An Open Source Toolkit for GPS Processing, Total Electron Content Effects, Measurements and Modeling

T. Gaussiran
gauss@arlut.utexas.edu

D. Munton, B. Harris, B. Tolman

Applied Research Laboratories
The University of Texas at Austin
International Beacon Satellite Symposium 2004
Trieste, Italy
October, 2004

Abstract

Applied Research Laboratories, The University of Texas at Austin (ARL:UT) has established a cross platform, open source software project called the GPS Toolkit, or GPSTk. The goal of the GPSTk is to provide a best-of-class software suite that supports GPS research, analysis, and development. The code is released under the terms of the Lesser GNU Public License.

The GPSTk software suite consists of a library and a set of applications that build on the library. The library provides base functionality common to most GPS processing, including algorithms and standards commonly defined in GPS textbooks. Library capabilities include reading and writing observations in standard formats such as RINEX; ephemeris evaluation; position determination using receiver autonomous integrity monitoring (RAIM); atmospheric delay modeling; cycle slip detection and correction; and P-code generation. The applications extend the capabilities of the library to support research and advanced applications of GPS.

The GPSTk provides the core set of functionality that is used for GPS research and development at ARL:UT. ARL:UT has been involved with satellite navigation since Transit in the 1960's and is currently conducting research in a wide variety of GPS related fields, including precise survey, monitor station networks, and ionospheric studies. ARL:UT continually improves the library and is committed to its development and maintenance. ARL:UT's goal is to make the GPSTk a community-wide resource for all users of GPS and GNSS technology. Participation is welcomed at all areas including: new algorithms, suggestions for improvement, bug reports and contributions of additional functionality or applications.

Introduction

Applied Research Laboratories, The University of Texas at Austin (ARL:UT) has established an open source software project called the GPS Toolkit, or GPSTk. The GPSTk software suite consists of a library and collection of applications that support GPS research, analysis, and development. The software is distributed as source code, released under the terms of the Lesser GNU Public License [1]. The LGPL grants the user a number of rights, notably the ability to choose whether to modify and redistribute the source with any application whether commercial or open source. The design

of GPSTk is object oriented and the code is written in platform independent ANSI C++. It has been installed and tested successfully under Windows, Linux, Solaris, and AIX.

The GPSTk distribution is available for download at the GPSTk website, http://www.gpstk.org. [2] The website is hosted by SourceForge [3], which provides web services to this and thousands of other open source projects. In addition, the GPSTk website provides an API reference for the latest stable release. The application programmer interface (API) provided by the GPSTk library can be generated by the user directly from the distributed source code using the doxygen package [4].

This article presents the GPSTk starting with the most general information first. The capabilities of the library are summarized by topic. Then, a simple example of using the library is presented and explained. An in-depth application of the GPSTk is also described, in which GPSTk applications are used in a process to model the total electron content of the ionosphere. In conclusion, the future direction of the GPSTk project is discussed.

Project Details

Background

ARL:UT has been active in satellite navigation research and application development since the time of the Transit constellation. ARL:UT has since diversified its work to include, among other areas, precise surveying, radio wave propagation, global tracking networks, receiver design, ionosphere modeling, and deployable sensors. All of these projects have included a substantial software development effort. Over the years, the work of all the GPS-related projects within ARL:UT have contributed software, experience and testing to the creation of a unified common code base for basic GPS software processing. The resulting ARL:UT software suite has now been released as the open source GPS Toolkit.

License

The source code for the GPSTk is licensed under the Lesser GNU Public License, or LGPL, which grants any user a number of rights and responsibilities. Under terms of this license, users are allowed to redistribute the source code and to modify it. However, unlike the related GNU Public License, or GPL, users may choose not to redistribute code. Notably, commercial applications may use the library and the resulting products may be sold for profit. The details of the LGPL are included in the GPSTk distribution.

