JOURNAL OF APPLIED ENGINEERING SCIENCES
ISSN: 2247-3769 / e-ISSN: 2284-7197

sciendo

VOL. 15(28), ISSUE 2/2025
ART.NO. 454 pp. 207-212

A PYTHON-BASED FRAMEWORK FOR THE TRANSFORMATION AND VISUALIZATION OF GNSS DATA IN ENGINEERING AND GEOSPATIAL APPLICATIONS

Alina Corina Bălă[a],*, Floarea-Maria Brebu [a]

[a] Politehnica University Timisoara, Faculty of Civil Engineering, Department of Overland Communication Ways, Foundations and Cadastral Survey, 2 Traian Lalescu Street, Romania, E-mails: * alina.bala@upt.ro, floarea.brebu@upt.ro

Received: 08.08.2025 / Accepted: 10.09.2025 / Revised: 31.10.2025 / Available online: 15.12.2025

DOI: 10.2478/jaes-2025-0024

KEY WORDS: GNSS data, RINEX 2.11, Python-based automation, Coordinate transformation, Geospatial visualization, Interactive mapping.

ABSTRACT:
Global Navigation Satellite System (GNSS) observables are standardized in RINEX files, but most existing tools for position extraction are fragmented and require manual intervention. This paper introduces a Python-based framework that automates the processing of RINEX 2.11 observation files by extracting approximate receiver positions, transforming them into the WGS84 system, filtering invalid data, and exporting results in structured formats. The workflow produces multi-format outputs (CSV, KML, interactive maps), enabling direct integration with engineering and geospatial applications. Implemented in Google Colab with open-source libraries, the framework was validated on real datasets, demonstrating reproducibility, modularity, and scalability. The proposed approach supports diverse applications, including surveying, cadastral mapping, geodetic monitoring, and spatial data science.

1. INTRODUCTION

Global Navigation Satellite Systems (GNSS) have become indispensable tools in geodesy, surveying, navigation, and environmental monitoring. The Receiver Independent Exchange Format (RINEX) provides a standardized exchange structure that ensures interoperability between heterogeneous GNSS receivers. Among its versions, RINEX 2.11 remains widely adopted in engineering and scientific applications due to its long-term dataset compatibility and integration with existing processing software ("EUREF Permanent GNSS Network," n.d.) (figure 1).

Example of a RINEX 2.11 Observation Block (simplified)

```
HEADER
MARKER NAME: STATION_A
APPROX POSITION XYZ: 4156202.7812 1614585.1439 4545577.0393
ANTENNA: DELTA H/E/N: 1.9109 0.0000 0.0000

OBSERVATION DATA
Epoch: 2023-02-12 10:15:00 UTC
PRN      C1 (m)         L1 (cycles)    P2 (m)         L2 (cycles)
G01   22762639.5    119618463.8    22439886.1    117922403.5
G11   20313569.2    106748517.2    24314519.9    121897123.2
G19   22313314.0    122246195.6    20555546.5    107512894.4
```

Figure 1. Example of a RINEX 2.11 observation block (simplified, based on ("EUREF Permanent GNSS Network," n.d.))

Despite its relevance, efficient extraction, processing, and visualization of GNSS observation data remain challenging. Traditional workflows rely on manual steps or commercial software, limiting reproducibility and scalability in scientific pipelines (Estey & Meertens, 1999) (figure 2).

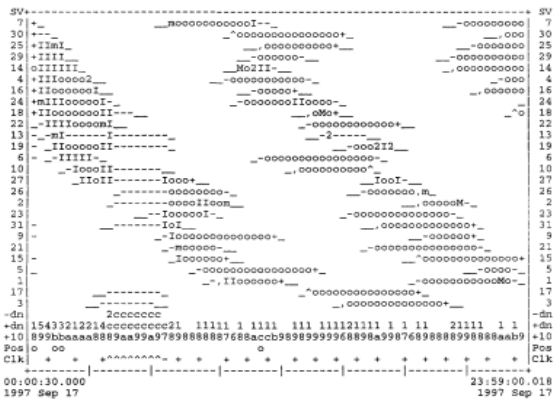

Figure 2. QC report is available in two portions—the short report segment (Estey and Meertens 1999)

Recent years have witnessed the emergence of open-source Python-based solutions addressing these limitations, providing lightweight alternatives for data preprocessing, quality control, and visualization. To better situate the present contribution, the following section reviews relevant advances in GNSS data formats, open-source processing frameworks, and methodological studies on positioning accuracy and reproducibility.

