experimental evidence that biological tissue attenuates GNSS signals, reducing the number and the geometry of usable satellites and potentially compromising positioning accuracy.

In Figure 12, we observe an interesting outcome: although measurements are generally more precise in the open-sky scenario (fig. 15), the results in the interference setting show a median only 4 m away from the real position. This may be due to a sort of filtering effect of the biological material, which limits the number of usable satellites to the ones that had a stronger and more stable signal.

However, this apparent gain in accuracy does not correlate to a better precision (fig. 25), as the various position states appear more scattered (see also fig. 18), suggesting that biological tissue introduced biases due to signal weakening and poor satellite geometry (HDOP). Therefore, in applications requiring both high accuracy and reliability, biological interference must be considered a non-negligible source of error, reinforcing the impracticality of using unassisted subdermal GNSS devices for precise human tracking.

5 CONCLUSION

Our experiments revealed how GNSS signal quality and positioning accuracy rely on hardware conditions as well as outside interference. We successfully emulated cyber-spoofing attacks, at the same time demonstrating both its effects and its limitations.

Our last experiment showed the substantial impact of biological attenuation on signal reception and highlighted the distinction between accuracy and precision in GNSS systems.

REFERENCES

[1] Department of Geography, Pennsylvania State University. 2025. Doppler Shift — GEOG 862: GPS and GNSS for Geospatial Professionals. https://www.e-education.psu.edu/geog862/node/1786. [Online; accessed 12-May-2025].

[2] Per Enge and Pratap Misra. 2010. *Global positioning system.* Ganga-Jamuna Press.

[3] Eos Positioning Systems. 2025. Elevation for Beginners: Orthometric Height, Ellipsoid Height, and Geoid Height. https://eos-gnss.com/knowledge-base/articles/elevation-for-beginners/. [Online; accessed 12-May-2025].

[4] Esri. 2025. Mean Sea Level, GPS, and the Geoid. https://www.esri.com/about/newsroom/arcuser/mean-sea-level-gps-geoid#:~:text=The%20accuracy%20of%20GPS%20height,two%20hardly%20ever%20match%20spatially [Online; accessed 08-May-2025].

[5] Google Developers. 2025. Google Earth Model Source (Google Groups). https://groups.google.com/forum/?fromgroups=#!msg/kml-support-advanced/SL82unzyOfc/yuGWIpHa8rcJ [Online; accessed 08-May-2025].

[6] Google Developers. 2025. gps-measurement-tools Codebase on GitHub. https://github.com/google/gps-measurement-tools [Online; accessed 08-May-2025].

[7] Google Play Store. 2025. GNSSLogger on Google Play Store. https://play.google.com/store/apps/details?id=com.google.android.apps.location.gps.gnsslogger [Online; accessed 08-May-2025].

[8] Google Play Store. 2025. Google Earth on Google Play Store. https://play.google.com/store/apps/details?id=com.google.earth [Online; accessed 08-May-2025].

[9] Google Play Store. 2025. GPSTest on Google Play Store. https://play.google.com/store/apps/details?id=com.android.gpstest [Online; accessed 08-May-2025].

[10] GSMArena. 2025. Samsung Galaxy S23 Ultra Specifications. https://www.gsmarena.com/samsung_galaxy_s23_ultra-12024.php [Online; accessed 08-May-2025].

[11] Jānis Kaminskis, Marita Koroļeva, Guntars Vaivads, Rūsiņš Cēbers, and Aleksandrs Jirgensons. 2008. Determination of Quasi-geoid as Height Component of the Geodetic Infrastructure for GNSS-Positioning Services in the Baltic States. *Geodesy and Cartography* 34, 4, 123–130. https://www.researchgate.net/publication/233968093_Determination_of_Quasi-geoid_as_Height_Component_of_the_Geodetic_Infrastructure_for_GNSS-Positioning_Services_in_the_Baltic_States [Online; accessed 08-May-2025].

[12] Charles Karney. 2025. Online Geoid Height Calculator. https://geographiclib.sourceforge.io/cgi-bin/GeoidEval [Online; accessed 08-May-2025].