Documentation

The GPSTk website is the best place to begin reading about the GPSTk. The website provides guidance on how to obtain, build and install the software suite. The website also describes example code found in the code distribution. Publications describing the GPSTk, similar to this one, are available for download. Furthermore, the entire library API for the latest stable release is available at the website. The library API can also be generated directly from the source code using the doxygen utility [3].

The website is hosted by SourceForge [3], which hosts a vast number of other open source projects. The SourceForge project supplies a number of online support services vital to the GPSTk project, including: file downloads, bug tracking, mail lists, task lists, and requested features lists.

The design goals of the GPSTk library are portability, modularity, understandability, extensibility, and maintainability. These goals allow the GPSTk to maximize the audience and lifetime of the library while decreasing the costs associated with long-term maintenance. The GPSTk meets these goals by strict adherence to the ANSI standard C++ and Object Oriented (OO) analysis and design techniques.

OOA/D contrasts with procedural design supported by the C, FORTRAN, and MATLAB languages. In procedural programming, a *function library* is provided to the user. In OO programming, a *class library* is provided. Each class is an independent module that can be invoked by the user as an object or extended by the user in the form of a new class. Classes can build upon each other through a number of object-oriented principles, such as inheritance or encapsulation. The GPSTk library relies heavily on the Standard Template Library, or STL [5], which is part of the ANSI C++ standard library. The STL provides uniform OO data structures known as 'containers'. These include linked lists, maps and vectors; the STL also includes standard operations such as the quicksort algorithm that act on containers.

All of the GPSTk code includes documentation designed for extraction by doxygen, a freely available package that generates HTML-based API documentation [4]. Like the GPSTk, doxygen is platform-independent. The doxygen documents are available on the GPSTk web site or easily generated from the code in the distribution.

The GPSTk library supports a wide range of functions generally necessary for processing GPS observations. Such functions are commonly described in GPS textbooks or in standards definitions. The library provides file input and output for standard file types such as RINEX and SP-3. The library provides time transformations necessary to reconcile GPS time with other calendars such as the Julian system. Fundamental mathematics such as matrix manipulation and bivariate statistics are also provided. Atmospheric delay models, such as from the troposphere, are available in the GPSTk library. Each of these topics is discussed in depth in the following paragraphs.

RINEX Input and Output

The library provides for the reading and writing of data files in the RINEX format that is common to many projects. The RINEX specification version 2.1 is completely supported. The library also includes the capability to define custom 'extended RINEX' observation types that are useful for storing results and intermediate quantities in RINEX-like files.

Ephemeris Handling

The library provides complete satellite ephemeris storage and evaluation capabilities with input from both broadcast ephemeris (RINEX navigation files) [6] and SP-3 format files [7]. The ephemeris storage classes encapsulate all the details of ephemeris handling. They require only input file names to load the ephemerides and implement the standard computation algorithms such as the satellite position computation algorithm defined in ICD-GPS-200 [8]. The ephemeris store objects are designed using object-oriented methods (inheritance and polymorphism) so that user can trivially switch usage between broadcast and precise ephemeris.

Time Handling

Dates and times are fundamental to GPS and to the GPSTk. The '`DayTime`' class implements many useful date and time formats into a single very powerful object. These formats include Modified Julian Date, GPS time, and calendar dates.

Operators are provided to enable the user to treat a `DayTime` object like any other fundamental C type like `int` or `double`. There are also text string-based routines that allow custom formatting of time tags, as well as text string interpretation.

Vector and Matrix Routines

The GPSTk library includes an extensive matrix and vector package built entirely using templates. The package includes the usual arithmetic operators like addition, subtraction, multiplication, and division. It also includes LU decomposition, singular value decomposition, and Cholesky decomposition. There are also several mathematical algorithms in the library that include bivariate statistics, polynomial fitting, Lagrange interpolation, and Runge-Kutta interpolation.