2. RELATED WORKS

The RINEX format has been the backbone of GNSS data interoperability for decades. The official specification ("EUREF Permanent GNSS Network," n.d.) defines the structure of observation and navigation files, ensuring receiver independence. While RINEX 2.11 remains widely used due to its backward compatibility, several studies have highlighted inherent challenges. For example, Wanninger (Wanninger 2018) identified biases introduced by mixed GPS L2P(Y)/L2C carrier-phase observations in RINEX-2, underlining the importance of format consistency and further emphasized metadata completeness for reliable data transfer in engineering contexts (figure 3).

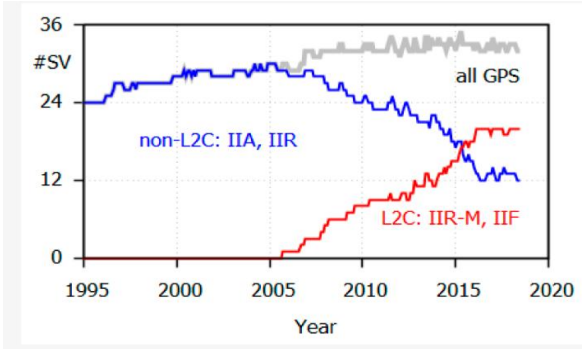

Figure 3. Development of the global positioning system (GPS) space segment with respect to L2C capable satellites (data source: IGS file of antenna corrections IGS14_2000.ATX) (Wanninger 2018)

Multiple software tools have emerged to facilitate GNSS data processing outside commercial environments. TEQC (Estey and Meertens 1999) represented one of the earliest comprehensive toolkits, still widely referenced for preprocessing. More recent efforts include PyRINEX (Han et al. 2024), which enables batch editing, quality control, and conversion of RINEX 2.x and 3.x files (figure 4); GNSSpy (Isik et al. 2021) designed for slicing, merging, and visualization; and gnss_lib_py (Knowles et al., 2024), which provides a modular analysis framework in Python.

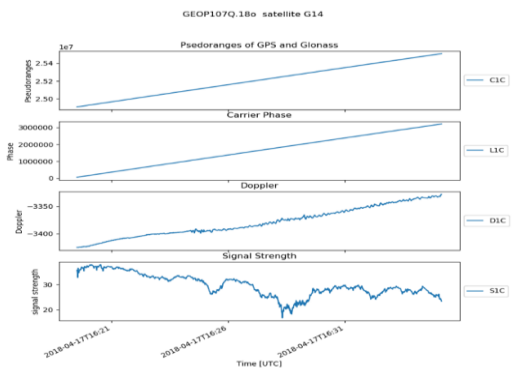

Figure 4. Example of PyRINEX interface enabling batch editing, quality control, and conversion of RINEX files (Han et al. 2024)

In addition, Kawamoto et al. (Kawamoto et al. 2023) developed RINGO (figure 5), a preprocessing tool with advanced cycle-slip and ionospheric correction capabilities, while pyrinex (Bruni [2018] 2025) offers lightweight parsing and NetCDF/HDF5 conversion.