[13] MathWorks, Inc. 2025. Location-Based Analysis of Visible GPS Satellites — Example Notebook. https://it.mathworks.com/help/satcom/ug/location-based-analysis-of-visible-gps-satellites.html [Online; accessed 08-May-2025].

[14] MathWorks, Inc. 2025. MATLAB. https://it.mathworks.com/products/matlab.html [Online; accessed 08-May-2025].

[15] NavSAS Research Group. 2025. NavSAS Research Group. https://navsas.polito.it/ [Online; accessed 08-May-2025].

[16] S. A. Ranamukhaarachchi, S. Lehnert, S. L. Ranamukhaarachchi, L. Sprenger, T. Schneider, I. Mansoor, K. Rai, U. O. Häfeli, and B. Stoeber. 2016. A micromechanical comparison of human and porcine skin before and after preservation by freezing for medical device development. *Scientific Reports* 6, 32074. https://doi.org/10.1038/srep32074 [Online; accessed 08-May-2025].

[17] StackExchange users. 2025. Google Earth Model Source. https://gis.stackexchange.com/questions/20259/what-datum-reference-ellipsoid-does-google-earth-use#:~:text=The%20heights%20on%20google%20earth,to%20the%20WGS%2084%20ellipsoid [Online; accessed 08-May-2025].

[18] Wikipedia contributors. 2025. Weighted Least Squares – Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/wiki/Weighted_least_squares [Online; accessed 08-May-2025].

A ADDITIONAL PLOTS

Figure 14: Place where all surveys were carried out.

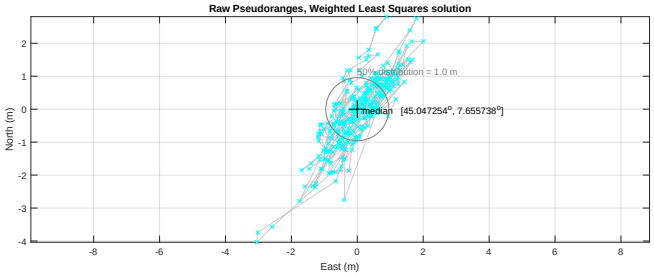

Figure 15: Positioning geoplot for logs collected in open-sky conditions (3)

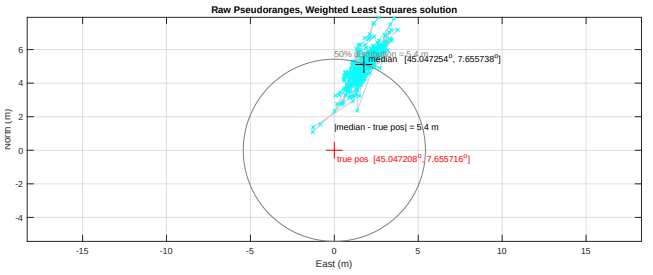

Figure 16: Positioning geoplot for logs collected in open-sky conditions, correlated with true position (3)

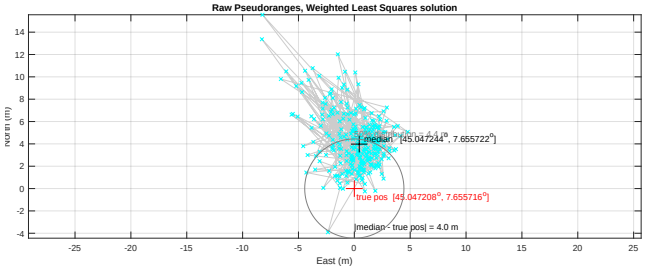

Figure 12: Positioning geoplot for logs collected in the interference scenario (3)

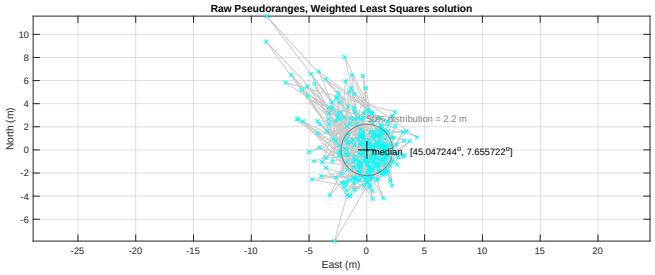

Figure 18: Positioning geoplot for logs collected in the interference scenario, correlated with true position (3)