Troposphere Delay Modeling

Several troposphere models are provided in the library. Their design uses inheritance and polymorphism so that the user may trivially switch models, or create a new one. There are currently five models implemented that include Hopfield models [9] and the New Brunswick UNB3 model [10].

Positioning

Positioning and navigation is an area of the library that is expected to grow substantially in the near future. Currently, functionality is limited to autonomous pseudorange solutions implemented using either linearized least squares or the algebraic solution [11]. A receiver autonomous integrity monitoring (RAIM) algorithm is also included [12].

Additional Functionality

There is much more functionality in the GPSTk library than can be detailed here. There is a P-code generator that produces the bit sequence for P-code as defined by the ICD-GPS-200 [8]. An application framework is defined that allows programmers to easily define and create console applications. There are a wide variety of low-level support functionalities, such as exception handling, string utilities, formatted file handling, and command line argument processing that ease the development of console based programs.

Overview of GPSTk Applications

The GPSTk software suite includes several complete applications. These programs utilize the library to read, process, write, and display GPS data. These programs serve both as useful utilities and as detailed examples of programming using the GPSTk library. GPSTk includes the following types of applications: RINEX utilities, RINEX plotting, cycle slip detection and estimation, residual analysis, date and time conversions, ionospheric modeling, and Octave bindings. Each of these is described in more detail in the following paragraphs.

RINEX Utilities

The RINEX utilities provide a set of applications that can be used to examine and manipulate RINEX observation files. They include capabilities to summarize a file, extract selected data into a simple columnar format, edit a file, and generate data residuals or corrections like ionospheric total electron content, elevation and azimuth, geometry-free phase, and wide-lane bias.

RINEX Plotting

The RINEX plotting application is a menu-driven graphical interface to the RINEX utilities. This application takes the outputs of these utilities and displays them in a 2D scatter plot. This is the only code in the GPSTk distribution that is not C++; it is a Perl script that uses the Perl::Tk graphics module. Consequently, for this application, Perl [13] and Tk [14] are required in addition to the base GPSTk requirements.

Cycle Slip Correction

The cycle slip application implements an algorithm based on the work of Blewitt (1990) [15]. This application detects and estimates the size of the cycle slip using the geometry-free and wide lane linear combinations of raw dual frequency pseudorange and carrier phase data. The results can be utilized by the RINEX utilities to produce a cycle slip corrected RINEX observation file.

Residual Analysis

The residual analysis application computes two types of measurement residuals using a single receiver or two receivers in a zero baseline configuration. For a single receiver, observed range deviations (observed minus expected pseudoranges) are computed using a broadcast or precise ephemeris. For the zero baseline, double differences across receivers and satellites are computed which are a function of only receiver noise and hardware biases. The outputs are suitable for plotting in gnuplot [16] or other plotting programs.

Time Conversions

The time conversion utility is a simple application that converts one time format into a variety of other formats. This is handy when looking at two different file formats that contain different time representations like GPS week and seconds of week or year, month, day, hour, minute, and second.

Ionospheric Modeling

The ionospheric modeling applications utilize the two frequency TEC estimate from the RINEX utilities and computes a model of the ionosphere. This process includes estimating the biases in the TEC observation due to hardware biases in the GPS receivers and satellites. The results are suitable for plotting in gnuplot or other plotting program. These applications are discussed in detail, with examples, in the following section.

Octave Bindings

The Octave bindings are not an end application but an example of how specific GPSTk functions can be called from other languages or products. Octave [17] is an open source computer algebra system that is quite similar to MATLAB. This concept could be extended to other products or languages.

Using the GPSTk to Generate TEC and TEC Maps

Background

The GPSTk includes an application that involves processing GPS data for remote sensing of the ionosphere. It takes data from a large set of receivers that are distributed within a compact geographical region and computes a contour map of the ionospheric vertical TEC.