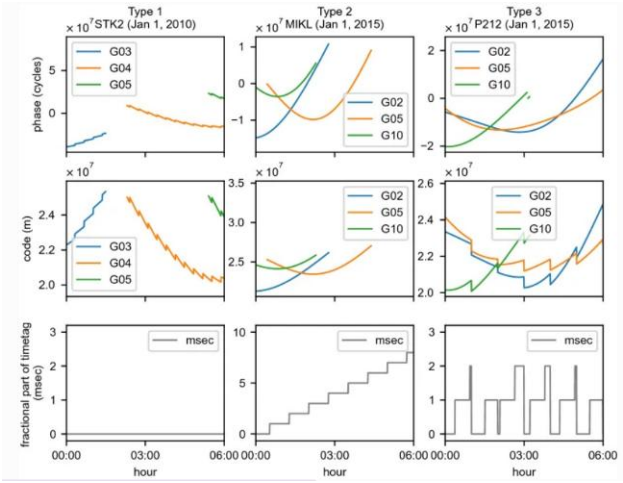

Figure 5. RINGO - Examples of observation data using different types of clock jumps: **a** Type 1, **b** Type 2, and **c** Type 3 clock jumps. The top, middle and bottom panels show phase and code pseudoranges, and millisecond part of time tag, respectively. The observation data of IGS sites STK2 (January 1, 2010) and MIKL (January 1, 2015), and GEONET site P212 (January 1, 2015) are shown (Kawamoto et al. 2023)

Several studies have investigated how preprocessing strategies and observation duration influence GNSS positioning accuracy. Langley et al. (2017) highlighted the sensitivity of accuracy to session length and data cleaning strategies. Lau (Lau and Tai 2023) proposed a systematic quality assessment approach for CORS stations, focusing on multipath and carrier-phase data integrity. Similarly, Afifi et al. (Afifi and El-Rabbany 2016) explored the benefits of multi-GNSS PPP for improving precision under various session lengths (figure 6).

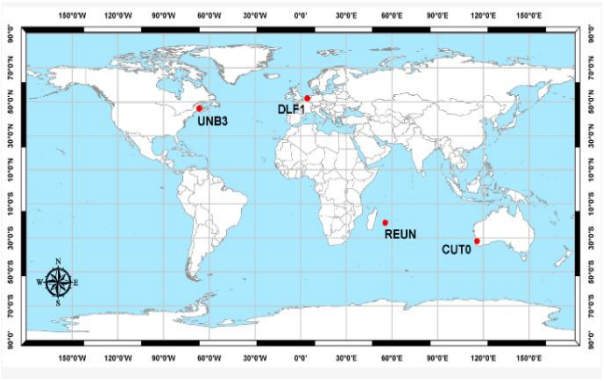

Figure 6. Analysis station (Afifi and El-Rabbany 2016)

Vierinen et al. (Vierinen et al. 2015) presented a statistical framework for estimating GNSS hardware biases, analyzed the impact of tropospheric modelling on vertical positioning

precision. Collectively, these works stress the necessity of standardized, automated pipelines for reliable GNSS analysis.

Effective visualization enhances the interpretability of GNSS datasets. Tools such as Matplotlib (Hunter 2007) and Folium ("Folium — Folium 0.20.0 Documentation," n.d.) provide widely used solutions for temporal and spatial representation of GNSS observations. At the same time, emerging applications demonstrate the extension of GNSS workflows beyond classical surveying. For example, Hernández Olcina et al. (Hernández Olcina et al. 2024) introduced a Python toolbox enabling Android raw measurements to be converted into RINEX format, while Zangenehnejad et al. (2023) reviewed the challenges of processing GNSS data from smartphones (figure 7).

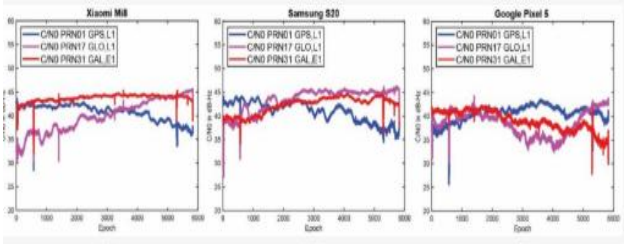

Figure 7. C/N0 measurements for selected PRNs and three smartphone (Xiaomi Mi8, Google Pixel 5 and Samsung S20) (Zangenehnejad et al. 2023)

The Springer Handbook of GNSS (Langley et al. 2017) remains a comprehensive reference covering theoretical models, error sources, and practical applications across disciplines.

3. METHODOLOGY

Building upon previous contributions, this study proposes an automated Python-based framework for processing GNSS session raw data stored in the RINEX 2.11 format. Although RINEX 2.11 remains widely used, the GNSS community increasingly adopts the RINEX 3.x standard, which supports multi-constellation observations and provides enhanced metadata structures. This distinction is important for understanding both the scope and the scalability of the proposed framework.

