HELIOSAT - An orbit determination software with applications to deep space missions

Øystein Olsen

June 2007

Series of dissertations submitted to the
Faculty of Mathematics and Natural Sciences, University of Oslo.
No. 649

ISSN 1501-7710

Cover: Inger Sandved Anfinsen.
Printed in Norway: AiT e-dit AS, Oslo, 2007.

Produced in co-operation with Unipub AS.
The thesis is produced by Unipub AS merely in connection with the
thesis defence. Kindly direct all inquiries regarding the thesis to the copyright
holder or the unit which grants the doctorate.

Unipub AS is owned by
The University Foundation for Student Life (SiO)

Acknowledgements.

I would like to express my gratitude to my advisor Dr. Kaare Aksnes for his support, the chance to earn my doctorate at the University of Oslo and for introducing me to the international radio science community.

I want to thank to Dr. John D. Anderson for the opportunity to come to Jet Propulsion Laboratory as a research assistant and for being my advisor during my year at JPL. It was his suggestion that prompted me to start developing orbit determination software. Without him, it would never have happened.

Thank to Dr. Per-Helge Andersen for long discussions on estimation theories and media corrections. HELIOSAT would not have evolved beyond run-of-the-mill software without his guidance, help and support.

Contents

CHAPTER I

Introduction

"God save thee, ancyent Marinere!
"From the fiends that plague thee thus–
"Why look'st thou so?"–with my cross bow
I shot the Albatross.

The author of this thesis has created a set of software libraries to facilitate analysis of problems in orbital dynamics. Possible tasks range from simple simulations to estimation of spacecraft orbits using tracking data. A simple simulation will perhaps not involve much more than an Ordinary Differential Equation (ODE) solver and a few basic routines to convert between different orbital elements. Tracking a spacecraft is, however, a much more complex task. It requires transformations between body-fixed and space-fixed frames of reference, different space fixed frames of reference and different time scales. Also required are solutions to the light travel paths between the tracking stations and the spacecraft, media corrections, interpolation of planetary and spacecraft ephemerides, estimation algorithms and models of the observables and their partials with respect to the solve-for parameters. Such complex tasks are susceptible to programming errors, round-off errors and compiler errors. The design goal of HELIOSAT has not been efficient algorithms, but rather numerical accuracy and stability. This thesis will give an overview of the libraries, published results, details of selected algorithms and will discuss a few design decisions.

1.1 HELIOSAT - thesis overview

John D. Anderson at the Jet Propulsion Labratory (JPL) suggested the idea to develop an independent orbit determination program. The development of the software started during the author's stay at JPL as a resident research assistant in 2001/2002. It is now written entirely in Fortran 95. The Fortran 95 specification does not include object oriented features like for example inheritance or templates, which may lead to a large code base if we need the same functions for different data types. On the positive side, the syntax in Fortran programs is very similar to mathematical expressions, it has unique array support and strict pointer semantics that make it easy to write, debug and maintain reasonably performing code.

The author has in a few cases implemented tools readily available elsewhere to achieve high numerical precision and stability. Most noticeable are the ordinary differential solver and several batch and sequential estimation techniques. Subroutines to interpolate JPL Planetary ephemerides and to transform between the International Terrestrial Reference Frame (ITRF) and International Celestial Reference Frame (ICRF) have also been implemented. For example both the JPL[1] and International Earth Rotation Service (IERS) (McCarthy & Petit 2003) software suffered from significant round-off errors when using specific combination of compilers and CPUs, see section 2.1.

An overview of the libraries in HELIOSAT is presented in figure 1.1. It shows the names of the libraries, their contents and interdependencies. Repesentation of numbers and numerical precision is discussed in chapter 2. Chapter 3 introduces HELIOSAT's ODE solver with error control and variable step size and order. Chapter 4 summarizes the estimation algorithms. The magnitude of different effects on the computed observables are studied in chapter 5. Finally, chapter 6 studies the effect of different troposphere models on the Doppler residuals.

1.2 Overview of publications

Three publications relevant to the development of HELIOSAT are included.

"Shape, Mean Radius, Gravity Field and Interior Structure of Ganymede" was first presented at American Astronomical Society, DPS Meeting number 33, see also Bulletin of the American Astronomical Society, Vol. 33. The authors are Anderson, J. D.[2], Jacobson, R. A.[2], Lau, E. L.[2], Moore, W. B.[3], Olsen, Ø.[4], Schubert, G.[3], Thomas, P. C.[5] and the Galileo Gravity Science Team.

The paper "Orbital resonance widths in an uniformly rotating second degree and order gravity field" was published in Astronomy & Astrophysics, Volume 449, Issue 2, April II 2006.

[1] http://iau-comm4.jpl.nasa.gov/

[2] Jet Propulsion Labratory

[3] University of California, Los Angeles

[4] University of Oslo, Norway

[5] Cornell University

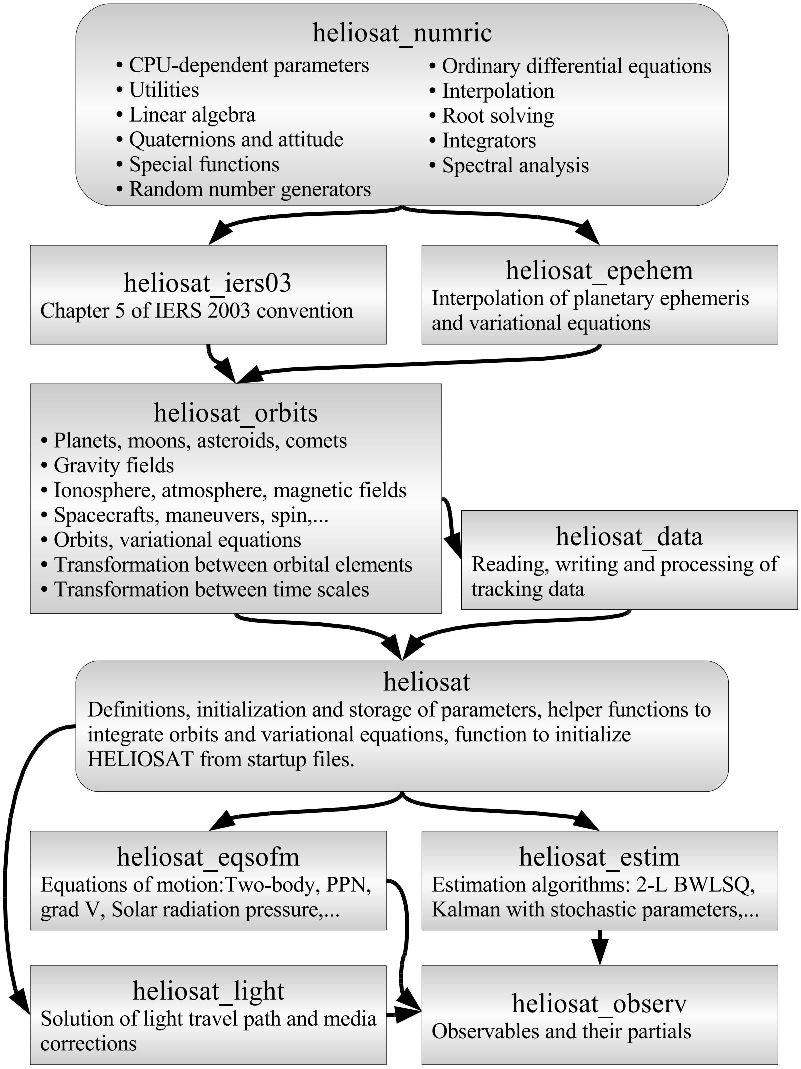
heliosat_numric
• CPU-dependent parameters
• Utilities
• Linear algebra
• Quaternions and attitude
• Special functions
• Random number generators
• Ordinary differential equations
• Interpolation
• Root solving
• Integrators
• Spectral analysis
heliosat_iers03
Chapter 5 of IERS 2003 convention
heliosat_epehem
Interpolation of planetary ephemeris and variational equations
heliosat_orbits
• Planets, moons, asteroids, comets
• Gravity fields
• Ionosphere, atmosphere, magnetic fields
• Spacecrafts, maneuvers, spin,...
• Orbits, variational equations
• Transformation between orbital elements
• Transformation between time scales
heliosat_data
Reading, writing and processing of tracking data
heliosat
Definitions, initialization and storage of parameters, helper functions to integrate orbits and variational equations, function to initialize HELIOSAT from startup files.
heliosat_eqsofm
Equations of motion:Two-body, PPN, grad V, Solar radiation pressure,...
heliosat_estim
Estimation algorithms: 2-L BWLSQ, Kalman with stochastic parameters,...
heliosat_light
Solution of light travel path and media corrections
heliosat_observ
Observables and their partials

Figure 1.1: The collection of libraries which constitutes HELIOSAT. This figure show the names of the libraries, their contents and interdependencies.

"The constancy of the Pioneer Anomalous Acceleration" was published in Astronomy & Astrophysics, Volume 463, Issue 1, February III 2007.

1.2.1 Shape, Mean Radius, Gravity Field and Interior Structure of Ganymede

This publication is a large collaborative work. Only the analysis performed by the author of this thesis is presented. A gravity field complete through degree and order four is required in order to fit the data to the noise level. John D. Anderson suggested that the cause might be mass concentrations similar to those detected on Mercury (Esposito et al. 1976). Unfortunately, JPL's Double-Precision Orbit Determination Program (DPODP) cannot easily be modified to include mass concentrations on planetary moons. We therefore attempted to fit a model consisting of point masses to the observed gravity field to see if an equivalent gravity field could be described by fewer parameters.

The nonlinear Levenberg-Marquardt algorithm (Press et al. 1992) was the first estimation method implemented from scratch in HELIOSAT. At the time, it was implemented in Mathematica, but it was removed from HELIOSAT when the libraries were converted into Fortran 95, and experience showed that very little was gained from using non-linear batch estimation methods. HELIOSAT now includes partitioned linear batch and sequential estimation algorithms.

1.2.2 Orbital resonance widths in an uniformly rotating second degree and order gravity field

Orbital resonances exist in a uniformly rotating second degree and order gravity field (Hu & Scheeres 2004). Spacecraft in orbit around a comet or an asteroid will experience such resonances. The following relation between semi-major axis of the orbit (a), the gravitational constant of the centerbody (μ) and the centerbody's rotational speed (ω_T) defines the location of a resonance:

$$n\sqrt{\frac{\mu}{a^3}} = 2\omega_T. \tag{1.1}$$

Resonances exist for every positive integer n.

This paper presents both a numerical and an analytical study of such resonances. There exist large areas of orbital instability, but retrograde orbits are in general quite stable. The overlap criteria (Lichtenberg & Lieberman 1992) explain this observation and the regions of orbital instability.

The simulations consist of several hundred thousand numerically integrated orbits. In addition to a fast ODE solver, basic routines to convert between orbital elements were required. A first attempt at simulations was implemented in Mathematica [6], but Mathematica's ODE solvers were not fast enough to complete the task in a reasonable time. Comparisons between HELIOSAT and Mathematica showed that HELIOSAT was at the time approximately 500 times faster than Mathematica even without any particular optimizations. It took almost two weeks on an Intel(R) Xeon(TM) CPU 3.20GHz processor to perform these simulations using HELIOSAT.

1.2.3 The constancy of the Pioneer Anomalous Acceleration

In the last paper, all the libraries come together. Data from the Pioneer 10 and 11 spacecraft are analyzed to study the Pioneer anomalous acceleration. Studies made by (Turyshev et al. 1999), (Anderson et al. 2002) and (Markwardt 2002) indicated an apparent anomalous acceleration acting on the Pioneer 10 and 11 spacecraft. Its magnitude is approximately $8 \times 10^{-8} cm/s^2$ directed toward the centre of the Solar System. The Pioneer 10 and 11 data set spans respectively eleven and a half years and almost four years. This paper discusses the constancy of the Pioneer anomalous acceleration. Based on the data alone, it is still not possible to exclude heat dissipation as an explanation for the anomalous acceleration. The following model is applicable if the radioisotope thermoelectric generators (RTG) are the cause of the acceleration:

$$a = a_0 \times 2^{-t/\tau}. \tag{1.2}$$

Estimation of τ results in a half-life of 117 years with a standard deviation of 16 years compared to 87 years for the half life of the ^{238}Pu in the RTGs. However, given the design of the RTGs it is difficult to explain the required anisotropic heat dissipation.