This analysis adopts the common model assumption, that the ionosphere consists of a thin shell at a fixed height, usually 350 or 400 kilometers, surrounding Earth. The two-dimensional density of free electrons in this shell is called the total electron content or TEC of the ionosphere. The GPS satellites broadcast a ranging signal on dual frequency bands from an altitude substantially above that of the ionosphere. Thus the GPS signals pass through the ionosphere, in general at an oblique angle, and the ionospheric TEC imposes a dispersive delay on the signals. This delay, and the TEC along the line of sight, or 'slant TEC', that produces it, can easily be computed from the ranging signals as follows.

$$TEC_{slant} = \frac{(R_1 - R_2)}{\alpha} \bullet TECUperM$$

$$\alpha = \frac{f_1}{f_2} - 1$$

$$TECUperM = \frac{f_1^2}{20.48} \bullet 1.0e-16$$

where pseudorange observations are denoted with symbol R (subscripts refer to the GPS frequencies), TECUperM is a constant that converts meters to TEC units (TECU), and α is a constant dependent only on the two GPS broadcast frequencies f_1 and f_2.

The slant TEC is related to the vertical TEC by an obliquity factor, easily computed from simple geometric considerations.

$$VTEC = TEC_{slant} \bullet obliquity$$

with

$$obliquity = \sqrt{1 - a^2}$$

$$a = \frac{R\cos(e)}{R + h},$$

where e is the elevation of the satellite as seen at the receiver (assumed on Earth's surface), R is Earth's radius and h is the height of the ionosphere.

The point at which the signal crosses the model ionospheric shell is called the ionospheric pierce point or IPP. The IPP for a given satellite and receiver can be computed from the receiver position and the observed elevation and azimuth angles of the satellite as seen at the receiver, along with the height of the ionosphere. If the angle subtended at Earth's center by the receiver and the IPP is

$$p = \frac{\pi}{2} - e - \sin^{-1}\left(\frac{R\cos(e)}{R + h}\right),$$

then the latitude ϕ and longitude λ of the IPP are

$$\phi_{IPP} = \sin^{-1}\left(\sin(\phi)\cos(p) + \cos(\phi)\sin(p)\cos(a)\right)$$

$$\lambda_{IPP} = \lambda + \sin^{-1}\left(\sin(p)\sin(a)/\cos(\varphi)\right),$$

where the receiver position is given by latitude and longitude (ϕ, λ) and (e,a) are the observed elevation and azimuth of the satellite; of course the radius of the IPP is the same as that of the ionosphere, R+h.

The GPS measurement of ionospheric TEC is complicated by the fact that hardware delays exist in both the satellite and the GPS receiver. These delays arise from differing path lengths for the two GPS signal frequencies; they are essentially constant over short periods (days), but vary over longer timescales. The sum of the two biases, satellite+receiver, may be estimated using GPS data, provided the ionospheric TEC is known or modeled.

Procedure

The approach is first to estimate the 'satellite plus receiver' biases from the nighttime portion of the data, when the ionospheric TEC is negligibly small. Then the estimated biases are applied to the slant TEC measurements, from which the vertical TEC is computed, and this data, along with the estimated biases, are then used to fit a simple model ionosphere at the points of a 2-d grid at the ionospheric height using a weighted least squares estimator.

The first step is preprocessing of the raw data. The GPSTk uses RINEX format for data files, including an extension of the standard RINEX specification that allows intermediate results, as well as the raw data, to be stored within the RINEX format. The GPSTk application `ResCor` takes dual frequency range data in the input RINEX file (obtained, for example, from IGS sites on the internet) and computes the slant ionospheric delay, the latitude and longitude of the 'pierce point', and the elevation and azimuth of the satellite, storing the results in an 'extended' RINEX format file. The output file includes, in the file header, a unique label for the receiver as well as the precise receiver position.