The framework integrates several essential tasks into a reproducible pipeline, including:

1. Extraction of **approximate ECEF positions** and **timestamps** from raw data RINEX 2.11 observation file headers.
2. Transformation of ECEF coordinates into **geodetic latitude, longitude, and ellipsoidal height** using the **WGS84/Pseudo Mercator reference system (EPSG:3857)**.
3. **Filtering and validation** of null or invalid values to ensure data quality.
4. Export of structured results into **CSV** and **KML** formats for subsequent analysis and GIS integration.
5. Visualization of spatial patterns through **interactive web maps** (Folium) and exploration of temporal trends using **time-series plots** (Matplotlib).

By combining these steps, the framework connects raw GNSS observations with engineering applications, supporting surveying, cadastral mapping, geodetic monitoring, and spatial data science.

The proposed methodology consists of five main stages, illustrated schematically in Figure 8:

1. **Data Input and Parsing**
   - The framework accepts **RINEX 2.11 observation files** (*. YYo) as input.
   - Metadata, including **APPROX POSITION XYZ** (approximate receiver positions in ECEF coordinates) and **TIME OF FIRST OBS** (observation timestamps), is extracted from file headers.
   - A filtering step discards files lacking positional metadata.
2. **Coordinate Transformation**
   - Extracted ECEF coordinates are converted to **geodetic latitude, longitude, and ellipsoidal height** using the **WGS84 /Pseudo Mercator reference system (EPSG:3857)**.
   - Invalid or null coordinates, such as values near zero, are automatically filtered out to ensure accuracy.
3. **Data Structuring and Export**
   - Validated observations are organized into a **tabular dataset** containing session ID, acquisition date, and geographic coordinates.
   - Results are exported to:
     - **CSV format** – for general-purpose data analysis.
     - **KML format** – for direct integration with **GIS software** and **Google Earth**.
4. **Visualization Tools**
   - **Interactive web maps** are generated using the **Folium library**, with session markers displayed in clusters to improve readability in dense areas.
   - **Time-series plots** created with **Matplotlib** provide insights into temporal variations, such as positional accuracy or session altitude trends.
   - These outputs support both **spatial** and **temporal interpretation** of GNSS session data.
5. **Implementation and Reproducibility**
   - The workflow was implemented entirely in **Python 3.10**, executed within **Google Colab**, ensuring cross-platform compatibility and easy reproducibility.
   - Core dependencies include:
     - pandas – data handling and structuring.
     - pyproj – coordinate transformations.
     - simplekml – KML file generation.
     - folium – web-based map rendering.
     - matplotlib – data visualization.

- The **modular architecture** enables straightforward extensions to support additional GNSS formats (e.g., **RINEX 3.x**) or real-time data streams.

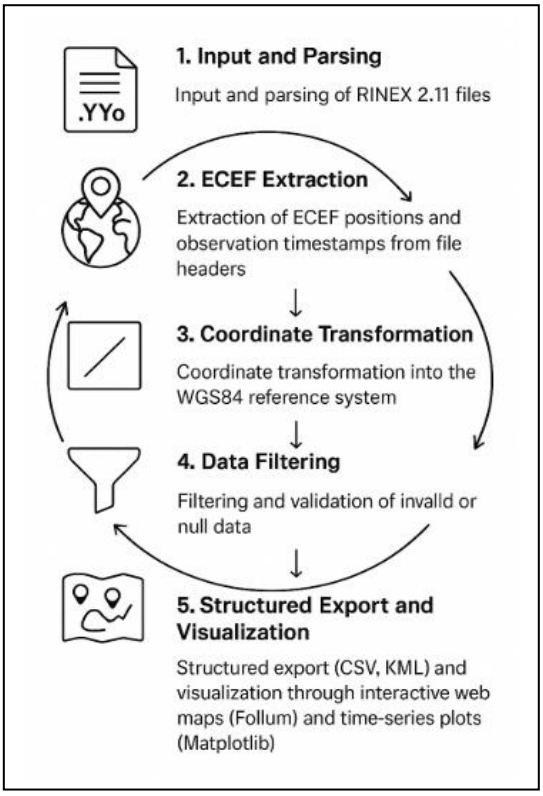

Figure 8. Workflow for GNSS raw data processing from RINEX 2.11 observation files

Figure 8 presents the proposed workflow in a clean, schematic format, highlighting the five core stages without including detailed sub-processes. This enhances readability and provides a concise overview of the framework.