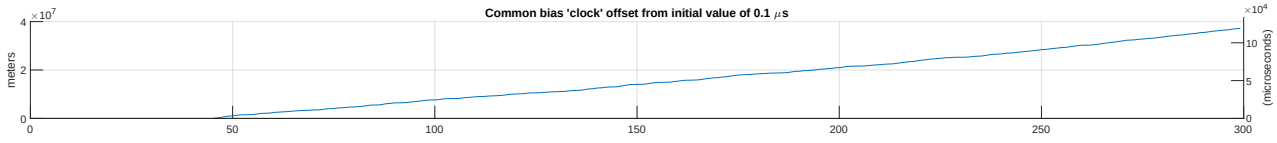

Figure 19: Common bias clock offset in HW clock discontinuity

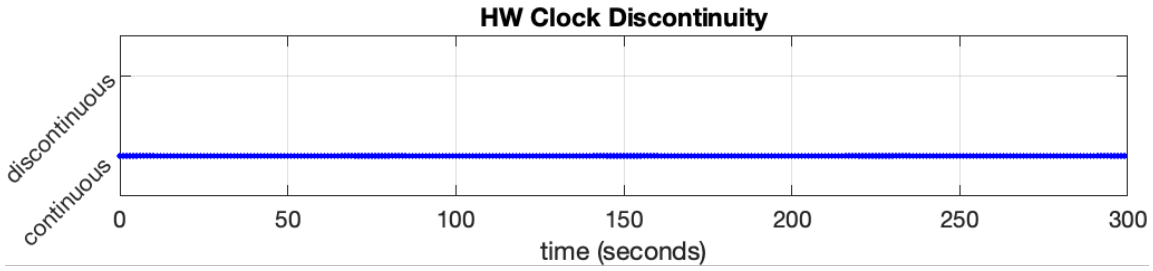

Figure 20: Status of the GNSS hardware clock over time with GNSSLogger and GPSTest in parallel

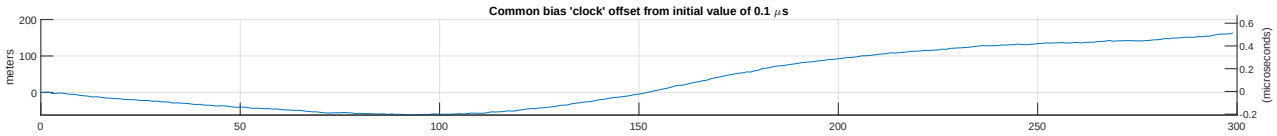

Figure 21: Common bias clock offset in open-sky and HW clock continuity conditions (3)

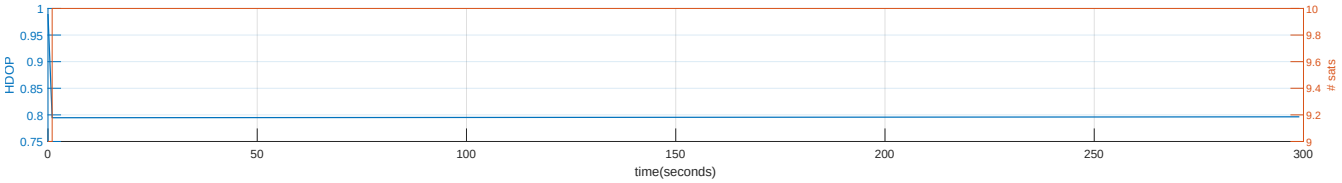

Figure 22: Horizontal diluition and number of satellites for logs collected in open-sky conditions (3)