Bias Determination

Program `IonoBias` simultaneously reads all the preprocessed files, making use of a solar ephemeris to limit consideration to data that was collected during the local nighttime hours. `IonoBias` applies all the TEC data to a least squares fit of both a simple model of the (nighttime) ionospheric TEC and all the biases:

$$VTEC = bias + obliquity \bullet F(\phi, \lambda),$$

where the function F represents the model of the ionosphere, and the bias term represents, as per user selection, either the 'satellite+receiver' bias or a single receiver bias (effectively averaging over satellites). The user is also allowed to select either a linear, quadratic or cubic model for the fit. The cubic model is

$$\begin{aligned} F(\phi, \lambda) = {} & A_0 + A_1\phi + A_2\lambda \\ & + B_0\phi^2 + B_1\phi\lambda + B_2\lambda^2 \\ & + C_0\phi^3 + C_1\phi^2\lambda + C_2\phi\lambda^2 + C_3\lambda^3 \end{aligned}$$

The quadratic and linear models are the same without the higher order terms, specifically the linear model has only the A terms, while the quadratic has only A and B terms. The number of unknowns is the number of biases plus the number of model coefficients, which is 3 (linear), 6 (quadratic) or 10 (cubic).

The data is applied to an unweighted sequential least squares estimator. In this algorithm, the data equation is

$$\vec{d} = P \bullet \vec{X} \ ,$$

where $\vec{d}$ is the data (VTEC), $\vec{X}$ is the solution vector consisting of the unknowns, and P is the partials matrix which relates the two. For each data point, the partials matrix is computed from the information found in the preprocessed RINEX file, and the following running sums are updated:

$$I\vec{d} = \sum_{data} d \bullet \vec{P}$$
$$IC = \sum_{data} P^T \bullet P$$

After all the data is processed, the matrix IC is inverted to give the covariance of the solution $\vec{X}$, as

$$Cov = inverse\left(IC\right)$$
$$\vec{X} = Cov * I\vec{d}.$$

The results, including the estimated biases, with an uncertainty equal to the square root of the corresponding diagonal element of the covariance matrix, are output to a file.

Mapping

Program `TECMaps` reads the same preprocessed RINEX files that `IonoBias` does, as well as the estimated 'satellite + receiver' biases that were output by `IonoBias`, and at each epoch uses the delay data to estimate the ionospheric TEC on each point of a horizontal grid. The user inputs the boundaries of a region in latitude and longitude over which the map is defined, as well as the grid resolution and type. Two types of grids are supported, one uniform in latitude and longitude and the other uniform in spatial separation of the grid points.

`TECMaps` reads all the preprocessed RINEX data files in parallel, and creates a map at each data epoch in turn. First the RINEX headers are all read and a list of all receiver labels and their positions is made. The mapping algorithm is as follows.

The first step at each time point is to compute the average VTEC value; the final computed map will have this as its average value by definition. The program then executes a double loop, over all grid points and all data values, computing each grid value using all the data. The ionospheric pierce point is computed, along with the obliquity and the vertical TEC (VTEC). An uncertainty in the VTEC value is computed as

$$VTECerr = \sqrt{d^2 + r^2}$$

where d is the uncertainty in the VTEC measurement, and r is the decorrelation error, equal to the range times a decorrelation rate which has been input by the user.

The bearing and range of the separation between the current grid point and the ionospheric pierce point of the current data is then computed. This separation vector is broken into its 2-d Cartesian components (X and Y) and is weighted using a decorrelation rate (as input from the user). Also, a minimum separation limit is enforced (also user input).

At the completion of the loop over data, the final grid value at the current grid point is computed via a least squares fit of the data to a model (f) of the surface, weighted by the uncertainty VTECerr; that is, by minimizing the function

$$\sum_{i=data} \left((d_i - f_i) / \sigma_i \right)^2 .$$

The user is given a choice of models, either constant $\left(f_i = a = const \right)$ or linear in separation $\left(f_i = a + bX + cY \right)$.

Results

In this section we present various results of the GPSTk bias determination and TEC mapping applications. Figure 1 shows receiver locations and the chosen grid boundaries for a typical run of `IonoBias` and `TECMaps`. The receiver locations represent IGS stations, the data from which is easily available on the Internet.