4. RESULTS AND DISCUSSION

The pipeline was tested using multiple GNSS sessions, all located consistently within the same area. This consistency validates the reliability of the **APPROX POSITION XYZ** metadata extracted from RINEX headers. Table 9 provides an overview of the spatial distribution of the sessions derived from the raw data and validates the positional accuracy by transforming the coordinates from the ECEF (geocentric) WGS84/ITRF reference frame to the WGS84/Pseudo-Mercator projection (EPSG:3857).

Session GNSS Rinex files	date	latitude	longitude	altitude
52150431	2023-02-12	45.744566	21.229938	129.926008
52152980	2023-10-25	45.744554	21.229953	135.790736
57210801	2025-03-21	45.744550	21.229942	138.282000
57210890	2024-03-29	45.744546	21.229939	134.775392

Table 9. Spatial consistency of GNSS sessions processed with the proposed framework (EPSG:3857 WGS84/Pseudo Mercator)

To assess the internal consistency of the extracted coordinates, a statistical analysis was performed on the four GNSS sessions summarized in Table 9. The standard deviation of latitude and longitude was 8.6×10^{-6}° and 6.9×10^{-6}°, respectively, corresponding to approximately 0.96 m and 0.76 m in ground distance. The ellipsoidal height showed a standard deviation of 3.5 m, reflecting the expected sensitivity of the vertical component when relying solely on RINEX header metadata. The results confirm that the framework maintains sub-meter planimetric stability within sub-meter accuracy in planimetric coordinates, while the vertical discrepancies underline the limitations of approximate positions recorded in RINEX 2.11 files. Despite this, the obtained precision level is sufficient for visualization, preliminary spatial consistency checks, and rapid data quality assessment in engineering applications.

To ensure seamless execution, the following Python libraries must be installed:

- ✓ pip install pandas pyproj simplekml folium matplotlib numpy tqdm
- ✓ Example of library imports in Python:
- ✓ import pandas as pd # Data handling
- ✓ from pyproj import Transformer # Coordinate transformation
- ✓ import simplekml # KML file generation
- ✓ import folium # Web-based interactive maps
- ✓ import matplotlib.pyplot as plt # Visualization
- ✓ import numpy as np # Numerical operations (optional)
- ✓ from tqdm import tqdm # Progress tracking (optional)

Despite the overall positional stability in planimetric coordinates, the extracted ellipsoidal heights revealed variations between 129.9 m and 138.3 m. This difference of nearly 9 meters may be attributed to receiver-specific estimation methods for approximate positions, session-dependent environmental conditions, or rounding precision in header records. These results confirm earlier studies (Wang & Rothacher, 2016; Tudorache et al., 2022), which showed that vertical positioning is the most sensitive GNSS component. Visualization outputs provided complementary insights (figure 10).

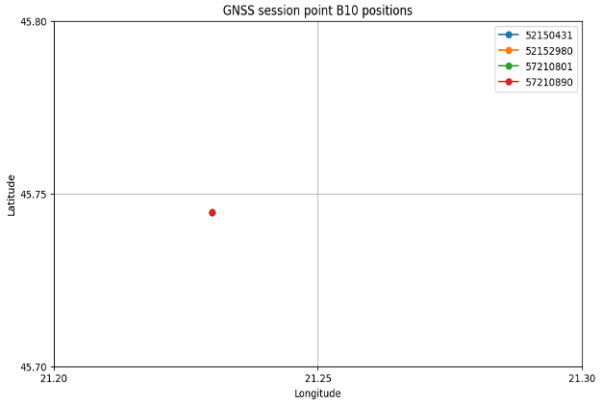

Figure 10. Visualization outputs of GNSS session points in WGS84 projection (EPSG:3857)

The CSV export ensured compatibility with statistical and engineering software, while the KML (figure 11) outputs allowed direct validation within GIS platforms and Google Earth. Folium-based interactive maps enabled clustering and exploration of session markers, reducing visual overlap in dense areas. Temporal analysis with Matplotlib further highlighted session-to-session altitude differences, demonstrating the utility of combining spatial and temporal perspectives.