A similar analysis was performed after this paper was submitted but now with two different half-lives. The major part of the acceleration was now constant, i.e. a half-life of 1000 years. The half-life of the remaining 10% of the acceleration was just below 40 years. This is comparable to how fast the RTGs' electricity production decayed. This model fits the data slightly better, but it still leaves a constant acceleration to be explained. Fortunately, Toth & Turyshev (2006) have retrieved the entire Pioneer 10 and 11 data sets. These sets do not only contain tracking data, but also include spacecraft telemetry. With these data sets, it may be possible to

[6] http://www.wolfram.com/

discern between different acceleration models and determine whether the direction of the acceleration is towards Sun, Earth or somewhere else.

Bibliography

Anderson, J. D., Laing, P. A., Lau, E. L., Liu, A. S., Nieto, M. M., & Turyshev, S. G. 2002, Phys. Rev. D, 65, 082004

Esposito, P. B., Anderson, J. D., & Ng, A. T. Y., eds. 1976, Experimental determination of Mercury's mass and oblateness

Hu, W. & Scheeres, D. J. 2004, Planet. Space Sci., 52, 685

Lichtenberg, A. J. & Lieberman, M. A. 1992, Regular and Chaotic Dynamics (New York, NY, USA: Springer-Verlag)

Markwardt, C. B. 2002, ArXiv General Relativity and Quantum Cosmology e-prints

McCarthy, D. D. & Petit, G. 2003, IERS Conventions (2003) (Frankfurt am Main: Verlag des Bundesamts für Kartographie und Geodäsie)

Press, W. H., Teukolsky, S. A., Flannery, B. P., & Vetterling, W. T. 1992, Numerical Recipes in FORTRAN: The Art of Scientific Computing (New York, NY, USA: Cambridge University Press)

Toth, V. T. & Turyshev, S. G. 2006, ArXiv General Relativity and Quantum Cosmology e-prints

Turyshev, S. G., Anderson, J. D., Laing, P. A., Lau, E. L., Liu, A. S., & Nieto, M. M. 1999, ArXiv General Relativity and Quantum Cosmology e-prints

CHAPTER II

Numerical Precision and representation of numbers

Calculating a Doppler observable will at least involve numerically integrating the equations of motion, transforming the terrestrial coordinates of tracking stations to celestial coordinates, modelling tropospheric and plasma delays, calculating relativistic corrections and transforming between different timescales. Furthermore, determination of the parameters that influence the orbit requires the partials of the computed observables with respect to solve-for parameters and some kind of estimation scheme. A typical Doppler observable will be at approximately 2.2 GHz or 8.5 GHz depending on the frequency band (S or X-band). The Doppler noise is well below 1 mHz in parts of the Pioneer 10 tracking data. This is a difference between the observable and its noise level of 12 to 13 orders of magnitudes. At this level of accuracy, it becomes necessary to monitor the round-off error accumulated by the orbital determination software. This chapter introduces a few necessary considerations that any user of HELIOSAT should be aware of.

2.1 Compilers and operating systems

The software was developed with different compilers on different platforms to ensure that the compilers were not the cause of any errors. The combinations of platforms and compilers are:

- Compaq Alpha, OSF1 V5.1, 64 bit processor, Compaq Fortran Compiler
- Intel Pentium 4, 32-bit processor, Intel Fortran Compiler for Linux
- Intel Pentium 4, 32-bit processor, G95, Linux
- Intel Itanium, 64-bit processor, Intel Fortran Compiler for Linux

The JPL software for interpolating the planetary ephemerides suffered from significant round-off errors when using an old version of the Intel Fortran compiler. The offending code consisted of the two following lines written as a single line:

```
T(1) = REAL(NR-3, QP)*SSd(3) + SSd(1)
T(1) = (PJD(1) - T(1) + PJD(4))/SSd(3)
```

Written as a single line it resulted in a 0.5Hz diurnal signal in the residuals. The problem only appeared when using 64-bit (double) precision for floating point numbers. A similar but small compiler error appeared in the IERS software together with the G95 compiler.

Another advantage of developing the software on several systems is that it becomes easier to catch non-obvious programming errors. One programming error caused a function to evaluate the square root of a negative number. It only happened in a few highly unusual cases. The statement appeared together with other statements such that a real number was returned, and the final value of the function appeared to be correct at first glance. The correct set of compiler options would have caught the error, but there was little reason to suspect a library that had already worked for months. Moving the software from a PC to a Unix workstation allowed the error to be found within minutes. The program threw a segmentation fault and the debugging symbols allowed the offending line to be found.

HELIOSAT now works with the above compilers and operating systems. Although HELIOSAT has not been extensively tested on Microsoft Windows, it appears to work there too.

2.2 Binary representation of numbers

A floating-point number is constructed as (Landau & Páez 1997)

$$x_{float} = (-1)^s \times mantissa \times 2^{expfld-bias}, \tag{2.1}$$

where s is the sign bit. If 8 bits are used to store the exponents, then their range will be $[0, 255]$. Numbers with negative exponent are represented with a bias equal to 127 such that the range of the exponent is $[-127, 128]$. For 32-bit (single precision) numbers 23 bit of storage remains for the mantissa. It is represented as

$$mantissa = m_1 \times 2^{-1} + m_2 \times 2^{-2} + \ ... \ + m_3 \times 2^{-23} \tag{2.2}$$

Fortran's inquiry functions HUGE and TINY return the range of floating point numbers. 32-bit numbers are in the range:

$$1.2 \times 10^{-38} \lesssim x_{float} \lesssim 3.4 \times 10^{38}. \tag{2.3}$$

64-bit (double precision) and 128-bit (quadruple precision) numbers are in the ranges

$$2.2 \times 10^{-308} \lesssim x_{float} \lesssim 1.8 \times 10^{308} \tag{2.4}$$

and

$$3.4 \times 10^{-4932} \lesssim x_{float} \lesssim 1.2 \times 10^{4932}. \tag{2.5}$$

An integer parameter called QP defines the kind of floating-point numbers used by HELIOSAT. The following statement defines a floating-point number with a KIND equal to QP:

```
REAL(KIND=QP) :: var
```

All numbers are defined in this way. To ensure that the compiler uses the correct kind for all numbers in a specific statement, floating-point numbers are written as:

```
var = 3.14_QP + 0.5E+03_QP
```

instead of

```
var = 3.14 + 0.5D+03
```

The precision of a floating point number is defined as the maximum number that can be added to a number stored as 1 without changing the number:

$$1_s = 1_s + \epsilon_m \tag{2.6}$$

$\epsilon \approx 1.2 \times 10^{-7}$ for single precision, $\epsilon \approx 2.2 \times 10^{-16}$ for double precision and $\epsilon \approx 1.9 \times 10^{-34}$. Hence, it is not trivial to use single precision for Doppler tracking, since Doppler tracking requires at least a relative precision of 10^{-12} to 10^{-13}. Some or most of the variables must be stored in 2-dimensional arrays. Even double precision requires reasonable good control of the round off errors.

HELIOSAT can be run with 32-bit, 64-bit and 128-bit precision, but with decreasing speed. This is achieved by redefining the parameter QP and recompiling HELIOSAT. Using different precisions and a debugger to step through the program is an easy method to find where significant round off errors occurs. The relative accumulated round-off error is now below 10^{-14} when calculating theoretical values of the Doppler observables using double precision.

2.3 Numerical calculation of the derivative

The implemented estimation theories require the partials of the computed observables with respect to the solve-for parameters. These partials are calculated using partials of round-trip light times (equation 5.4). Partial derivatives of the precision round-trip light time ρ with respect to the parameter vector $\mathbf{q}$ due to the variations in position vector needs the value of (Moyer 2000)

$$\frac{\partial \mathbf{r}_i(t)}{\partial \mathbf{q}}, \tag{2.7}$$

which is given by the solution of the variational equations:

$$\frac{d^2}{dt^2}\left(\frac{\partial \mathbf{r}}{\partial \mathbf{q}}\right) = \frac{\partial \mathbf{a}}{\partial \mathbf{r}}\frac{\partial \mathbf{r}}{\partial \mathbf{q}} + \frac{\partial \mathbf{a}}{\partial \dot{\mathbf{r}}}\left(\frac{d}{dt}\frac{\partial \mathbf{r}}{\partial \mathbf{q}}\right) + \frac{\partial \mathbf{a}}{\partial \mathbf{q}} \tag{2.8}$$

$\mathbf{a}$ is the acceleration acting on the participating spacecraft/object. Although the variational equations can be solved directly, it might be sufficient and simpler to vary each parameter and compute the partials from the difference in trajectories. A user of HELIOSAT can set this method for selected parameters in the configuration file.

The partials are estimated numerically with the forward difference formula:

$$f'(x_0) = \frac{f(x_0+h) - f(x_0)}{h} - \frac{f''(x_0)h}{2} + \mathcal{O}(h^2) \tag{2.9}$$

The approximation error is proportional to the step size, but errors due to loss of precision will limit the achievable accuracy (Landau & Páez 1997). A simple test case is the derivative of $f(x) = \tan^{-1}(x)$ at $x = \sqrt{2}$. The exact value is 1/3. The relative error between the exact value and the forward difference algorithm is plotted as function of the stepsize in figure 2.1. The error decreases linearly with decreasing step size until $h \approx 10^{-7}$ where loss of precision causes the relative error to increase. The optimum step size can be computed given the machine precision ϵ_m:

$$\frac{f(x_0+h) - f(x_0)}{h} \approx \frac{\epsilon_m}{h} \tag{2.10}$$

The approximation error is of order h:

$$\begin{aligned} f'(x_0) &= \frac{f(x_0+h) - f(x_0)}{h} - \frac{f''(x_0)h}{2} + \mathcal{O}(h^2) \\ &= \frac{f(x_0+h) - f(x_0)}{h} + \frac{x_0\, h}{(1+x_0)^2} + \mathcal{O}(h^2), \end{aligned} \tag{2.11}$$

and at $x = \sqrt{2}$

$$\epsilon_{ap}^{f(x_0)} \approx \left.\frac{x\, h}{(1+x^2)^2}\right|_{x=\sqrt{2}} \approx 0.16h. \tag{2.12}$$

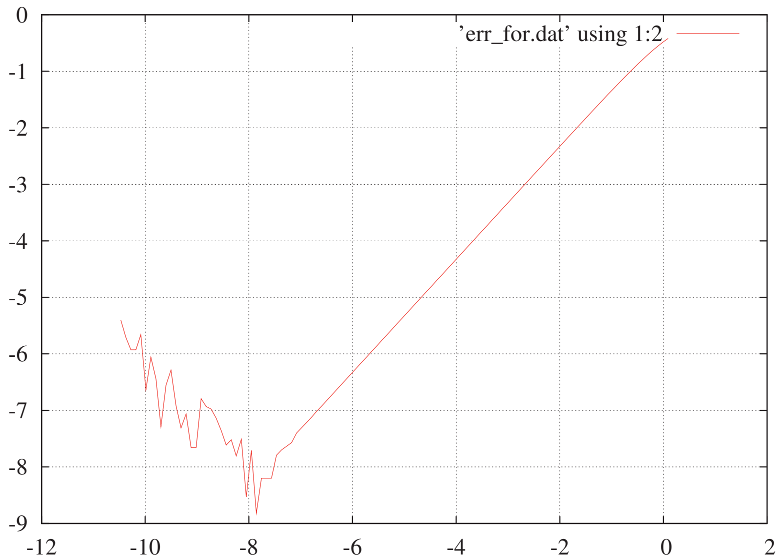

Figure 2.1: A log-log plot of the relative error as a function of step size for the forward difference algorithm with 64-bit precision.

Equations 2.12 and 2.10 yield the optimum step size for the forward difference algorithm:

$$h \approx \sqrt{\frac{\epsilon_m}{0.16}} \tag{2.13}$$

The optimum step sizes are $h_s \approx 0.0009$, $h_d \approx 4\times10^{-8}$ and $h_q \approx 3\times10^{-17}$ for 32-bit, 64-bit and 128-bit floating point numbers. A good choice for double precision is a relative step size of 10^{-7}. Users of HELIOSAT must define the step size if using the finite difference method to calculate the partials.