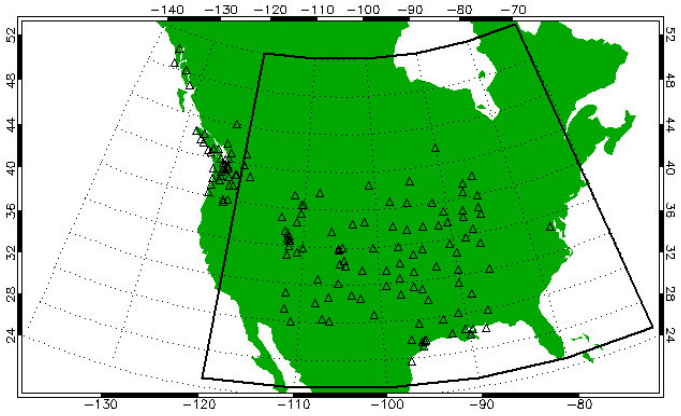

Figure 1 - Receiver positions and grid boundary.

Figure 2 presents the combined 'satellite+receiver' biases that were computed by `IonoBias` for one IGS site on days 271 through 276 of 2004. These results show the general consistency of the biases over the six day period, although there is variation with satellite, as well as some change with time.

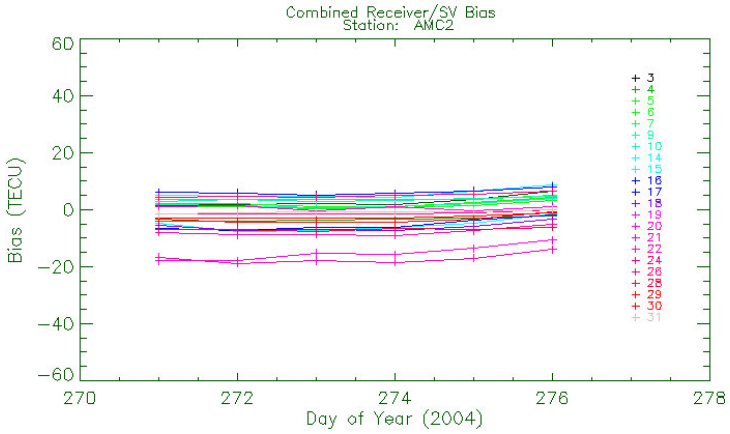

Figure 2. Satellite + receiver biases for site 'AMC2' as estimated by GPSTk application `IonoBias`.

Turning to another example, TEC maps were generated by the GPSTk applications for day 210 of 2004 using data from 366 IGS sites in a region centered on eastern North America, every 30 seconds for 24 hours. The following figures give some representative TEC map results at different times of day. Figure 3 is the result at local sunset; note the strong decrease in TEC in the east, as sunset approaches. Figure 4 is at local sunrise, and shows the increase of TEC in the east with sunrise. Finally, Figure 5 is near mid-day, when TEC is large throughout the region, and spatial variations are evident.

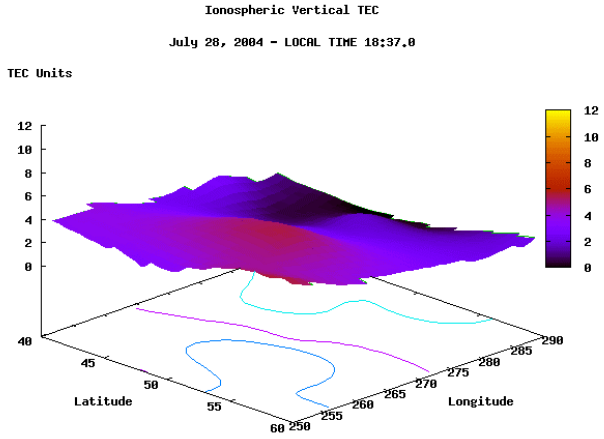

Figure 3. Vertical TEC map produced by GPSTk application `TECMaps` near local sunset.

