# G-NUT SOFTWARE LIBRARY – STATE OF DEVELOPMENT AND FIRST RESULTS

**Pavel VÁCLAVOVIC*, Jan DOUŠA and Gabriel GYÖRI**

*Research Institute of Geodesy, Topography and Cartography, Geodetic Observatory Pecný, 244,*
*251 65 Ondřejov*
**Corresponding author's e-mail:* pavel.vaclavovic@pecny.cz

(*Received* January 2013, *accepted* May 2013)

**ABSTRACT**

A Global Navigation Satellite System (GNSS) software library called G-Nut has been developed at the Geodetic Observatory Pecný (GOP) since 2011. Several applications built of the library will be provided as an open source in 2013 and consequently users are able to modify source code and use it for processing their own data free of charge. The main purpose of the project is to create a programming package suitable for implementing various end-user applications such as kinematic position estimation, long-term permanent station coordinates monitoring, zenith tropospheric delay estimation, satellite clock estimation and others. The library is written in C++ programming language following the object-oriented concept. Basic class structure implementing inputs/outputs and product/data containers support both real-time and post-processing modes. Integration of all available global navigation satellite systems (GPS, GLONASS, Galileo, BeiDou, QZSS) as well as new tracking signals is properly handled. The configuration is governed through the XML format. The estimation model currently supports the least square adjustment, the Kalman and square root covariance filtering methods based on processing undifferenced data and fixed precise orbit and clock products. The estimated state vector includes receiver coordinates and clocks, troposphere zenith path delays and initial carrier phase ambiguities. The first applications based on G-Nut library are shown with examples for off-line/online kinematic/static precise point positioning and ultra-fast troposphere estimation.

**KEYWORDS:** GNSS, Precise Point Positioning, coordinates, tropospheric path delays, G-Nut, C++ development library

## 1. INTRODUCTION

Many up-to-date Global Navigation Satellite System (GNSS) processing tools take advantage of the Precise Point Positioning technique, PPP (Zumberge et al., 1997). While supported by the precise GNSS products (Kouba and Héroux, 2001), the PPP became a powerful and widely used technique also reliable to substitute traditional relative positioning in many end-user applications. Most important advantage of the PPP is that it is an autonomous technique, using undifferenced data from a single receiver, and does not require raw data from a reference network. Obviously, the network observations are still essential for the precise product estimations, such as orbits and clocks. The number of reference stations required for the PPP is, however, significantly smaller than that for the relative positioning, when properly accounting for the size of area. On the other side, while differencing observations eliminates or significantly reduces many systematic errors, undifferenced PPP-like processing needs to model all effects precisely.

In 2011, a project aimed to develop a generic library for implementing various end-user GNSS data processing tools, initially based on the PPP method, started at the Geodetic Observatory Pecný (GOP). The core of the software package is called *G-Nut library* and represents a collection of various subroutines and tools for developing end-user applications which could serve either GNSS experts or other users of GNSS in their own domain of interest. The first G-Nut applications described in this paper together with relevant parts of the G-Nut library are expected to be provided as an open source. Currently, the first versions are being consolidated and optimized for making them available on GOP's website (http://www.pecny.cz) during the first half of 2013. The following principal reasons could be identified behind initiating the G-Nut developments at GOP:

- stimulate exploitation of highly efficient and autonomous PPP technique in various domains,
- stimulate up-to-date and high-level expertise and development in GOP towards a multi-GNSS constellation and real-time analyses,
- update GOP analysis centre products contributing to various services (EUREF Permanent Network, International GNSS Service, the EUMETNET GNSS Water Vapour Programme and others) with new or enhanced solutions,

- support an easy development of end-user applications targeted for geodesy, seismology, meteorology, climatology and other scientific areas either for GNSS experts or for other users,
- contribute to a robust production of precise products for supporting PPP (in near future).

The main purpose of this paper is to present and describe features and functionalities of the software package, demonstrate currently achieved results and outline developments anticipated for future. First, the G-Nut core library is described in the next section. Second, the end-user applications currently derived from the library, and mainly based on the PPP technique, are introduced in the third section. Initial results demonstrating the current capabilities are provided within the fourth section. Finally, conclusions and an outlook are given in the last section.