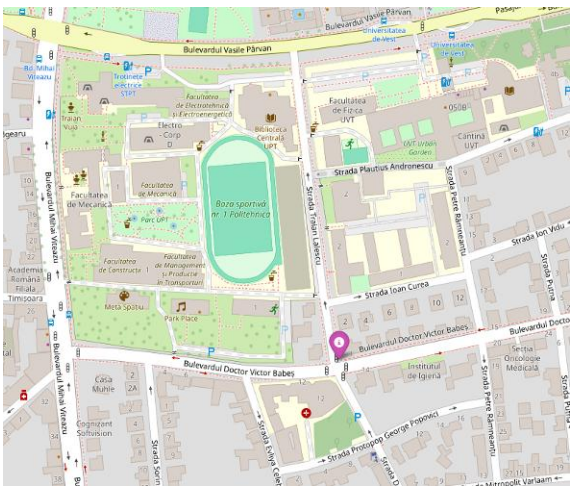

Figure 11. Interactive maps enabled clustering and exploration of session markers

These results align with recent research advocating for reproducible and interactive visualization methods in geospatial analysis (Lau and Tai 2023, Zangenehnejad et al. 2023). Nevertheless, several limitations must be emphasized. Since the framework relies solely on metadata fields rather than raw pseudorange or carrier-phase observations, the extracted positions cannot substitute precise GNSS processing. Furthermore, files lacking valid APPROX POSITION XYZ entries are discarded, which may limit applicability for incomplete datasets. Future extensions should therefore integrate observation-based position estimation and quality control modules (Wanninger 2018, Vierinen et al. 2015) to enhance robustness.

5. CONCLUSIONS

This study proposed and demonstrated an automated Python-based framework for extracting, transforming, and visualizing GNSS session data stored in RINEX 2.11 format. By integrating parsing of header metadata, ECEF–WGS84 coordinate transformation, structured data export, and multi-format visualization, the framework provides a reproducible workflow that reduces manual intervention and enhances data interpretability.

The experimental application on four RINEX files confirmed the framework's ability to consistently retrieve and transform approximate receiver positions. While planimetric stability across sessions was maintained, variations of up to 9 meters were observed in the vertical component, underlining the sensitivity of height estimation in GNSS workflows. These results demonstrate both the usefulness and the inherent limitations of relying solely on header metadata for engineering applications.

The multi-channel export strategy represents a key contribution:

- CSV outputs support statistical and engineering analyses,
- KML files allow seamless integration with GIS platforms and Google Earth,
- Folium maps provide interactive and web-accessible visualizations,
- Matplotlib plots enable temporal and comparative studies.
- Beyond data processing and visualization, the framework offers additional advantages:
- Reproducibility: The use of open-source Python libraries and implementation in a Google Colab environment ensure transparency and platform independence.
- Modularity: Each stage of the workflow is implemented as an independent module, allowing step-by-step verification, easy debugging, and future extension.
- Scalability: The framework is designed to be extended to multi-GNSS datasets or even real-time data streams, supporting larger and more complex applications.

The framework connects GNSS metadata with engineering applications. Its lightweight and adaptable design makes it suitable for surveying, cadastral mapping, and environmental monitoring.

Future work will pursue three main directions. First, the framework will be extended to ensure full compatibility with **RINEX 3.x** specifications and **multi-constellation GNSS datasets**, enabling the joint processing of GPS, GLONASS, Galileo, and BeiDou observations. Second, upcoming developments will incorporate **raw observation–based position estimation** using both pseudorange and carrier-phase measurements to achieve higher positional accuracy and to move beyond metadata-level analyses. Third, the framework will integrate **advanced quality-control modules**, including cycle-slip detection, multipath assessment, and statistical accuracy evaluation using reference station coordinates. Collectively, these enhancements will significantly improve the **robustness, precision, and**

applicability of the framework in both academic research and professional GNSS data processing environments.