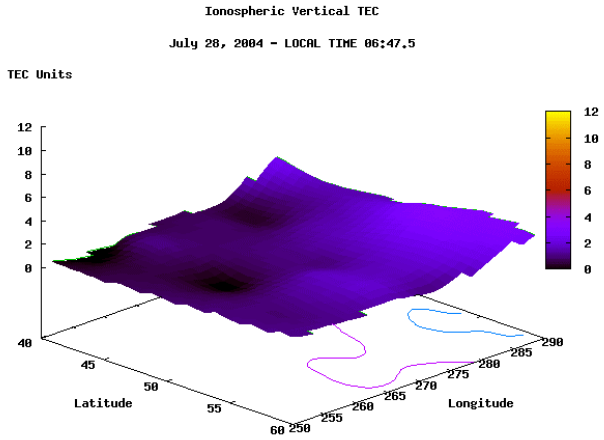

Figure 4. Vertical TEC map produced by GPSTk application `TECMaps` near local sunrise.

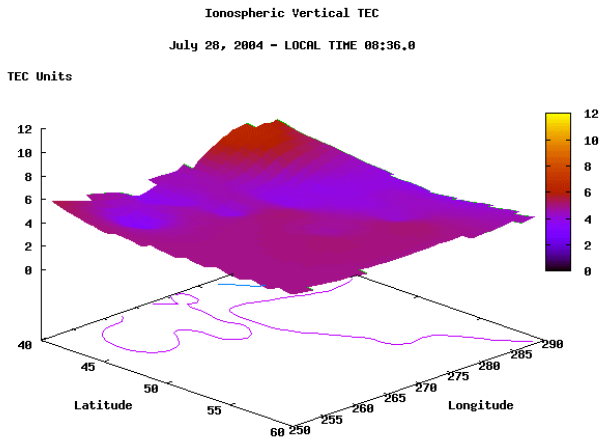

Figure 5. Vertical TEC map produced by GPSTk application `TECMaps` during daytime.

Using GPSTk for TEC analysis

ARL:UT routinely runs, each day, a process which performs data editing and computes slant TEC for RINEX format data files from a large number of sites, using applications that are available as part of the GPSTk distribution. The process runs automatically and requires human intervention only when uniquely anomalous data is encountered. The process is illustrated in Figure 6. It includes preprocessing for cycle slip detection and removal `using DiscFix`, the GPSTk discontinuity corrector. `DiscFix` also performs a phase-averaging process on the pseudoranges. The GPSTk applications `ResCor` generates slant TEC and removes satellite biases. At the end of the process the corrected TEC values are then available for analysis in 'extended' RINEX format files.