2.4 Other considerations

There are many more things to consider when writing software to solve numerical problems. HELIOSAT includes software to invert matrices, interpolate series, calculate the gradient of a spherical harmonic expansion, finding roots of equations, etc. Even a task as straightforward as inverting a matrix requires special care if the matrix is almost singular, but well-known methods like singular value decomposition (Press et al. 1992) work very well. A detailed discussion of all the algorithms in HELIOSAT is not needed, since entire books have been written about computational physics. There is not room for a detailed discussion in a document such as this either. The monograph by Moyer (2000), which is 550 pages long, is mostly contained within two libraries. Just printing the software itself would require more than a thousand pages. The next three chapters will therefore present only what is needed to use HELIOSAT.

Bibliography

Landau, R. H. & Páez, M. J. 1997, Computational Physics, Problem Solving with Computers (New York, NY, USA: John Wiley and Sons)

Moyer, T. D. 2000, Formulation for Observed and Computed Values of Deep Space Network Data Types for Navigation (California Institute of Technology)

Press, W. H., Teukolsky, S. A., Flannery, B. P., & Vetterling, W. T. 1992, Numerical Recipes in FORTRAN: The Art of Scientific Computing (New York, NY, USA: Cambridge University Press)

CHAPTER III

Ordinary differential equation solvers

The equations of motion are usually formulated as a set of ordinary differential equations. This chapter describes the ordinary differential equation solver in HELIOSAT. It is a Nordsieck method with automatic error control and with automatic changes in both order and step size. HELIOSAT's ODE-solver is tested against two of the ODE-solvers provided in Numerical Recipes (Press et al. 1992).

3.1 Multistep methods

A general first order ordinary differential equation is

$$\frac{dy}{dt} = f(t, y) \tag{3.1}$$

Questions about the existence of solutions are covered in any text on differential equations, see for example Boyce & DiPrima (1992).

A multistep uses the solutions and derivatives at the preceding points to estimate the solution at t_{n+1} (Kahaner et al. 1989):

$$y_n = \sum_{i=1}^{k} \alpha_i y_{n-i} + h \sum_{i=0}^{k} \beta_i f(t_{n-i}, y_{n-i}), \tag{3.2}$$

where the α's and β's are coefficients and h is the step size. The method is explicit if $\beta_0 = 0$, otherwise it is implicit. An implicit method requires estimates of y_{n+1} to compute y_{n+1}. Functional iteration[1] or Newton's method are popular algorithms for solving this problem. Methods were $\alpha_2 = \alpha_3 \cdots = 0$ are Adams methods. Explicit methods are called Adams-Bashforth while implicit methods are called Adams-Moulton. The coefficients of a three step Adams-Moulton are determined by requiring that the multistep method in-

[1] The technique is known as predictor-corrector.

tegrate the five equations $y' = k\,t^{k-1}$:

$$\begin{aligned} y(t) &= 1 & 1 &= \alpha_1 \\ y(t) &= t & 1 &= \beta_0 + \beta_1 + \beta_2 + \beta_3 \\ y(t) &= t^2 & 1 &= 2\,[\beta_0 + 0 - \beta_2 - 2\beta_3] \\ y(t) &= t^3 & 1 &= 3\,[\beta_0 + 0 + \beta_2 + 4\beta_3] \\ y(t) &= t^4 & 1 &= 4\,[\beta_0 + 0 - \beta_2 - 8\beta_3] \end{aligned} \tag{3.3}$$

These equations are derived from equation 3.2 using $h = 1$, $t_0 = 0$ and $n = 0$. Flaherty (2006) derives general formulas for the coefficients of Adams-Bashforth and Adams-Moulton methods. The coefficients for k-step Adams-Bashforth methods are:

$$\beta_j = \sum_{i=j-1}^{k-1} \gamma_i \binom{i}{j-1} \tag{3.4}$$

where

$$\gamma_i = (-1)^i \int_0^1 \binom{\tau}{i} d\tau \tag{3.5}$$

The local error term is

$$d_n = \gamma_k h^{k+1} y^{k+1}(\eta), \quad \eta \in (t_{n-1}, t_n) \tag{3.6}$$

The corresponding coefficients for Adams-Moulton methods are:

$$\beta^j = \sum_{i=j-1}^{k-1} \gamma_i^* \binom{i}{j-1} \tag{3.7}$$

where

$$\gamma_i^* = (-1)^i \int_0^1 \binom{-\tau+1}{i} d\tau \tag{3.8}$$

and

$$d_n = \gamma_k^* h^{k+1} y^{k+1}(\eta), \quad \eta \in (t_{n-1}, t_n) \tag{3.9}$$

There are several weaknesses with k-step methods, since they require knowledge of the solution at the previous $k - 1$ steps. Hence, there is no elegant way to start a multistep method and it is difficult to change step size and/or order for the purpose of error control.

3.2 Nordsieck methods

Norsieck methods use a Taylor series around a point t_n and propagate the coefficients to t_{n+1} in contrast to multistep methods which use the solution at t_n, t_{n-1},... The method is easiest to understand with an example (Press et al. 1992). The Nordsieck vector for a four-value method is

$$\mathbf{y}_n \equiv \begin{pmatrix} y_n \\ hy'_n \\ (h^2/2)y''_n \\ (h^3/6)y'''_n \end{pmatrix} \tag{3.10}$$

Now, use the Taylor expansion of $y(t)$ around t_n and differentiate to get estimates of y_{n+1}, y'_{n+1}, y''_{n+1} and y'''_{n+1}:

$$\tilde{\mathbf{y}}_{n+1} = B\mathbf{y}_n \tag{3.11}$$

where B is the Pascal upper triangular matrix P:

$$B_{ij} = \begin{cases} \binom{j}{i} & i \leq j \\ 0 & \text{otherwise} \end{cases} \tag{3.12}$$

The final approximation of $\mathbf{y}_{n+1}$ is

$$\mathbf{y}_{n+1} = \tilde{\mathbf{y}}_{n+1} + \alpha\mathbf{l}, \tag{3.13}$$

where $\mathbf{l} = (l_0, l_1, l_2, l_3)^T$ is a specific vector for each method. α is defined by the equation

$$y'_{n+1} = f(t_{n+1}, y_{n+1}). \tag{3.14}$$

Therefore

$$l_1 = 1, \qquad \alpha = hf(t_{n+1}, y_{n+1}) - h\tilde{y}'_{n+1} \tag{3.15}$$

It can be proven that each Nordsieck method is equivalent to a multistep method (Ramanantoanina 2005). The generating polynomials for the Nordsieck and multistep methods are:

$$\begin{aligned} \rho(t) &= \det(tI - B)\mathbf{e}_1^T(tI - B)^{-1}\mathbf{l} \\ \sigma(t) &= \det(tI - B)\mathbf{e}_0^T(tI - B)^{-1}\mathbf{l} \end{aligned} \tag{3.16}$$

where $\mathbf{e}_0 = (1, 0, 0, \cdots, 0)^T$ and $\mathbf{e}_1 = (0, 1, 0, \cdots, 0)^T$. These equations can be written as

$$\rho(t) = \det \begin{pmatrix} t-1 & l_0 & -1 & \cdots & \cdots & -1 \\ 0 & l_1 & -2 & \cdots & \cdots & -k \\ \vdots & 0 & \ddots & \vdots & \cdots & \vdots \\ \vdots & \vdots & 0 & t-1 & \cdots & \vdots \\ \vdots & \vdots & \vdots & & \ddots & \vdots \\ 0 & l_k & 0 & \cdots & \cdots & t-1 \end{pmatrix} \tag{3.17}$$

and

$$\sigma(t) = \det \begin{pmatrix} l_0 & -1 & -1 & \cdots & \cdots & -1 \\ l_1 & t-1 & -2 & \cdots & \cdots & -k \\ \vdots & 0 & \ddots & \vdots & \cdots & \vdots \\ \vdots & \vdots & 0 & t-1 & \cdots & \vdots \\ \vdots & \vdots & \vdots & & \ddots & \vdots \\ l_k & 0 & 0 & \cdots & \cdots & t-1 \end{pmatrix} \tag{3.18}$$

The two first components of equation (3.10) satisfy the linear multistep equation (3.2). If a vector l can be found such that $\sigma(t)$ is the generating vector for a multistep method, then equation (3.13) is the corresponding Nordsieck method.

A simple example demonstrates how to find the Nordsieck vector corresponding to an Adam-Moulton method. The generating polynomial for a 4-step Adams-Moulton algorithm is

$$\begin{aligned} p &= \beta_0 t^3 + \beta_1 t^2 + \beta_2 t + \beta_3 \\ &= \frac{3}{8}t^3 + \frac{19}{24}t^2 - \frac{5}{24}t + \frac{1}{24} \end{aligned} \tag{3.19}$$

Equation (3.18) provides the polynomial as a function of $\mathbf{l}$ with $k = 3$. Comparison of powers of t gives three equations with three unknowns, l_0, l_2 and l_3. l_1 equals 1 by definition. The solution is

$$\mathbf{l} = \left(\frac{3}{8}, 1, \frac{3}{4}, \frac{1}{6}\right)^T \tag{3.20}$$

The same method can be used to find the l-vector for Adams-Bashforth methods. The **l**-vector that is equivalent to a k-step Adams-Moulton, is also the **l**-vector for a $(k-1)$-step Adams-Bashforth method with the exception of l_0, which equals zero.

Table 3.1 provides the **l**-vectors for several Adams-Moulton-Bashforth methods. The **l**-vectors were computed with Mathematica.

Table 3.1: The **l**-vectors for 2 to 11-value methods. These methods are equivalent to Adams-Bashforth-Moulton multi step methods. $\gamma_k^* k!$ with $\nabla a_{n,k+1}$ determines the local approximation error.

k	2	3	4	5	6	7	8	9	10	11
l_0	$\frac{1}{2}$	$\frac{5}{12}$	$\frac{3}{8}$	$\frac{251}{720}$	$\frac{95}{288}$	$\frac{19087}{60480}$	$\frac{5257}{17280}$	$\frac{1070017}{3628800}$	$\frac{25713}{89600}$	$\frac{26842253}{95800320}$
l_1	1	1	1	1	1	1	1	1	1	1
l_2		$\frac{1}{2}$	$\frac{3}{4}$	$\frac{11}{12}$	$\frac{25}{24}$	$\frac{137}{120}$	$\frac{49}{40}$	$\frac{363}{280}$	$\frac{761}{560}$	$\frac{7129}{5040}$
l_3			$\frac{1}{6}$	$\frac{1}{3}$	$\frac{35}{72}$	$\frac{5}{8}$	$\frac{203}{270}$	$\frac{469}{540}$	$\frac{29531}{30240}$	$\frac{6515}{6048}$
l_4				$\frac{1}{24}$	$\frac{5}{48}$	$\frac{17}{96}$	$\frac{49}{192}$	$\frac{967}{2880}$	$\frac{267}{640}$	$\frac{4523}{9072}$
l_5					$\frac{1}{120}$	$\frac{1}{40}$	$\frac{7}{144}$	$\frac{7}{90}$	$\frac{1069}{9600}$	$\frac{19}{128}$
l_6						$\frac{1}{720}$	$\frac{7}{1440}$	$\frac{23}{2160}$	$\frac{3}{160}$	$\frac{3013}{103680}$
l_7							$\frac{1}{5040}$	$\frac{1}{1260}$	$\frac{13}{6720}$	$\frac{5}{1344}$
l_8								$\frac{1}{40320}$	$\frac{1}{8960}$	$\frac{29}{96768}$
l_9									$\frac{1}{362880}$	$\frac{1}{72576}$
l_{10}										$\frac{1}{3628800}$
$-\gamma_k^* k!$	$\frac{1}{6}$	$\frac{1}{4}$	$\frac{19}{30}$	$\frac{9}{4}$	$\frac{863}{84}$	$\frac{1375}{24}$	$\frac{33953}{90}$	$\frac{57281}{20}$	$\frac{3250433}{132}$	$\frac{1891755}{8}$

The local error for a k-step implicit method can be estimated from

$$\begin{aligned} d_n &\approx \gamma_k^* h^{k+1} y^{(k+1)}(t_n) + \mathcal{O}(h^{k+2}) \\ &\approx \gamma_k^* k! \, \nabla a_{n,k+1} \end{aligned} \tag{3.21}$$

where

$$\nabla a_{n,k+1} = \frac{h^k}{k!}(y_n^{(k)} - y_{n-1}^{(k)}) = \frac{h^{k+1}}{k!} \frac{y_n^{(k)} - y_{n-1}^{(k)}}{k} \approx \frac{h^{k+1}}{k!} y^{k+1}(t_n) \tag{3.22}$$

It can be kept at a predetermined level ϵ by changing the step size from h to αh where

$$\alpha \approx \left(\frac{\epsilon}{|\gamma_k^* k! \nabla a_{n,k+1}|} \right)^{1/(k+1)} \tag{3.23}$$

Changing the step size from h to αh requires only that the components of the $\mathbf{y}_n$ vector are rescaled with the appropriate powers of α. HELIOSAT keeps the error relative to the magnitude of y_n by scaling the equation (3.21) with $|y_{n-1}| + |hy'_{n-1}|$.