Authors' Contributions

The authors made the following contribution to this paper: Conceptualization (ACB, FMB). Writing - review and editing (ACB, FMB). All authors read and approved the final manuscript.

Acknowledgements

This paper benefited from financial support through the Program on "Supporting the research activity by funding an internal grant competition ", Competition 2025, financing contract no. 13618/May 19, 2025.

Conflict of Interests

The authors declare that there are no conflicts of interest related to this article.

References

Afifi, Akram, and Ahmed El-Rabbany. 2016. "Precise Point Positioning Using Triple GNSS Constellations in Various Modes." Sensors 16 (6): 779. https://doi.org/10.3390/s16060779.

Bruni, Sergio. (2018) 2025. Serioca/Pyrinex. Python. July 5, released August 23. https://github.com/serioca/pyrinex.

Estey, Louis H., and Charles M. Meertens. 1999. "TEQC: The Multi-Purpose Toolkit for GPS/GLONASS Data." GPS Solutions 3 (1): 42–49. https://doi.org/10.1007/PL00012778.

"EUREF Permanent GNSS Network." n.d. Accessed September 18, 2025. https://www.epncb.oma.be/_documentation/formats/rinex.php.

"Folium — Folium 0.20.0 Documentation." n.d. Accessed September 18, 2025. https://python-visualization.github.io/folium/latest/.

Han, Jinzhen, Seung Jun Lee, Hong Sik Yun, Kwang Bae Kim, and Sang Won Bae. 2024. "PyRINEX: A New Multi-Purpose Python Package for GNSS RINEX Data." PeerJ Computer Science 10 (January): e1800. https://doi.org/10.7717/peerj-cs.1800.

Hernández Olcina, Jorge, Ana B. Anquela Julián, and Ángel E. Martín Furones. 2024. "Python Toolbox for Android GNSS Raw Data to RINEX Conversion." GPS Solutions 28 (2): 95. https://doi.org/10.1007/s10291-024-01631-9.

Hunter, John D. 2007. "Matplotlib: A 2D Graphics Environment." Computing in Science & Engineering 9 (3): 90–95. https://doi.org/10.1109/MCSE.2007.55.

Isik, Mustafa Serkan, Volkan Ozbey, Serdar Erol, and Ergin Tari. 2021. "GNSSpy: Python Toolkit for GNSS Data." 2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS, July 11, 8550–53. https://doi.org/10.1109/IGARSS47720.2021.9553880.

Kawamoto, Satoshi, Naofumi Takamatsu, and Satoshi Abe. 2023. "RINGO: A RINEX Pre-Processing Software for Multi-GNSS Data." Earth, Planets and Space 75 (1): 54. https://doi.org/10.1186/s40623-023-01811-w.

Langley, Richard B., Peter J.G. Teunissen, and Oliver Montenbruck. 2017. "Introduction to GNSS." In Springer Handbook of Global Navigation Satellite Systems, edited by Peter J.G. Teunissen and Oliver Montenbruck. Springer International Publishing. https://doi.org/10.1007/978-3-319-42928-1_1.

Lau, Lawrence, and Kai-Wing Tai. 2023. "A Data Quality Assessment Approach for High-Precision GNSS Continuously Operating Reference Stations (CORS) with Case Studies in Hong Kong and Canada/USA." Remote Sensing 15 (7): 1925. https://doi.org/10.3390/rs15071925.

Vierinen, Juha, Anthea J. Coster, William C. Rideout, Philip J. Erickson, and Johannes Norberg. 2015. "Statistical Framework for Estimating GNSS Bias." Preprint, arXiv. https://doi.org/10.48550/ARXIV.1508.02957.

Wanninger, Lambert. 2018. "Detection of RINEX-2 Files With Mixed GPS L2P(Y)/L2C Carrier Phase Observations." Sensors 18 (12): 4507. https://doi.org/10.3390/s18124507.

Zangenehnejad, Farzaneh, Yang Jiang, and Yang Gao. 2023. "GNSS Observation Generation from Smartphone Android Location API: Performance of Existing Apps, Issues and Improvement." Sensors 23 (2): 777. https://doi.org/10.3390/s23020777.