## 2. G-NUT LIBRARY DEVELOPMENT

The G-Nut software library has been written in C++ taking advantage of the object-oriented concept: 1) object/classes are designed for coupling relevant data and methods, 2) hiding/protecting selected components from external entities, 3) inheriting classes for extending or overriding existing object functionalities, 4) enabling interfaces for deferred implementation of object methods and 5) polymorphism supporting the ability to replace an object with its derived sub-object in run time. All these features are important for developing a flexible, sustainable and expandable core library and its extensive functionality. The object-oriented design has been used, for example, in developing a unique implementation of a multi-GNSS and multi-signal processing capability, support of various input/output operations, real-time or post-processing analysis, multi-threaded operations and many others.

The G-Nut is designed primarily for Linux operating systems and providing command-line applications since it is usually well suitable for scientific purposes. Some other platforms including Windows will be supported in near future. The library implementation is thus independent of any large software framework needed for the developing graphical user interface. However, the G-Nut initially exploits the Boost library (Kempf, 2002) for a platform independent multi-thread implementation, but even this dependence is likely to be revised in near future. For matrix representation and factorization methods, the G-Nut utilizes the New-Mat 09 matrix library developed by Davies (Davies, 1997) and, for parsing XML configurations, the Pugixml light-weight C++ library (Kapoulkine, 2012) is used. The XML mark-up language was selected for the library classes as well as for end-user application settings due to its high flexibility, ease extension and support by various other editing and viewing user-tools.

### 2.1. *INPUTS AND OUTPUTS*

Figure 1 (left side) shows the strategy to implement access to input data and products and output results. While post-processing solutions are based on the file inputs and outputs, in case of the real-time processing continuous data streams need to be established. While in the first approach the files could be read at once before the processing start up, the latter requires buffered input/output (IO) operations maintained during the whole run of the processing due to possible disruptions of data streams. Any operation on continuous streams could be handled using either with blocking or non-blocking access. The first one is easier to implement, but requires a multi-threaded application where each continuous stream is handled by an individual thread. The second one can be used also in a single thread application. By designing relevant classes as thread-safe objects within the G-Nut, both approaches are generally supported.

File and stream IO operations are implemented using a virtual *gio* class which can be replaced in run time by its appropriate inherited class. Such implementation supports both real-time and post-processing approaches in a unique way for all end-user applications. The internal buffer of *gio* class supports the handling of text as well as binary data representation. If input data decoding (or output data encoding) is required, the *gio* object (or correctly its inherited object) is linked with any object derived from the generic *gcoder* object class in order to provide specific decoder (or encoder). All these usually supports both stream and file IO operations. The main difference is typically in containing header and data parts in files, while only data in streams. Header and data are thus handled separately in decoder/encoder classes and the header operations are active only in the *gio* file implementation (*gfile*).

### 2.2. *DATA AND PRODUCT REPRESENTATIONS*

Figure 1 (right side) displays the representation of individual data and products and its relation to the IO operations. While any single IO stream/file is represented by an individual *gio* thread (or via a sequence of calls in the main thread), each *gio* instance is associated with various data structures represented by a common virtual *gall* class. Inherited objects are implemented using containers of the Standard Template Library (Musser, 2001) and each object represents a collection of data or product elements designed by a virtual parent *gdata* class. Finally, Figure 1 shows various types of specific classes inherited from *gall* and *gdata* for representing specific data or product types. The element classes are designed in a self-contained form so they are usable and removable independently from the others. The examples are: single-station, single-epoch, single-satellite GNSS observations; single-epoch, single-satellite GNSS navigation message; single-epoch, single-satellite orbit and clock corrections etc.