HELIOSAT stores an additional component in the Nordsieck-vector to enable order increases. That component is

$$\frac{h^{k+1}y_n^{k+1}}{(k+1)!} \approx \frac{\nabla a_{n,k+1}}{k+1} \tag{3.24}$$

It is computed from equation (3.22) after each successful step. The component has a double purpose: It is used to estimate the step size with the increased order and as a start value of component number $k+1$ if the integrator decides to increase the order. HELIOSAT also checks if decreasing the order increases the step size. Component $k-1$ is already in the Nordsieck vector and requires no extra storage. HELIOSAT does not try to change order before at least $k-1$ successful steps at order number k. Tests performed on two-body orbits using HELIOSAT show that this is the optimum rate of order changes. The maximum order of HELIOSAT's multivalue ordinary differential equation solver is 10.

Starting the integrator is simple. The differential equations provide $h\,y_0'$, which is enough to start a two-value method. The order control mechanism increases the order to the appropriate level. Finally, the error of a two-value method provides an estimate of the initial step size:

$$h_0 = \sqrt{\frac{2\epsilon}{|y'(t_0)|}} \tag{3.25}$$

3.3 Testing the ODE-solvers

This section compares the author's ODE-solver to Numerical Recipes' Runge-Kutta and Burlish-Stoer methods. These two methods have variable step size and error control. The Runge-Kutta method is the classical fourth order method. Burlisch-Stoer has no order associated with the algorithm.

Identical two-body orbits were integrated 10 times with each ODE-solver using the same compiler and compiler settings. The required accuracy after the final step was 10^{-11}. Table 3.2 shows the time in took for each ODE-solver to integrate the two-body orbit and how many times the force function was called from the ODE-solver. The Runge-Kutta method is by far the slowest method and requires an order of magnitude more function calls than the Nordsieck ODE-solver. The number of function calls determines the time it takes to solve the initial value problem if a complex force function is used. In fact, HELIOSAT's ODE-solver is more than three times faster for satellites in low Earth orbit than Numerical Recipes' implementation of the Burlisch-Stoer method.

Table 3.2: Comparing different ODE-solvers. The table shows the time it took for each ODE-solver to integrate the two-body orbit and how many times the force function was called from the ODE-solver.

Method	Time (ms)	Functions calls
Multivalue	5.20 ± 0.03	3529
Burlisch-Stoer	3.85 ± 0.07	11321
Runge-Kutta	16.41 ± 0.3	49280

The Burlisch-Stoer method has one major problem with interpolation between steps. Figure 3.1 shows the 10-logarithm of distance between the interpolated position and the exact position. The absolute accuracies at each step are comparable for both methods, but the errors in the interpolated positions using the Burlisch-Stoer solution (red) is approximately 5 orders of magnitude larger than the errors using the Nordsieck solutions. These large interpolation errors happen because the Burlisch-Stoer method has no natural interpolating function and since each step is very large. The magnitude of the interpolation errors in the Burlisch-Stoer solution is independent of polynomial, rational function and cubic spline interpolation. The Nordsieck method is based on Taylor expansion and polynomial interpolation is a natural choice unless the entire Nordsieck vector is stored.

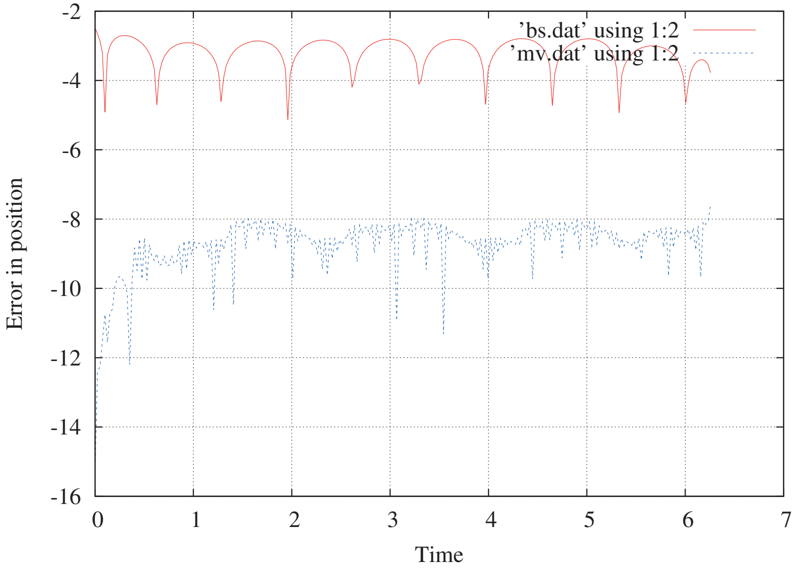

Figure 3.1: 10-logarithm of the distance between interpolated positions and the exact positions using the Burlisch-Stoer solution (red) and using the Nordsieck solution (blue).

Bibliography

Boyce, W. E. & DiPrima, R. C. 1992, Elementary differential equations and boundary value problems (New York, NY, USA: John Wiley and Sons)

Flaherty, J. E. 2006, Course Notes - Ordinary Differential Equations, CSCI-6820 / MATH-6820, http://www.cs.rpi.edu/~flaherje/

Kahaner, D., Moler, C., & Nash, S. 1989, Numerical methods and software (New Jersey, USA: Prentice-Hall)

Press, W. H., Teukolsky, S. A., Flannery, B. P., & Vetterling, W. T. 1992, Numerical Recipes in FORTRAN: The Art of Scientific Computing (New York, NY, USA: Cambridge University Press)

Ramanantoanina, A. 2005, Nordsieck Methods and General Linear Methods for Ordinary Differential Equations, African Institute for Mathematical Sciences, http://www.aims.ac.za/resources/archive/2004/ar.ps

CHAPTER IV

Batch and Sequential Estimation Theory

The estimation algorithms used in HELIOSAT are all derived in the course "Batch and Sequential Estimation Theory" at the Institute of Theoretical Astrophysics, University of Oslo. This chapter starts with a short description of the purpose of an estimation theory and then describes the different estimation algorithms implemented in HELIOSAT. HELIOSAT includes two Bayesian Weighted Least Squares (BWLSQ) algorithms and two sequential filtering algorithms. Rauch-Tung-Striebel smoothing is available for both of the sequential filtering algorithms.

4.1 Introduction

Assuming a set of m independent observations $\tilde{\mathbf{z}}$ and an observational model depending on a set of n parameters $\mathbf{x}$:

$$\tilde{\mathbf{z}} = \mathbf{z}\left(\mathbf{x}\right) + \epsilon, \tag{4.1}$$

where ϵ is the noise vector. The purpose of an estimation algorithm is to maximize the a posteriori probability density function $P(\mathbf{x}|\tilde{\mathbf{z}})$. It follows from Bayes theorem (Larsen & Marx 1986) that

$$\begin{aligned} P(\mathbf{x}|\tilde{\mathbf{z}}) =& \frac{\tilde{P}\left(\mathbf{x}\right)}{P\left(\tilde{\mathbf{z}}\right)} P(\tilde{\mathbf{z}}|\mathbf{x}) \\ \propto & \exp\left(-\frac{1}{2}\left(\tilde{\mathbf{z}}-\mathbf{z}\right)^T \tilde{P}_z^{-1}\left(\tilde{\mathbf{z}}-\mathbf{z}\right)^T\right. \\ & \left. -\frac{1}{2}\left(\tilde{\mathbf{x}}-\mathbf{x}\right)^T \tilde{P}_x^{-1}\left(\tilde{\mathbf{x}}-\mathbf{x}\right)^T\right) \end{aligned} \tag{4.2}$$

if both the a priori estimate of the parameters $\tilde{\mathbf{x}}$ and the probability density function $P(\tilde{\mathbf{z}}|\mathbf{x})$ are multi-normal. Maximizing $P(\mathbf{x}|\tilde{\mathbf{z}})$ is equivalent to minimizing

$$\Phi\left(\mathbf{x}\right) = -\frac{1}{2}\left(\tilde{\mathbf{z}}-\mathbf{z}\right)^T \tilde{P}_z^{-1}\left(\tilde{\mathbf{z}}-\mathbf{z}\right)^T - \frac{1}{2}\left(\tilde{\mathbf{x}}-\mathbf{x}\right)^T \tilde{P}_x^{-1}\left(\tilde{\mathbf{x}}-\mathbf{x}\right)^T. \tag{4.3}$$

4.2 Bayesian Weighted Least Squares

Any minima of the function given by equation (4.3) requires that

$$\frac{\partial \Phi}{\partial \mathbf{x}} = \mathbf{0}. \tag{4.4}$$

A simple application of the Newton-Rapson method gives the following iterative algorithm:

$$\hat{\mathbf{x}}_{k+1} = \hat{\mathbf{x}}_k + \Delta\hat{\mathbf{x}}_{k+1} \tag{4.5}$$

where

$$\begin{aligned}
\Delta\hat{\mathbf{x}}_{k+1} &= \left(N_k + \tilde{W}_x\right)^{-1} \left(\mathbf{U}_k + \tilde{W}_x \left(\tilde{\mathbf{x}} - \hat{\mathbf{x}}_k\right)\right) \\
N_k &= A_k^T \tilde{W}_z A_k \\
\mathbf{U}_k &= A_k^T \tilde{W}_z \left(\tilde{\mathbf{z}} - \mathbf{z}_k\right) \\
A_k &= \frac{\partial \mathbf{z}}{\partial \hat{\mathbf{x}}_k} \\
\hat{\mathbf{x}}_0 &= \tilde{\mathbf{x}}
\end{aligned} \tag{4.6}$$

The weight matrices $\tilde{W}_z$ and $\tilde{W}_x$ are the inverse of the a priori covariance matrices $\tilde{P}_z$ and $\tilde{P}_x$. The covariance matrix for $\Delta\hat{\mathbf{x}}$ is:

$$\hat{P}_x = \left(N_k + \tilde{W}_x\right)^{-1} \tag{4.7}$$

A condition for stopping the iteration is necessary. Iterating to convergence is a waste of time since the minimum is only a statistical estimate of the parameters $\mathbf{x}$ (Press et al. 1992). Although guidelines based on, for example, changes in the chi-square from one iteration to the next are possible, but it will ultimately be the experience of the analyst which determines when to stop iterating.

The reduced chi-square is an important indicator of the validity of the model and the assigned observational errors. Its definition is

$$\chi_r^2 = \frac{\chi^2}{m - n} \tag{4.8}$$

where

$$\chi^2 = \sum_{i=1}^{m} \frac{(\tilde{z}_i - z_i(\hat{\mathbf{x}}_k))^2}{\tilde{\sigma}_z^2} \tag{4.9}$$

m is total number of observations and n is the number of solve-for parameters. If $\chi_r^2 \gg 1$ then the assigned observational errors are too small or the model is not appropriate. If $\chi_r^2 \ll 1$ then the assigned observational errors are to large or too many parameters allow an artificially good fit.