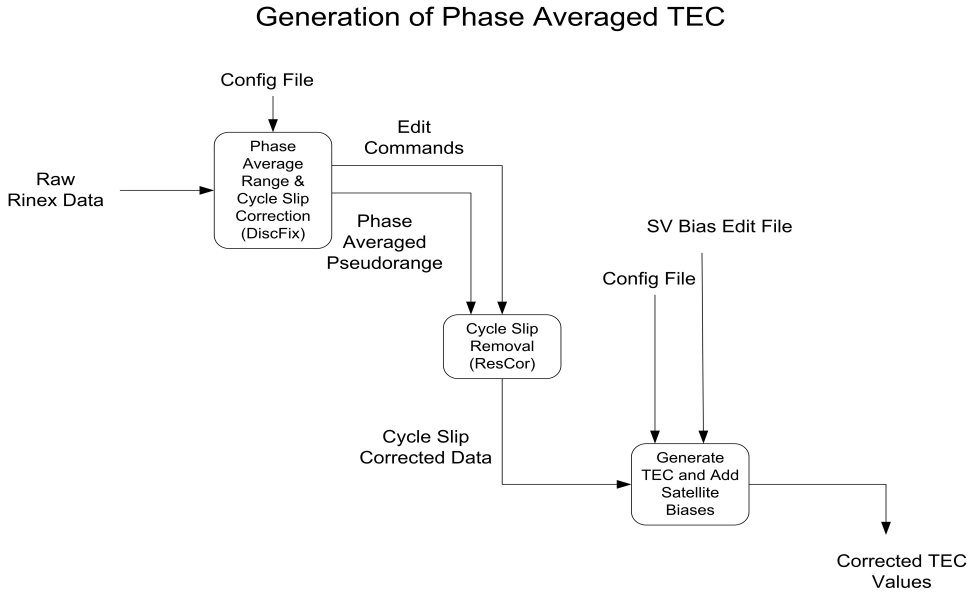

Figure 6. Automated ARL:UT TEC Data Processing

Future Directions

The growth of the GPSTk will depend strongly on user contributions. The contributions can take a variety of forms, from bug reports to new and modified source code. Users within ARL:UT have already identified a number of near term goals. These include platform support for other development environments such as Mac OS X. Other near term goals include the addition of differential positioning GPS with carrier phase ambiguity resolution, and software receiver capabilities.

Growth of the GPSTk will also be driven by shifts in the GNSS community. We anticipate supporting RINEX version 2.2 soon, and any further developments of the RINEX standard. In the near term, the first satellite to provide dual frequency pseudoranges to civilians is scheduled for launch in 2005. Furthermore, the European community is creating Galileo, which will provide a public, regulated service that is compatible with GPS, essentially augmenting the current constellation with a new one. In the long term GPS will have new signals in the L5 and M code. The GPSTk, with its emphasis on fundamental observations, can provide the basis to explore and exploit these changes.

It is our hope that university students, laboratory researchers, system engineers and software developers will contribute to, as well as benefit from, the GPS Toolkit. We have already seen many benefits to using this code within ARL:UT, and believe that the GPS community as a whole will see a similar benefit.

References

[1] The Lesser GNU Public License (LGPL). http://www.gnu.org/licenses/lgpl.html

[2] The GPSTk home page, hosted by SourceForge.net.. http://www.gpstk.org/

[3] The Sourceforge project web site. http://www.sourceforge.net/

[4] The doxygen utility. http://www.doxygen.org/

[5] The Standard Template Library Programmer's Guide. http://www.sgi.com/tech/stl/

[6] RINEX Standards. http://www.aiub.unibe.ch/donwload/rinex

[7] The SP-3 format for precise ephemeredes. http://gibs.leipzig.ifag.de/cgi-bin/sp3_doc.cgi?en

[8] ICD-GPS-200. http://www.navcen.uscg.gov/pubs/gps/icd200/

[9] Hopfield, H. S. "Tropospheric Effect on Electromagnetically Measured Range: Prediction from Surface Weather Data." Applied Physics Laboratory, Johns Hopkins University. Baltimore, MD, July 1970.

[10] Collins, J. P. and Langley, R. B. "A Tropospheric Delay Model for the User of the Wide Area Augmentation System," Geodesy and Geomatics Engineering, University of New Brunswick, Technical Report No. 187, September 1997.

[11] Bancroft, S. "An Algebraic Solution of the GPS Equations." IEEE Transactions on Aerospace and Electronic Systems. 21.7, (1985).

[12] R. Grover Brown, “Chapter 5: Receiver Autonomous Integrity Monitoring”, Global Positioning System: Theory and Application. AIAA. Washington, DC. 1996.

[13] The Perl language site. http://www.perl.org

[14] Tk web site. http://td.sourceforge.net

[15] Blewitt, G., "An Automatic Editing Algorithm for GPS Data." Geophysical Research Letters 17. 3 (1990):199-202.

[16] The Gnuplot utility website. http://www.gnuplot.info/

[17] The octave utility website. http://www.octave.org