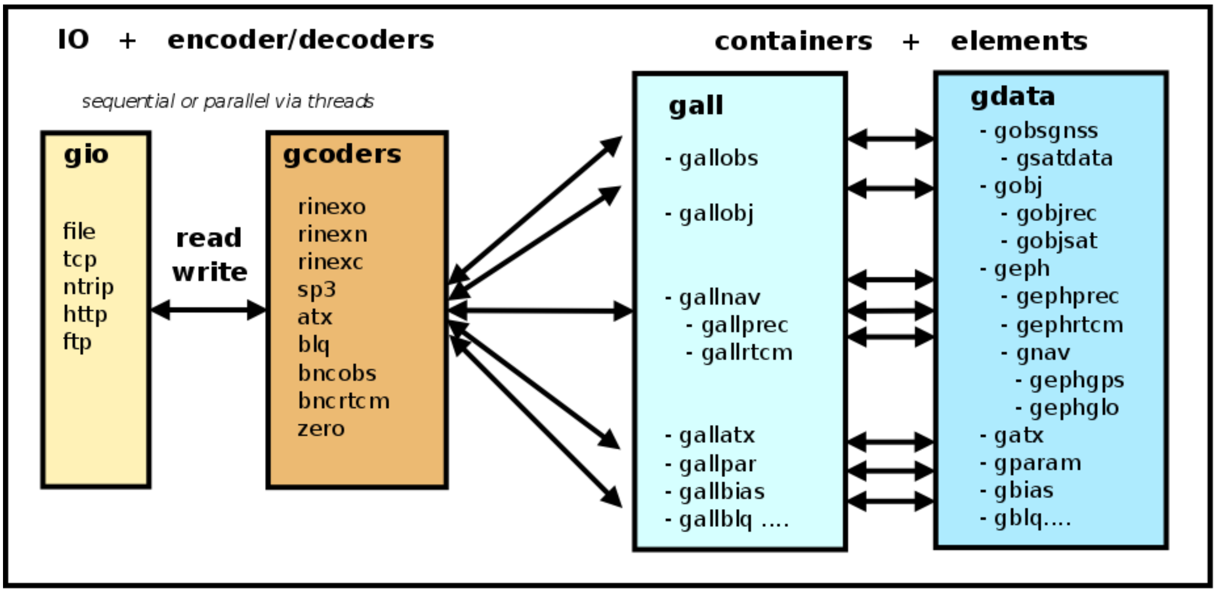

**Fig. 1** G-Nut library input/output implementation, data and product representations and its linkage.

The collection classes (*gall*) represent containers usable in both real-time and post-processing solutions via a common interface for getting required data or products. The *gall* classes are designed as thread-safe in order to support multi-thread applications with concurrent accesses to any common collection class. Finally, the collection classes support usually two cleaning operations – the first by setting a maximum number of elements stored in the collection (backward into the past) and secondly by the cleaning all data out of the specified interval. These are sufficiently flexible either for a continuous operation in real-time or for the post-processing in a filter or batch mode.

### *2.3. ADJUSTMENT METHODS AND PROCESSING MODELS*

The Kalman filter (Kalman, 1960) was used as a basic procedure for the parameter estimation. This recurrent algorithm, which is based on a sequential adjustment, is very efficient for dynamical processes. It consists of two parts: prediction and update. While the state vector is predicted to the new epoch according to its expected behaviour, in the update step new observations are added into the processing system and the least squares adjustment is applied for a new statistically optimal estimate. The Kalman filter is thus very effective because only actual observations with the updated state and covariance matrix from the last epoch are processed. However, it was shown that the Kalman filter suffers from the numerical instability due to the rounding errors (Bierman, 1977). To avoid this problem, the square root form has been introduced and it is up to the user to configure whether the Kalman or the square root filter is used for the parameter estimation. The precision of filtered parameters is growing with the time (and new observations) so they are less accurate at the beginning of the interval than at the end. It could be convenient to acquire reasonably homogeneous accuracy for all involved parameters over the entire interval so the technique for backward smoothing has been developed too. It profits from the converged parameters estimated during the initial run and from a further observations filtering in the opposite direction. It should be noted that the backward filtering cannot be provided in real-time.

Classical sequential least squares adjustment as an alternative estimation technique to the Kalman filter is under development. Main disadvantage of this approach is the amount of parameters to be adjusted which requires the inversion of large information matrices. One way to decrease the matrix dimension is the pre-elimination of some parameters However, it is computationally still very inefficient to process a long period data span in a single step. The solution can be divided into more short processing intervals, forming individual normal equations and stacking them into a unique system. In this way we got the identical system for the solution like we would have processed all the data at once (Mervart and Lukeš, 2007).