4.3 2-level BWLSQ

Some of the solve-for parameters are probably not constant over the entire trajectory of the spacecraft. Drag coefficients, media corrections, clock parameters, etc, can change rapidly. A simple solution is to divide the entire set of data into intervals where each solve-for parameter is approximately constant. It is trivial to find an estimate using the BWLSQ algorithm, but it may require inversion of very large matrices. A better approach is to divide the state vector into common parameters and arc parameters:

$$\Delta\mathbf{x}^T = (\Delta\mathbf{x}_c^T, \Delta\mathbf{x}_a^T) \tag{4.10}$$

where

$$\Delta\mathbf{x}_a^T = (\Delta\mathbf{x}_{s1}^T, \ldots, \Delta\mathbf{x}_{sk}^T, \ldots, \Delta\mathbf{x}_{sK}^T) \tag{4.11}$$

and where $\Delta\mathbf{x}_{sk}^T$ are the solve-for parameters in subarc k. The A-matrix now becomes:

$$A = \begin{pmatrix} A_{c1} & A_{a1} & 0 & 0 & 0 & 0 & 0 \\ A_{c2} & 0 & A_{a2} & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ A_{ck} & 0 & 0 & 0 & A_{ak} & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ A_{cK} & 0 & 0 & 0 & 0 & 0 & A_{aK} \end{pmatrix} \tag{4.12}$$

where

$$A_{ck} = \frac{\partial\mathbf{z}_k}{\partial\mathbf{x}_c} \tag{4.13}$$

and

$$A_{ak} = \frac{\partial\mathbf{z}_k}{\partial\mathbf{x}_{ak}} \tag{4.14}$$

The 2-level version of the BWLSQ algorithm is found by inserting the above matrices into equation 4.6:

$$\begin{aligned} \Delta\hat{\mathbf{x}}_a = & \left[\sum_{k=1}^{K}\left(N_{a_k} - N_{as_k}(N_{s_k} + \tilde{W}_{s_k}^{-1})N_{as_k}^T\right) + \tilde{W}_a\right]^{-1} \\ & * \left[\sum_{k=1}^{K}\left(\mathbf{U}_{ak} - N_{as_k}(N_{s_k} + \tilde{W}_{s_k}^{-1})(\mathbf{U}_{sk} + \tilde{W}_{s_k}(\tilde{\mathbf{x}}_{s_k} - \hat{\mathbf{x}}_{s_k}))\right)\right. \\ & \left. + \tilde{W}_a(\tilde{\mathbf{x}}_a - \hat{\mathbf{x}}_a)\right] \end{aligned} \tag{4.15}$$

and

$$\hat{P}_a = \left[\sum_{k=1}^{K} \left(N_{a_k} - N_{as_k} (N_{s_k} + \tilde{W}_{s_k}^{-1}) N_{as_k}^T \right) + \tilde{W}_a \right]^{-1} \tag{4.16}$$

Each arc is sequentially processed after $\Delta\hat{\mathbf{x}}_a$ and $\hat{P}_a$ have been estimated:

$$\Delta\hat{\mathbf{x}}_{s_k} = (N_{s_k} + \tilde{W}_{s_k})^{-1} \left[\mathbf{U}_{s_k} + \tilde{W}_{s_k} (\tilde{\mathbf{x}}_{s_k} - \hat{\mathbf{x}}_{s_k}) - N_{as_k}^T \Delta\hat{\mathbf{x}}_a \right] \tag{4.17}$$

and

$$\hat{P}_{s_k} = (N_{s_k} + \tilde{W}_{s_k})^{-1} + (N_{s_k} + \tilde{W}_{s_k})^{-1} N_{as_k}^T \hat{P}_a N_{as_k} (N_{s_k} + \tilde{W}_{s_k}) \tag{4.18}$$

This technique is attractive if there are hundreds of solve-for parameters. Instead of inverting the large N-matrix, K small $(N_{s_k} + \tilde{W}_{s_k})$ matrices and one $(N'_a + \tilde{W}_a)$ must be inverted. This saves both time and limits the numerical round-off errors.

4.4 Kalman Filter

A typical Kalman filter (Kalman 1960) consists of two stages. The first stage is the time update equations. These equations predict the state at observation number $m + 1$ based on the estimate $\Delta\hat{\mathbf{x}}_m$, P_{x_m} from processing m observations.

$$\begin{aligned} \Delta\tilde{\mathbf{x}}_{m+1} =& \Phi(t_{m+1}, t_m) \Delta\hat{\mathbf{x}}_m + \mathbf{e}_m \\ \tilde{P}_{x_{m+1}} =& \Phi(t_{m+1}, t_m) \hat{P}_{x_m} \Phi^T(t_{m+1}, t_m) + Q_m \end{aligned} \tag{4.19}$$

where

$$\begin{aligned} \mathbf{e} =& \int_{t_m}^{t_{m+1}} \Phi(\tau, t_m) \mathbf{w}_m(\tau) d\tau \\ Q_m =& \int_{t_m}^{t_{m+1}} \int_{t_m}^{t_{m+1}} \Phi(\tau, t_m) E \left[\delta\mathbf{w}_m \delta\mathbf{w}_m^T \right] \Phi^T(\tau', t_m) d\tau d\tau' \end{aligned} \tag{4.20}$$

The variational equations together with the evolution of any stochastic parameters determine the state transition matrix $\Phi(t, t_0)$. $\mathbf{w}$ is the noise driving vector in the differential equation that determines the state vector $\mathbf{x}$:

$$\dot{\mathbf{x}} = F(t)\mathbf{x}(t) + \mathbf{w}(t). \tag{4.21}$$

$F(t)$ describes the dynamical model.

The measurement update equationsxs correct the predicted state at t_{m+1} using measurement number $m+1$:

$$\begin{aligned}\Delta\hat{\mathbf{x}}_{m+1} =& \Delta\tilde{\mathbf{x}}_{m+1} + K\left[(\tilde{z}_{m+1} - z_{m+1}) - A_{m+1}\Delta\tilde{\mathbf{x}}_{m+1}\right] \\ \hat{P}_{x_{m+1}} =& \left(I - KA_{m+1}\right)\tilde{P}_{x_{m+1}}\end{aligned} \tag{4.22}$$

where the Kalman gain is

$$K = \tilde{P}_{x_{m+1}} A_{m+1}^T \left[\tilde{W}_{z_{m+1}}^{-1} + A_{m+1}\tilde{P}_{x_{m+1}} A_{m+1}^T\right]^T \tag{4.23}$$

The Kalman gain matrix says how much the current observations should count compared to the previous m observations. Equations (4.22) and (4.23) can be derived from the least squares solution based on m observations by adding one extra observation to the equations.

4.5 Pseudo-epoch state variables

The time update equation for the Kalman filter can be written as

$$\begin{pmatrix}\Delta\tilde{\mathbf{e}}\\ \Delta\tilde{\mathbf{p}}\\ \Delta\tilde{\mathbf{y}}\end{pmatrix}_{m+1} = \begin{pmatrix}\Phi_e & \Phi_p & \Phi_y\\ 0 & M & 0\\ 0 & 0 & I\end{pmatrix}\begin{pmatrix}\Delta\hat{\mathbf{e}}\\ \Delta\hat{\mathbf{p}}\\ \Delta\hat{\mathbf{y}}\end{pmatrix}_m + \begin{pmatrix}0\\ \mathbf{n}_p\\ 0\end{pmatrix}_m \tag{4.24}$$

where $\mathbf{n}_p$ is the noise. The three groups of parameters $\mathbf{e}$, $\mathbf{p}$ and $\mathbf{y}$ are modeled as dynamical, stochastic and bias parameters. Φ_e, Φ_p and Φ_y are solutions to the variational equations while M is a diagonal matrix.

Defining the pseudo-epoch state variable in terms of the current state as

$$\Delta\hat{\mathbf{e}}'_m = \Phi_e^{-1}\left(t_m, t_0\right)\left[\Delta\hat{\mathbf{e}}_m - \Phi_y\left(t_m, t_0\right)\Delta\hat{\mathbf{y}}_m\right] \tag{4.25}$$

simplifies the time update equation to

$$\begin{pmatrix}\Delta\tilde{\mathbf{e}}'\\ \Delta\tilde{\mathbf{p}}\\ \Delta\tilde{\mathbf{y}}\end{pmatrix}_{m+1} = \begin{pmatrix}I & \Phi'_p & 0\\ 0 & M & 0\\ 0 & 0 & I\end{pmatrix}\begin{pmatrix}\Delta\hat{\mathbf{e}}'\\ \Delta\hat{\mathbf{p}}\\ \Delta\hat{\mathbf{y}}\end{pmatrix}_m + \begin{pmatrix}0\\ \mathbf{n}_p\\ 0\end{pmatrix}_m \tag{4.26}$$

where

$$\begin{aligned}\Phi'_p(t_{m+1}, t_m) =& \Phi_e^{-1}(t_{m+1}, t_0)\Phi_p(t_{m+1}, t_m) \\ =& \Phi_e^{-1}(t_{m+1}, 0)\Phi_p(t_{m+1}, 0) - \Phi_e^{-1}(t_m, 0)\Phi_p(t_m, 0)\end{aligned} \tag{4.27}$$

The pseudo-epoch state formulation saves memory and operations involved in storing the potentially large matrices Φ_e and Φ_y, but the main reason for pseudo-epoch state formulation is that the dynamical part of the state vector and their observation partials refer to the initial epoch of the integrated dynamical equations.

The A matrix used in the measurement update equation mut be modified too:

$$A = (A_e, A_p^c, A_y). \tag{4.28}$$

A_e and A_y are epoch partials while A_p^c is current state partials given by:

$$A_p^c = A_p - A_e \Phi_e^{-1}(t_{m+1}, t_0)\Phi_p(t_{m+1}, t_0) \tag{4.29}$$

The final estimates of $\hat{\mathbf{e}}_M$ and $\hat{\mathbf{y}}_M$ are based on all the observations. The estimate of a stochastic parameter at t_m is not optimal since it is based only on the m preceding observations. Rauch-Tung-Striebel (Rauch et al. 1965) smoothing repeats the filtering process backward to find optimal estimates for all dynamical parameters at all epochs. HELIOSAT includes Rach-Tung-Striebel smoothing for both Kalman filters.

Bibliography

Kalman, R. E. 1960, Transactions of the ASME–Journal of Basic Engineering, 82, 35

Larsen, R. J. & Marx, M. L. 1986, An Introduction to Mathematical Statistics and Its Applications (New Jersey, USA: Prentice-Hall)

Press, W. H., Teukolsky, S. A., Flannery, B. P., & Vetterling, W. T. 1992, Numerical Recipes in FORTRAN: The Art of Scientific Computing (New York, NY, USA: Cambridge University Press)

Rauch, H. E., Tung, F., & Striebel, C. T. 1965, American Institute of Aeronautics and Astronautics, 3, 1445

CHAPTER V

Computed values of Doppler observables, media corrections and partial derivatives

The high precision tracking data from the two Pioneer spacecraft and Rosetta are Doppler observables. Total-count phase, range, GPS/TOPEX pseudo-range and carrier-phase, spacecraft interferometry, quasar interferometry and angular observables are therefore not considered in this thesis, although they can be trivially added to HELIOSAT. This chapter explains how to compute theoretical values for the Doppler observables, studies the magnitudes of the major terms involved in the calculation and introduces a new algorithm to compute ionospheric corrections .

5.1 Computed values of the Doppler observables

Moyer (2000) discusses in detail how to compute theoretical values of the Doppler observables. Therefore, only a short overview will be given here. Moyer (2000) designates the transmission time from Earth as t_1, the epoch of retransmission as t_2 and the reception time as t_3. At each of these epochs the Solar-System barycentric (SSB) positions of the uplink station $\mathbf{r}_1$, spacecraft $\mathbf{r}_2$ and downlink station $\mathbf{r}_3$ must be calculated. The range r_{ij} is defined as the magnitude of the vector $\mathbf{r}_j - \mathbf{r}_i$. Computed values for two-way and three-way Doppler observables are then found from

$$F_{2,3} = \frac{M_{2R}}{T_C} \int_{t_{3s}}^{t_{3e}} f_T(t_3) dt_3 - \frac{M_2}{T_C} \int_{t_{1s}}^{t_{1e}} f_T(t_1) dt_1, \tag{5.1}$$

where $M_{2R} = M_2 = 240/221$ for S-band communication. $f_T(t_3)$ and $f_T(t_1)$ are the transmitter frequencies at reception and transmission time at the receiving and transmitting electronics. The integration limits t_{3s} and t_{3e} are the beginning and end of each count interval (T_C) at the receiving station, while t_{1s} and t_{1e} are the start and end times of the corresponding interval at the uplink station. t_{1s} and t_{1e} can be found from t_{3s} and t_{3e} and the precision round-trip light time solutions ρ_e and ρ_s. The precision round-trip light times are calculated from:

$$\begin{aligned}\rho &= \frac{r_{23}}{c} + RLT_{23} + \frac{r_{12}}{c} + RLT_{12} \\ &- \Delta T_{t3} - \Delta T_{t1} \\ &+ \frac{1}{c}\big(\Delta_A \rho(t_3) + \Delta_{SC}\rho_{23} + \Delta_A\rho(t_1) + \Delta_{SC}\rho_{12}\big),\end{aligned} \tag{5.2}$$

where RLT_{ij} is the Shapiro delay on the up-leg and down-leg light time solutions. r_{23} is the distance between receiver at reception time and the spacecraft at retransmission time. Similarly, r_{12} is the distance between transmitter at transmission time and the spacecraft at retransmission time. c is the speed of light. $\Delta_A\rho(t_i)$ are the off-axis antenna corrections to the receiving and transmitting station on Earth. Imbriale (2002) provides the values used to calculate these corrections. $\Delta_{SC}\rho_3$ and $\Delta_{SC}\rho_1$ are the solar corona corrections on the down and up leg. ΔT_{t3} and ΔT_{t1} are the differences between Coordinated Universal Time (UTC) and Barycentric Dynamical Time (TDB) at transmission and reception time.