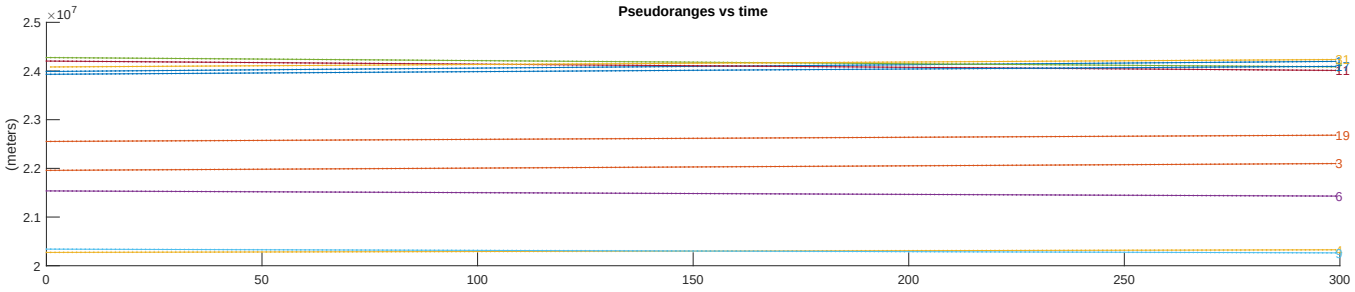

Figure 23: Pseudoranges vs. time for logs collected in open-sky conditions (3)

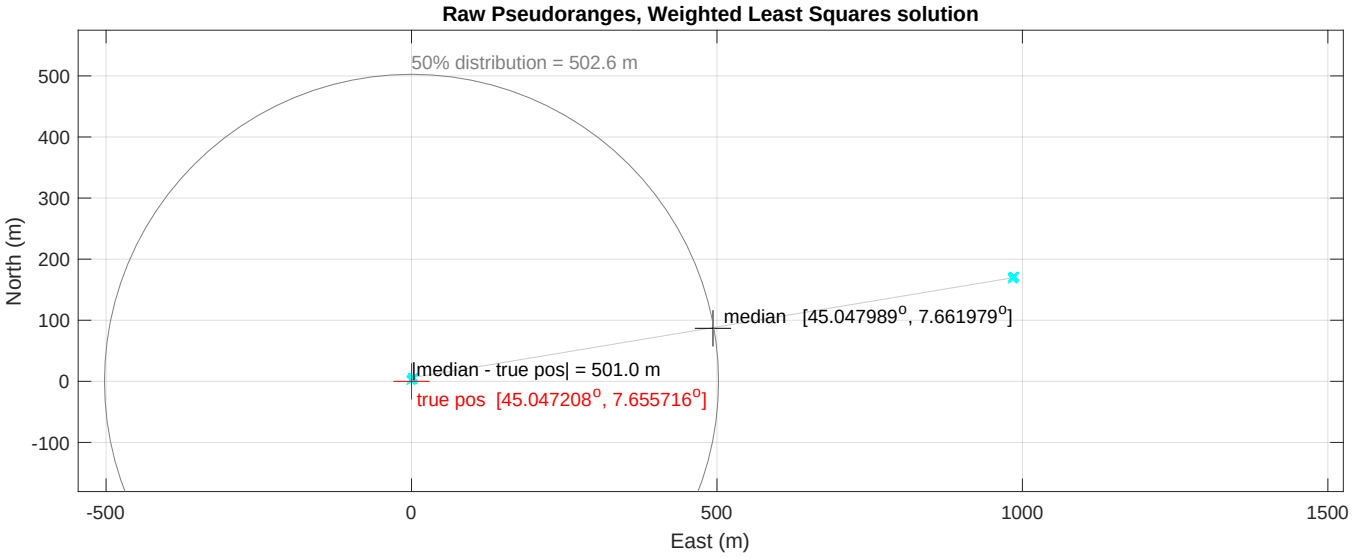

Figure 24: Positioning geoplot after applying spoofing to a location 1 Km from the real one, with no additional time delay used

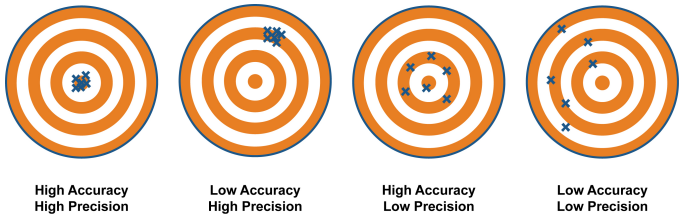

Figure 25: Accuracy vs. Precision