Station coordinates, receiver clock corrections, zenith troposphere delays and initial phase ambiguities belong to the parameters estimated in the PPP process. Station coordinates and receiver clock corrections were designed as stochastic processes driven by white noise. According to the quantity of the coordinate noise it is decided whether the receiver position is considered as static or kinematic. Apart from the stochastic estimation, coordinates could be fixed to a priori values. A fundamental property of the white noise is a zero mean, thus white noise is not suitable for modelling the zenith troposphere delays (ZTDs) and the stochastic process with random walk is applied instead. The behaviour of both processes could be set in the user XML configuration file. Ambiguity parameters are modelled as constants according to the definition, but estimated as real values only until now.

## 3. END-USER APPLICATIONS

First applications developed using the G-Nut core library are called Anubis, Geb and Tefnut (mythology names). The Anubis is a tool aimed for checking, editing, converting and concatenating GNSS observation and navigation files in the Receiver Independent Exchange Format - RINEX (Gurtner and Estey, 2009). The implementation of multi-GNSS observations defined in new RINEX V3.01 format was tested using the data set collected within the IGS multi-signals tracking campaign - MGEX (Weber, 2012). Although some capabilities are still under development, the tool is already used for the RINEX editing and content monitoring at the GOP operational, data and analysis centres. The detailed description is, however, out of the scope of this paper.

Geb and Tefnut are first tools for the PPP-based processing of undifferenced phase and code GNSS observations derived from the G-Nut. While the Geb was designed for the precise receiver position monitoring and navigation tasks, the Tefnut is aimed for monitoring of the troposphere, in particular to support numerical weather forecasting, now-casting or climatology. Both applications support post-processing and real-time modes. The accuracy of the estimated parameters depends mainly on the quality of precise satellite orbit and clock products available either from files or continuous real-time data streams.

### *3.1. PROCESSING STRATEGY*

The section 2.1 describes decoders/encoders which are responsible for transferring data from files or streams into data structures and vice versa. The RINEX format up to the version 3.01 for GNSS observations and broadcast ephemerides, the CLK-RINEX format for precise clock corrections and the SP3 format for precise orbit products are supported as input for the post-processing. For real-time, data are collected in a format defined by the Radio Technical Commission for Maritime Services (RTCM). The BKG Ntrip Client (Weber et al., 2012) users are able to process off-line data as well as products from real-time data stored in files (internal text format). To achieve the highest accuracy, antenna phase offsets and variations in the ANTEX format are applied too. In case input orbit and clock products refer to satellite centres of masses, the receiver corrections are applied only.

As mentioned in section 2, all structures for GNSS data were developed to support multi-GNSS (if necessary using class inheritance). G-Nut implementation is ready to read and store GPS, GLONASS, Galileo, BeiDou data and other from satellite-based augmentation systems, but only GPS data were tested in the processing until now. The others are expected to be included in 2013. The most important part of the GNSS software is the pre-processing which is responsible for checking an occurrence of cycle slips in precise carrier phase measurements and an incoherency between code and phase observations caused by the receiver clock jumps. Cycle slips are checked by the simultaneous monitoring of changes in the geometry-free and in the Melbourne-Wübbena linear combinations. If the changes exceed a defined threshold, the cycle-slip is detected and a new initial ambiguity is set up within the estimated parameter vector.

Finally, the PPP solution is based on undifferenced GNSS phase and code observations applying the ionosphere-free linear combination to eliminate the first order ionospheric refraction effect. The tropospheric delay cannot be eliminated in this way and therefore the zenith total delay is estimated in the parameter vector. The Global Mapping Function, GMF (Boehm et al., 2006) is used to convert the estimated zenith delays to any receiver-satellite elevation. Troposphere parameters are a priori modelled for the hydrostatic effect using the Global Pressure and Temperature model, GPT (Boehm et al., 2007), while the estimated part is considered to be due to wet gases. If precise a priori ZTDs are available, they can be introduced as known for the shortening of the initial solution convergence.