Troposphere and ionosphere corrections were not included in the light time solution, but added as a media correction to the Doppler observable. Media corrections for two-way and three-way computed Doppler observables are given by

$$\Delta F_{2,3} = \frac{M_2}{T_C}\left(f_T(t_{1e})\Delta\rho_e - f_T(t_{1s})\Delta\rho_s\right). \tag{5.3}$$

$\Delta\rho_e$ and $\Delta\rho_s$ are the correction to the precision round-trip light times at the end and beginning of the transmission interval. Media corrections are computed both for the up-leg and down-leg part of the light time solution. Finally, partials of the computed Doppler observables with respect to solve-for parameters $\mathbf{q}$ are required:

$$\frac{\partial F_{2,3}}{\partial \mathbf{q}} = \frac{M_2}{T_C}\left(f_T(t_{1e})\frac{\partial\rho_e}{\partial\mathbf{q}} - f_T(t_{1s})\frac{\partial\rho_s}{\partial\mathbf{q}}\right). \tag{5.4}$$

5.2 Relativistic light time delay

General relativity predicts how a massive body deflects an electro-magnetic wave. This increases the effective path length travelled by a photon, and causes a delay that must be included in the light travel time solution. The relativistic light time due to the Sun is calculated from (Moyer 2000):

$$\Delta t = \frac{(1+\gamma)\mu}{c^3}\ln\left(\frac{r_1 + r_2 + r_{12} + (1+\gamma)\mu/c^2}{r_1 + r_2 - r_{12} + (1+\gamma)\mu/c^2}\right), \tag{5.5}$$

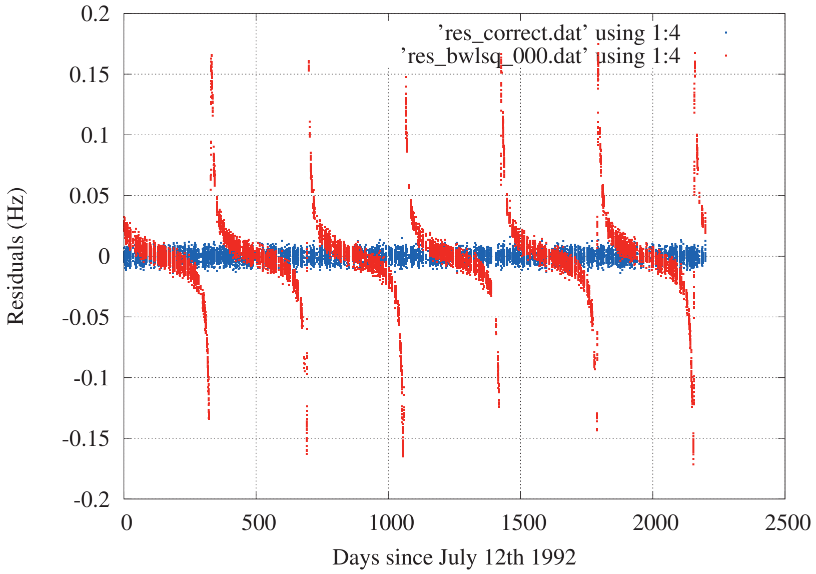

Figure 5.1: This figure shows in red the magnitude of S-band Doppler residuals if only relativistic light time delay is excluded from the estimation program. The blue residuals on the other hand include all effects and show the noise level of the Doppler data.

where μ is the gravitational constant of the Sun. γ is a parameter of the Parameterized Post-Newtonian (PPN) formulation of gravity (Will & Nordtvedt 1972). Relativistic light time delay gives in the S-band an annual signal with an amplitude of ±150mHz (Markwardt 2002).

It is possible to analyze how the different terms in the precision round-trip light time algorithm affect the Doppler residuals, but real Doppler data suffer from noise and unmodelled effects, see paper III, which may partially mask the term studied. Therefore, a set of simulated Doppler data was created using the trajectory of the Pioneer 10 spacecraft.

The output from the simulation program is compatible with JPL's Orbit Data File (ODF), and the simulated data can be analyzed with HELIOSAT's orbit determination programs in the same manner as real data. A straightforward way to study the effect of any specific term in the light travel time algorithm is to remove the corresponding code from HELIOSTAT. Figure 5.1 shows the residuals with and without relativistic light time delay. The magnitude of the Doppler residuals when including relativistic light time delay is 4.0mHz compared to 5.8mHz as seen in paper III. It is quite clear that relativistic light time delay due to the Sun is a trivial effect to account for, and that HELISOAT correctly models the relativistic light time delay.

Given the trajectory of the Pioneer spacecraft in the outer Solar System and the noise level of the Pioneer Doppler data, it is not possible to detect relativistic light time delay from any of the major planets of the Solar System. Still, HELIOSAT does compute this effect for all the planets too, but it uses a simplified version of equation (5.5):

$$\Delta t = \frac{(1+\gamma)\mu}{c^3} \ln\left(\frac{r_1 + r_2 + r_{12}}{r_1 + r_2 - r_{12}}\right) \tag{5.6}$$

In fact, use of the above approximation for the Sun's relativistic light time delay is more than sufficient given the noise level of the Pioneer Doppler data.

5.3 Solar corona corrections

Solar corona corrections are calculated from (Anderson et al. 2002):

$$\begin{aligned}\Delta_{sc}\rho = \pm & \left[A\left(\frac{R_S}{p}\right)F + B\left(\frac{R_S}{p}\right)^{1.7} e^{-(\phi/\phi_0)^2} + C\left(\frac{R_S}{p}\right)^5\right] \\ & \times \left(\frac{2.295\times 10^9}{f}\right)^2,\end{aligned} \tag{5.7}$$

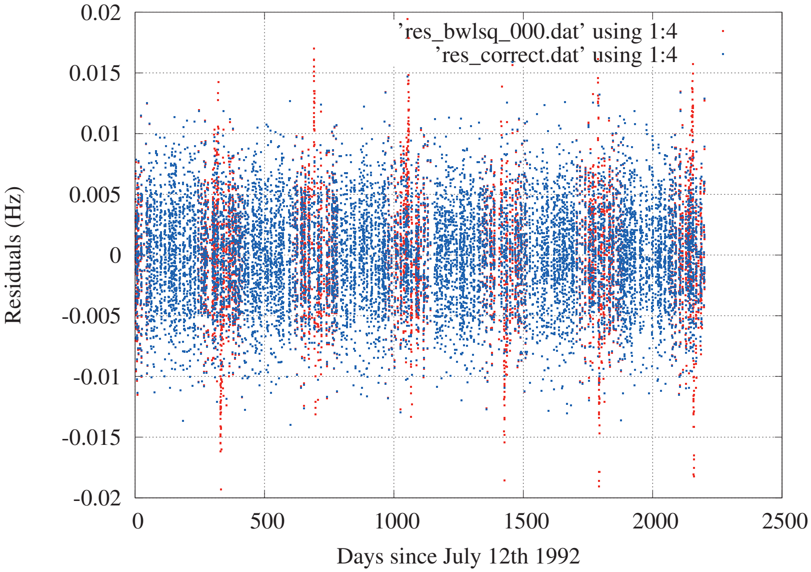

Figure 5.2: The magnitude of the Doppler residuals with and without Solar Corona corrections in the light travel time algorithm.

where A, B and C are solve-for parameters, R_S is the radius of the Sun, p is the closest approach of the light travel path to the Sun, ϕ is the latitude relative to the Sun's mean equator of date of the closest approach point to the Sun, ϕ_0 is a reference latitude of 10° and f is the carrier frequency. Solar corona studies done for the Cassini mission provide default values for the solve-for parameters. They are $A = 6.0 \times 10^3$, $B = 2.0 \times 10^4$ and $C = 0.6 \times 10^6$, all in meters.

The value of F is obtained from:

$$F = \frac{1}{\pi} \tan^{-1} \frac{\sqrt{r_2^2 - p^2}}{p} + \frac{1}{\pi} \tan^{-1} \frac{\sqrt{r_3^2 - p^2}}{p} \tag{5.8}$$

on the down-leg of the light time solution for a spacecraft. Change the index 3 to 1 to compute the F-factor on the up-leg of the light path. Moyer (2000) describes how to calculate all the necessary quantities. The sign in equation (5.7) is negative for Doppler data and positive for range data. Solar corona corrections are only computed when the tracking stations and spacecraft are on the opposite sides of the Sun.

Figure 5.2 shows that solar corona correction is not an important factor when analyzing the Pioneer tracking data.

5.4 Time scales

The time scale (ET) of the JPL ephemeris is equivalent to the IAU definition of Barycentric Coordinate Time (TBC). They differ only by an offset and a constant rate (Standish 1998). The JPL and ESA provide the Doppler data as cycles per unit of UTC time. It is therefore necessary to convert between UTC and ET. The time transformation consists of two steps:

$$UTC \rightarrow TAI \rightarrow ET. \tag{5.9}$$

IERS[1] provides the first transformation as a table. The last transformation is governed by the following differential equation (Moyer 1981b,a):

$$\frac{d\tau}{dt} = 1 - \frac{\mathcal{U}}{c^2} - \frac{1}{2}\frac{v^2}{c^2} + L, \tag{5.10}$$

where $\mathcal{U}$ is the gravitational potential at the tracking station using the positive sign convention and L is the long term average of the two terms preceding L, such that TAI and ET tick on average at the same rate.

Moyer (1981b,a, 2000) provides two different expression for calculating $ET-TAI$. The vector expression for $ET-TAI$ in the Solar System barycentric frame of reference is:

$$\begin{aligned} ET - TAI = 32.184s &+ \frac{2}{c^2}\left(\dot{\mathbf{r}}_B^S \cdot \mathbf{r}_B^S\right) + \frac{1}{c^2}\left(\dot{\mathbf{r}}_B^C \cdot \mathbf{r}_E^B\right) + \frac{1}{c^2}\left(\dot{\mathbf{r}}_E^C \cdot \mathbf{r}_A^E\right) \\ &+ \frac{\mu_J}{c^2(\mu_S + \mu_J)}\left(\dot{\mathbf{r}}_J^S \cdot \mathbf{r}_J^S\right) + \frac{\mu_{Sa}}{c^2(\mu_S + \mu_{Sa})}\left(\dot{\mathbf{r}}_{Sa}^S \cdot \mathbf{r}_{Sa}^S\right) \\ &+ \frac{1}{c^2}\left(\dot{\mathbf{r}}_S^C \cdot \mathbf{r}_B^S\right), \end{aligned} \tag{5.11}$$

where the subscripts or superscripts C, S, B, E, M, J, Sa and A represent the Solar-System barycenter, Sun, Earth-Moon barycenter, Earth, Moon, Jupiter, Saturn and the location of atomic clock on Earth which reads TAI. The expression is slightly modified if TAI is obtained from an Earth satellite:

$$ET - TAI_{SAT} = [ET - TAI]_{A=SAT} + \frac{2}{c^2}\left(\dot{\mathbf{r}}_{SAT}^E \cdot \mathbf{r}_{SAT}^E\right) \tag{5.12}$$

A few algorithms need an approximate expression for $ET - TAI$.

$$ET - TAI = 32.184 + 1.657 \times 10^3 \sin E \tag{5.13}$$

[1] http://www.iers.org/MainDisp.csl?pid=95-105

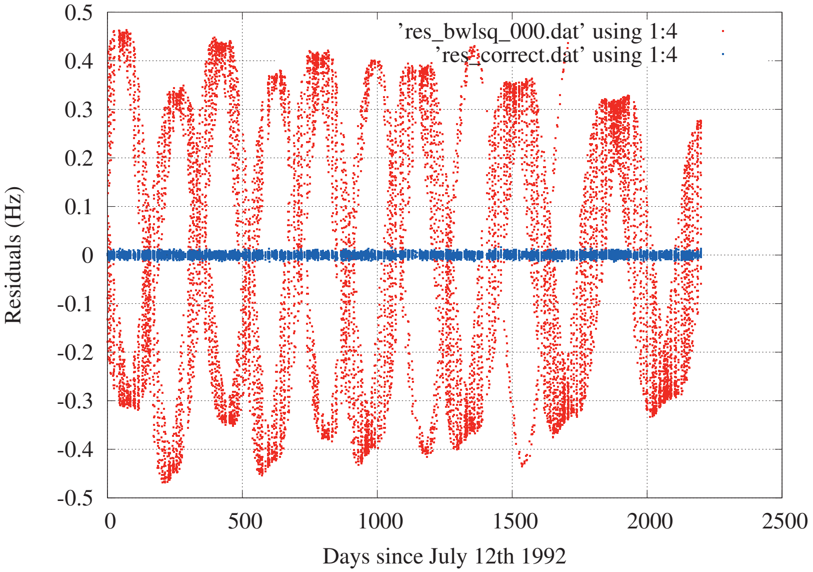

Figure 5.3: Magnitude of Doppler residuals (red) when assuming that ET ticks at the same rate as TAI.

where the eccentric anomaly for the heliocentric orbit of the Earth-Moon barycenter is approximated by (Moyer 1981a):

$$E = M + 0.0167 \sin M \tag{5.14}$$

The mean anomaly (M) in radians is calculated from:

$$M = 6.239996 + 1.99096871 \times 10^{-7} t, \tag{5.15}$$

where t is TAI or ET seconds since J2000.0.

Figure 5.3 shows the Doppler residuals in red if it is assumed that that ET ticks at the same rate as TAI. Using the vector expression for $TAI - ET$ results in blue residuals. The approximate expression for $ET - TAI$ gives residuals almost indistinguishable, i.e. less than 0.1mHz, from the residuals using the vector expression.

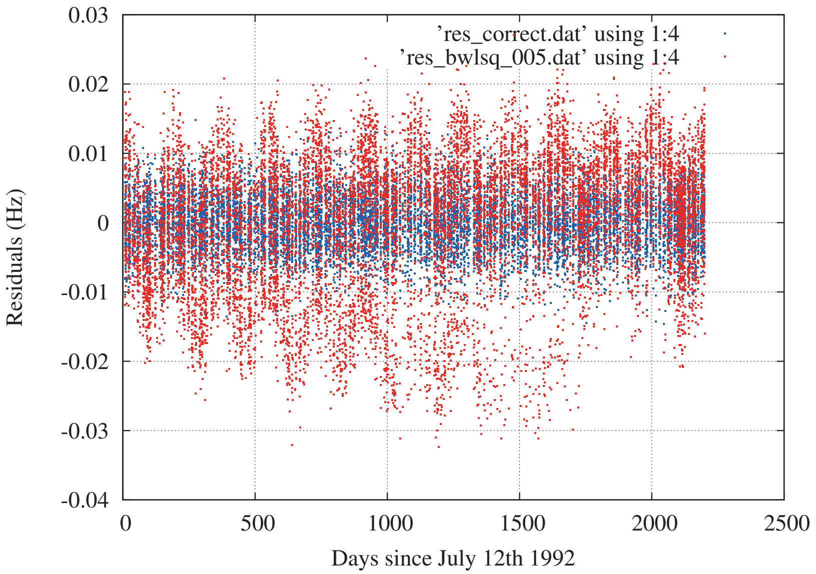

Figure 5.4: Doppler residuals due 25-meter errors in the tracking stations coordinates.

5.5 Tracking station locations

Figure 5.4 shows the Doppler residuals due to errors in the tracking station coordinates. Each of the tracking stations was moved 25 meters along the y-axis during the analysis of the simulated Doppler data. It is possible to detect errors of a few meters even with noisy S-band data. Hence, it is also necessary to include media corrections due to the ionosphere and troposphere. Orbit data files for Mars Express and Rosetta contain computed media correction for the propagation of the signal in the Earth's atmosphere, based on meteorological data observed at the ground station site. The ionosphere correction can be calculated using global maps of Earth's ionosphere available from Center for Orbit Determination (CODE) or from differential Doppler observables if available. HELISOAT can estimate the day and night components of ionosphere and corrections to the dry and wet components of the troposphere delay if no information about the ionosphere and troposphere is available.

5.6 Earth orientation

The transformation of the terrestrial reference system to the celestial reference system at the epoch t is given by (McCarthy & Petit 2003):

$$[CRS] = Q(t)R(t)W(t)[TRS] \tag{5.16}$$

$Q(t)$ defines the motion (precession-nutation figure) of the celestial pole in the celestial system. $R(t)$ and $W(t)$ are respectively rotation of the Earth around the pole and polar motion. A transformation from $[TRS]$ by applying $W(t)$ and $R(t)$ is the intermediate reference frame of epoch t. The Celestial Intermediate Pole (CIP) is the realized celestial pole according to IAU resolution B1.7, and its direction at J2000.0 is offset from the Geocentric Celestial Reference System (GCRS) specified by the IAU 2000A precession-nutation model. The motion of the CIP in GCRS is realised through the IAU 2000A precession-nutation model for periods greater than two days and through time-dependent corrections derived from astro-geodetic observations provided by the IERS. In a similar way IAU 2000A defines the polar motion $W(t)$ and Earth's rotation $R(t)$.

IERS provides a time-dependent list of observed differences between UT1 and UTC. Figure 5.5 shows typical Doppler residuals if the analyst ignores the changes in the length of day. Each discontinuity is caused by insertion of a leap second by IERS. Leap seconds are used to keep UTC within 0.9s of UT1. From figure 5.5 it appears that ignoring the secular change in UT1 results in Doppler residuals comparable to those seen in the Pioneer tracking data, but it is not a possible explanation. An error of 1 second in UT1 is equivalent to a positional error up to a few hundred meters for the tracking stations. Section 5.5 shows that such errors are easily detectable.

Polar motion has little effect on S-band Doppler residuals as can be seen from figure 5.6. The red dots show simulated Doppler residuals without polar motion in the ITRF to the ICRF transformation, while the blue dots show residuals with the full transformation.

Precession and nutation are easily detectable in Doppler residuals. Figure 5.7 shows the effect of removing the IAU 2000A precession-nutation model from ITRF to ICRF transformation (red). It is not necessary to include the observed corrections to the coordinates of CIP. They do not have a measurbale effect on the S-band residuals.

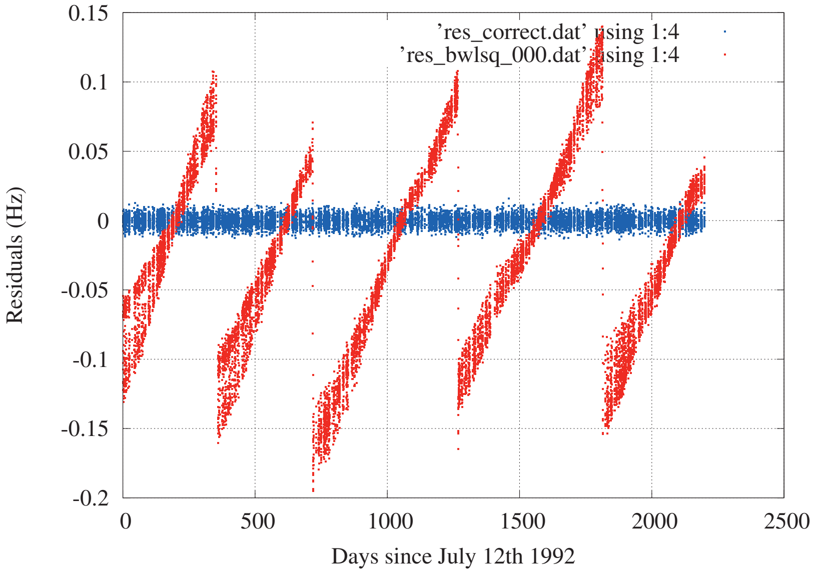

Figure 5.5: Realistic S-band Doppler residuals (blue) are compared to Doppler residuals when ignoring changes in the length of day (red). Each discontinuity is caused by insertion of a leap second by IERS.

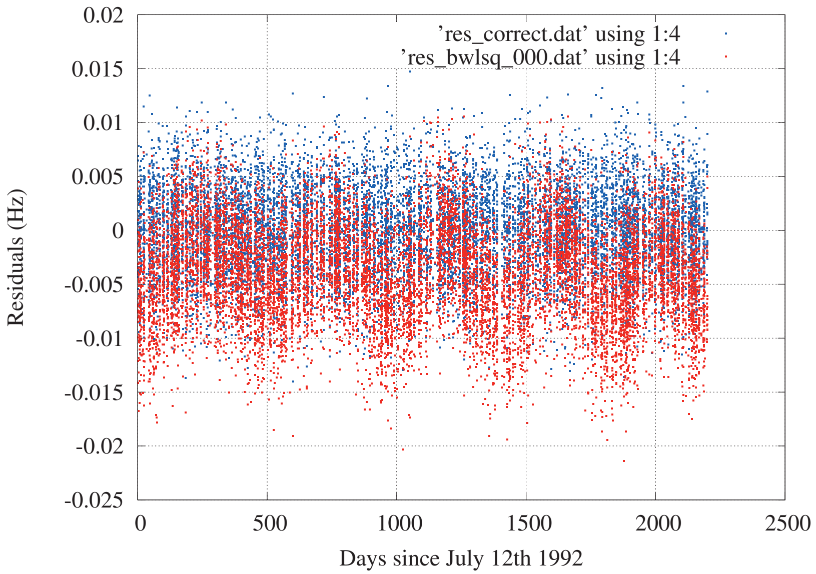

Figure 5.6: The red dots show simulated Doppler residuals without polar motion in the ITRF to the ICRF transformation, while the blue dots show residuals with the full transformation. Polar motion has little effect on the S-band Doppler residuals.

5.7 Updated ionosphere model

The delay of a radio signal through the ionosphere can be expressed by

$$\Delta t = \frac{40.3}{c\, f^2 \cos(Z)} VTEC \tag{5.17}$$

where $VTEC$ is the Vertical Total Electron Content in electrons/m^2, f is the frequency of the signal in Hz, Z is the angle between the topocentric direction to the satellite and the normal to the ionospheric layer through the pierce-point and c is the speed of light (m/s). The point on the surface of Earth below the pierce-point is the sub-ionospheric point. Both the Klobuchar model (Moyer 2000) used by JPL and the global ionosphere maps generated by Center for Orbit Determination in Europe (CODE) require the sub-ionospheric point to evaluate the total electron count.

Figures 5.8 and 5.9 display the computed vertical total electron content (VTEC) at the ionospheric pierce point in units of TECU. A global iono-

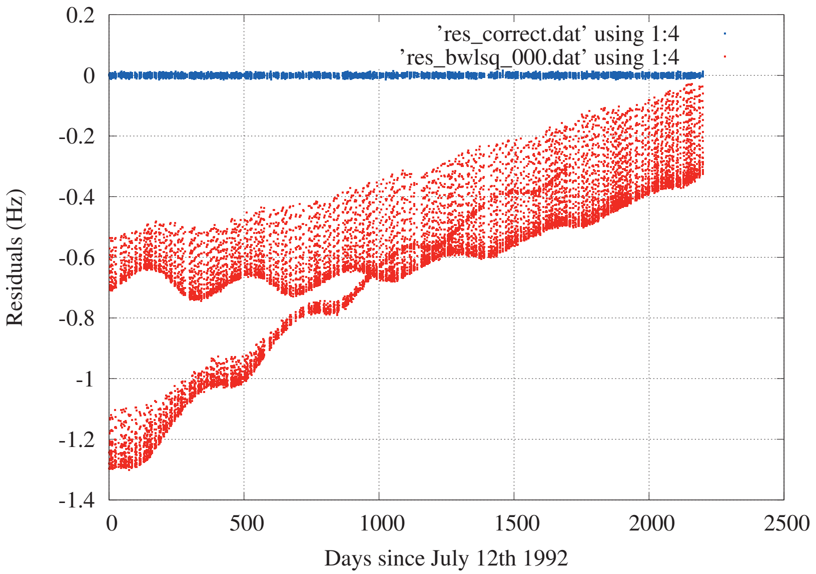

Figure 5.7: The effect of ignoring precession and nutation in the ITRF to ICRF transformation.

sphere map produced by CODE for the year 2000 and day number 178 was used to generate these figures. The coordinates represent the surface positions of emitters relative to the sub-satellite point at 75°N and 20°E with the Y-axis along the 20°E longitude line. The satellite altitude is 600km. Figures 5.8 and 5.9 show respectively the results using JPL's algorithm (Moyer 2000) and the new algorithm provided in this section. The errors from the JPL algorithm are as large as 8 TECU in this case. JPL's algorithm only fails if the line of sight passes by any of the poles. The North Pole is located at $x = 0km$ and $y \approx 1800km$ in both figures.