## 4. RESULTS

This chapter demonstrates first results achieved for selected station coordinates and ZTD monitoring, based on both static and kinematic modes and using different precise products. Figures 2 and 3 show estimated parameter time-series for WTZR station during April 19, 2012 in 30 seconds sampling rate. Station coordinates are displayed in the topocentric North, East and Up system while reduced to the mean values. No coordinate transformation was done, therefore, demonstrated coordinate results are in the processed epoch and in the same reference frame as satellites (IGS08). Additionally, ZTDs are compared with results of the EUREF repro1 campaign. Figure 2 gives more accurate results because providing the static solution while Figure 3 provides the kinematic solution. ZTD and height are strongly correlated and it is shown in Figures 2 and 3 that the correlation is more significant in kinematic mode. Repeatabilities of kinematic coordinates are about 10 cm so in this range Geb is able to be used for deformation detection. Considering ZTD, there are periods of better and worse consistency with respect to the EUREF-repro1 products, which is caused by imperfect modelling of some systematic effects. The initial convergence period for reaching 20 cm accuracy is a few tens of minutes and is clearly visible in both plots. This effect is removed in the post-processing solution by the backward smoothing technique (not shown).

In order to show the Geb's capability to track the large earthquake, such as Tohoku-Oki in Japan on 11th March 2010, data from permanent GNSS stations

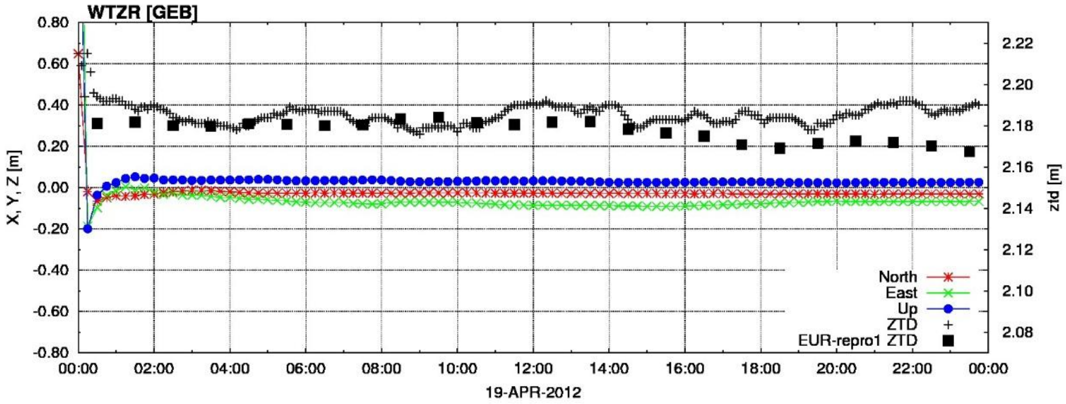

**Fig. 2** Static coordinate and ZTD solution for WTZR in simulated real-time.

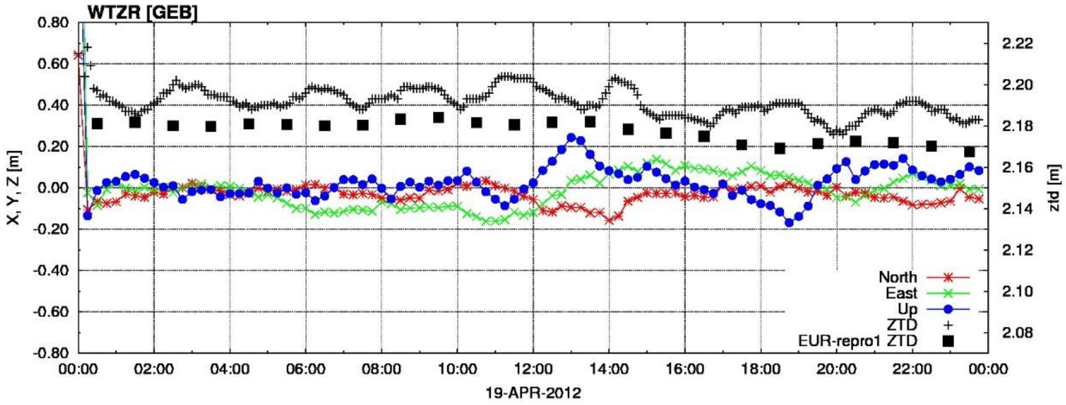

**Fig. 3** Kinematic coordinates and ZTD solution for WTZR in simulated real-time.

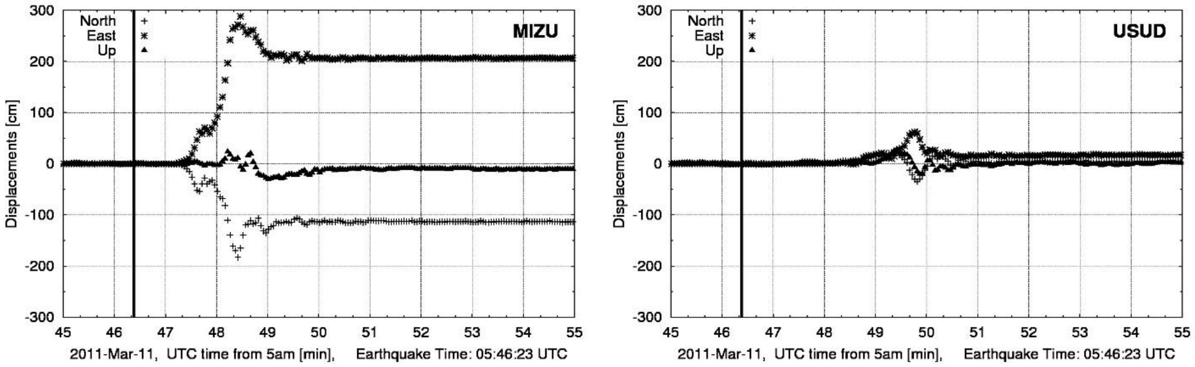

**Fig. 4** Displacements due to Tohoku-Oki earthquake at MIZU and USUD GNSS stations.

MIZU and USUD were processed. The earthquake happened at 5:46:23 UTC and reached the magnitude 9.0. Figure 4 shows estimated displacements of the two aforementioned stations. Station MIZU (left) was situated about 140 km from the epicentre and its position was shifted by a meter due to the strong initial earthquake shock waves. The second station USUD (right) was located more than 430 km far from the earthquake epicentre and only a smaller displacement after earthquake waves is observed in this case. The data were processed in a high-rate kinematic mode with fixed precise IGS orbits and clocks.

In order to assess the capability of ZTD monitoring, we prepared the benchmark campaign of 11 GNSS sites: BOGO, BOR1, BRST, GOPE, MAS1, NICO, ONSA, PDEL, POTS, WTZR and YEBE. The data were analysed during 44 days (April 2 – May 14, 2012) in post-processing and simulated real-time mode (a real-time processing in an offline mode, but utilizing archived real-time precise product). Figure 5 shows averaged ZTD biases and standard deviations

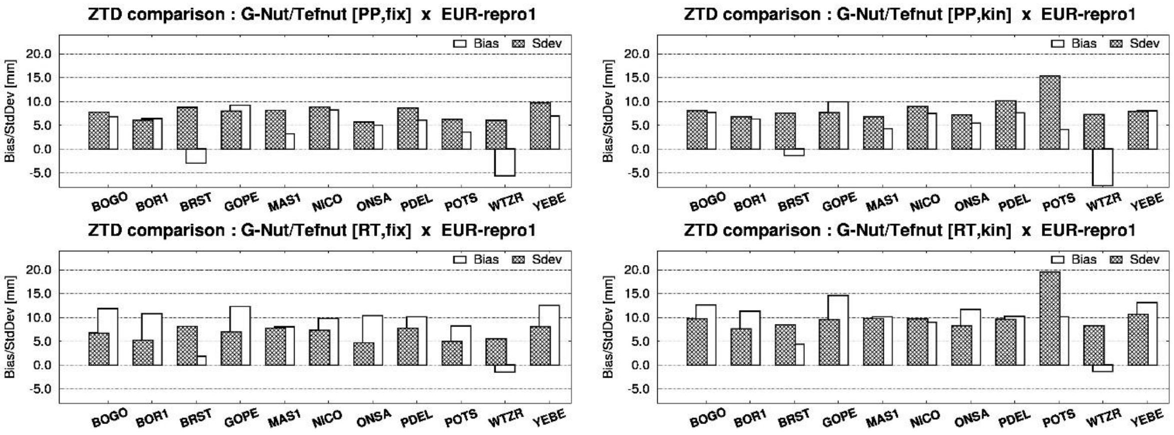

**Fig. 5** Statistics of ZTD over 44 days in post-processing (top) and simulated real-time (bottom) analysis and static (left) and kinematic (right) modes.

for all stations with respect to the EUREF-repro1 tropospheric product. All plots demonstrate that the standard deviation of ZTD is below 10 mm with a single case exception. Systematic errors are still significant which can be attributed to some incomplete models being under development, such as antenna phase variations, ocean loading and tide displacements. However, the GNSS biases are usually modelled in the NWP assimilation by estimating the station by station monthly mean values (Bennitt and Jupp, 2012).

## 5. CONCLUSION AND OUTLOOK

Objective of this paper was to present G-Nut project aiming at development of a GNSS software library. In the first part, fundamental properties and functionalities of the G-Nut core library and first derived applications were presented. Flexible input and output operations, data and products representation and adjustment methods were described including specific implementation aspects. Two end-user applications for coordinates and troposphere monitoring were introduced. First results were demonstrated by processing a benchmark campaign in post-processing and simulated real-time. The first release of the applications is expected before mid of 2013.

## ACKNOWLEDGMENT

This work was supported by the Czech Science Foundation (project no P209/12/2207).

## REFERENCES

Bennitt, G. and Jupp, A.: 2012, Operational assimilation of GPS Zenith total delay observations into the UK Met Office numerical weather prediction models. Mon. Weather Rev., 140(8), 2706–2719. doi: 10.1175/MWR-D-11-00156.1.

Biermann, G.J.: 1977, Factorization method for discrete sequential estimation. Academia Press, Inc., New York.

Boehm, J. and Niell, A.E., Tregoning, P. and Schuh, H.: 2006, Global Mapping Functions (GMF): A new empirical mapping function based on numerical weather model data, Geoph. Res. Letters, 33, L07304, doi: 10.1029/2005GL025545.

Boehm, J., Heinkelman, R. and Schuh, H.: 2007, Short note: a global model of pressure and temperature for geodetic applications, J. Geod. 81(10):679-683, doi: 10.1007/s00190-007-0135-3.

Davies, R.B.: 1997, Documentation for New-Mat 09, a matrix library in C++. Statistics Research Associates Limited, Wellington, New Zealand.

Gurtner, W. and Estey, L.: 2009, http://igscb.jpl.nasa.gov/igscb/data/format/rinex301.pdf.

Kalman, R.E.: 1960, A new approach to linear filtering and prediction problems. Journal of Basic Engineering, 82 (1), 35–45.

Kapoulkine, A.: 2012, http://www.pugixml.org.

Kempf, B.: 2002, The Boost. Threads Library. C/C++ Users Journal.

Kouba, J. and Héroux, P.: 2001, Precise point positioning using IGS orbit and clock products. GPS solution, 5 (2),12/28.

Mervart, L. and Lukeš, Z.: 2007, Adjustment Calculus. Nakladatelství ČVUT.

Musser, D.: 2001, STL tutorial and reference guide: C++ programming with the standard template library. Addison Wesley.

Weber, R.: 2012, The IGS Multi-Signals Tracking Campaign MGEX – Planning, Status and Perspectives, Presented at the IGS Workshop 2012, July 23-27, Olsztyn, Poland.

Weber, G. and Mervart, L: 2012, BKG Ntrip Client (BNC) v. 2.6 [software]. http://igs.bkg.bund.de

Zumberge, J. F., Heflin, M.B. Jefferson, D.C. Watkins, M.M. and Webb, F.H.: 1997, Precise point positioning for the efficient and robust analysis of GPS data from large networks, J. Geophys. Res., 102(B3), 5005–5017, doi: 10.1029/96JB03860.