5.7.1 Geodetic coordinates of the sub-ionospheric point

This section contains a new algorithm for the sub-ionospheric point.

Given the un-refracted spacecraft elevation angle γ at the tracking station, compute the zenith angle Z at the mean ionospheric height of 350km from

$$Z = \sin^{-1}(0.94798\cos(\gamma)) \tag{5.18}$$

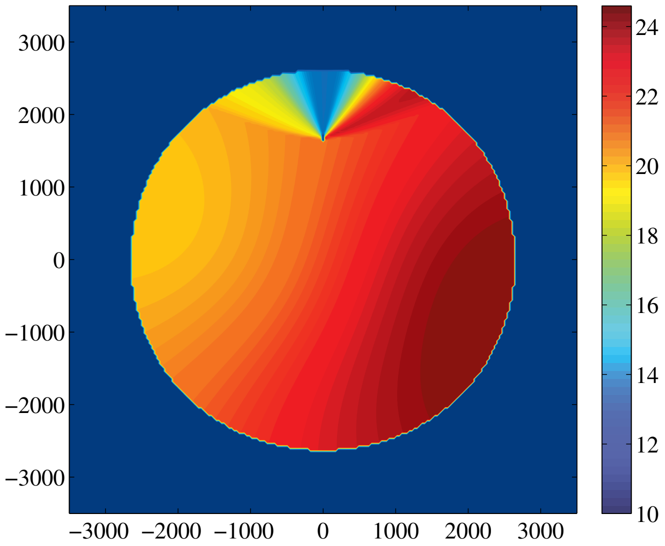

Figure 5.8: Computed vertical total electron content using Moyer's algorithm for the sub-ionospheric point. The coordinates represent the surface positions of tracking stations relative to the sub-satellite point at 75°N and 20°E with the Y-axis along the 20°E longitude line. The VTEC is given in TEC-units, while the distances are in km.

where the numerical coefficient is $a_e/(a_e + 350)$. a_e is the the equatorial radius of Earth. Calculate the angle between emitter, Earth's centre and the sub-ionospheric point from

$$\theta = \frac{\pi}{2} - Z - \gamma \tag{5.19}$$

Let ρ be the distance from Earth's centre to the emitter. The distance from the emitter to the ionospheric pierce point is

$$d = \rho\frac{\sin(\theta)}{\sin(Z)}. \tag{5.20}$$

Given the unit vector $\mathbf{L}$ directed from a tracking station on Earth towards the spacecraft and the position vector of the tracking station $\mathbf{r}_e$, both in

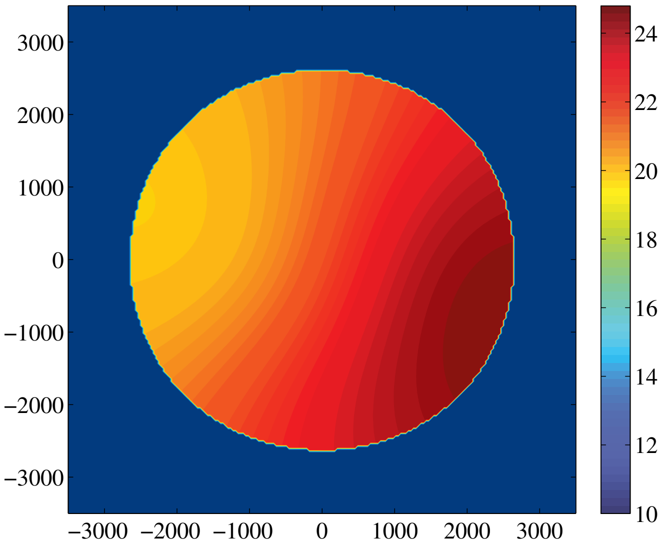

Figure 5.9: Computed vertical total electron content values using the algorithm given in this section for the sub-ionospheric point.

ITRF, calculate the ITRF position of the ionospheric pierce point as follows:

$$\mathbf{r}_i = \mathbf{r}_e + d\mathbf{L} \tag{5.21}$$

Now use the algorithms of Moyer (2000) to transform the Cartesian coordinates of the sub-ionospheric point into geodetic coordinates.

Bibliography

Anderson, J. D., Laing, P. A., Lau, E. L., Liu, A. S., Nieto, M. M., & Turyshev, S. G. 2002, Phys. Rev. D, 65, 082004

Imbriale, W. A. 2002, Large Antennas of the Deep Space Network (California Institute of Technology)

Markwardt, C. B. 2002, ArXiv General Relativity and Quantum Cosmology e-prints

McCarthy, D. D. & Petit, G. 2003, IERS Conventions (2003) (Frankfurt am Main: Verlag des Bundesamts für Kartographie und Geodäsie)

Moyer, T. D. 1981a, Celestial Mechanics, 23, 57

—. 1981b, Celestial Mechanics, 23, 33

—. 2000, Formulation for Observed and Computed Values of Deep Space Network Data Types for Navigation (California Institute of Technology)

Standish, E. M. 1998, in Bulletin of the American Astronomical Society, 1146–+

Will, C. M. & Nordtvedt, K. J. 1972, ApJ, 177, 757

CHAPTER VI

Analysis of Pioneer and Rosetta doppler data

There passed a weary time. Each throat
Was parched, and glazed each eye.
A weary time! a weary time!
How glazed each weary eye,
When looking westward, I beheld
A something in the sky.

Per Helge Andersen[1] uses numerical weather data and ray tracing to compute corrections for VLBI, GPS and satellite laser ranging observations. He provided me with light travel time delays due to the troposphere as a function of elevation, azimuth and time with resolutions in azimuth and time of 6° and 6 hours. Analysis performed by Andersen show that these ray-tracing files have accuracies better than 1cm even at low elevations. The purpose of this chapter is to study the effect of different troposphere models on the Doppler residuals.

6.1 The diurnal signal in the Pioneer Doppler residuals

Anderson et al. (2002) analyzed a limited dataset centered on the 1996 opposition and found a diurnal signal in the Pioneer Doppler residuals with amplitude of approximately $0.1mm/s$. It was attributed to a possible error in the spacecraft's orbital inclination with respect to Earth's equator. A diurnal signal was also seen by Olsen (2007). After Anderson et al. (2002) removed this term, the RMS Doppler residuals were reduced to $0.054mm/s$.

The Doppler data was reanalyzed using ray-tracing data to see if at least parts of the diurnal signal can be explained by the media corrections. These

[1] Norwegian Defense Research Establishment

datasets provide the light time delay used in equation (5.3) to compute the troposphere corrections.

Ionosphere maps from CODE were used to compute plasma corrections to the theoretical Doppler observables. Although the time spacing of these maps was as much as 24 hours in 1996, they still provide a reasonably good description of the ionosphere ,since the 1996 opposition coincided with a solar minimum. The solar minimum also helps to ensure that there was little change in the plasma between the Earth, the Sun and Pioneer 10.

The new results show no diurnal signal in the data, and the residuals are reduced to $0.034mm/s$. If the ionospheric corrections are excluded, then the residuals increase to $0.036mm/s$. The conclusion is that incorrect modeling of the troposphere causes the diurnal signal in the residuals.

6.2 Tropospheric corrections in the Rosetta data

Doppler data from the inbound part of Rosetta's Earth flyby in March 2005 were analyzed to check the sensitivity of the initial conditions to the troposphere model. Corrections due to the troposphere are included with the level 2 Rosetta Doppler data. Figure 6.1 shows the included corrections (red) and the corrections calculated using ray-tracing (blue). It includes data down to 10° elevation. Figure 6.2 shows the corresponding residuals when using the media corrections included with the Doppler data and when using ray-tracing.

Incorrect modeling of the troposphere does have an effect on the estimated orbit. The difference between the initial position using ray-tracing and included media corrections is

$$\Delta\mathbf{r} = (4.1174, -1.0425, -7.01138)km \pm (0.17, 0.06, 0.20)km, \tag{6.1}$$

with the standard deviations being identical to within 0.1% in both cases. The differences in the estimated initial coordinates range from 17 to 35 standard deviations.

A less CPU intensive algorithm for the troposphere corrections involves the NMFH2.0 and NMFW2.0 mapping functions (Niell 1996). The difference between the estimated coordinates using the Niell mapping function and ray-tracing data are

$$\Delta\mathbf{r} = (0.1114, -0.0342, 1.9239)km. \tag{6.2}$$

This is between 0.5 to 9.5 standard deviations.

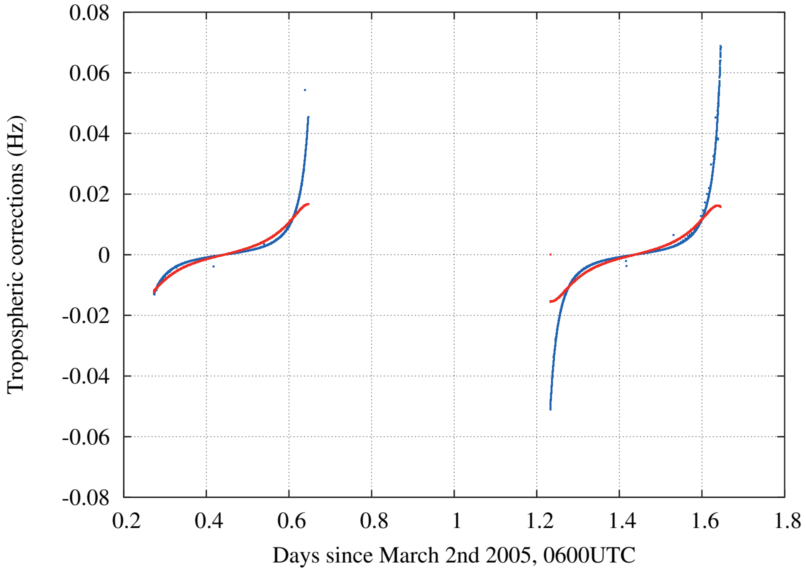

Figure 6.1: The troposphere corrections included with the Rosetta tracking data (red) and corrections computed with the ray-tracing data (blue).

To conclude, media corrections do have a significant effect on the estimated orbits. The effect will be especially important for short data arcs as will be available from flybys of asteroids, and this may result in incorrect estimates of for example the asteroids' mass and gravity field.

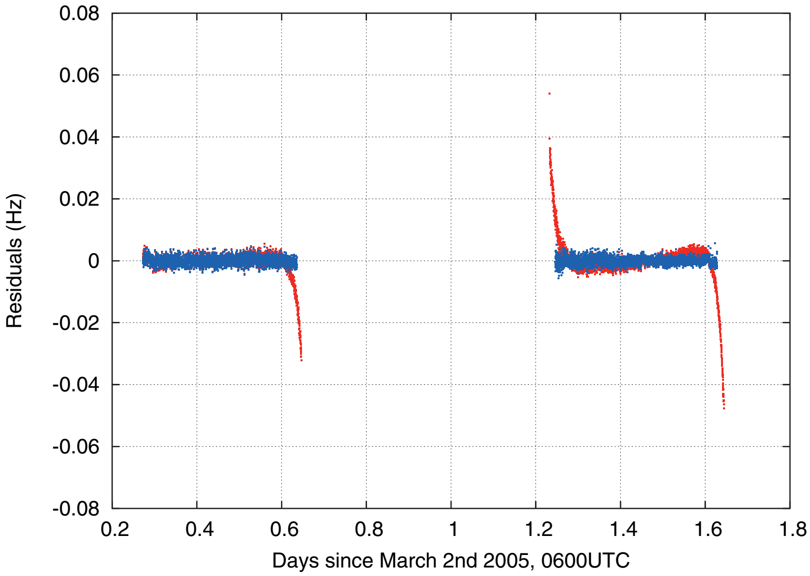

Figure 6.2: The residuals when using the media corrections included with the Doppler data (red) and when using ray-tracing calculations(blue).

Bibliography

Anderson, J. D., Laing, P. A., Lau, E. L., Liu, A. S., Nieto, M. M., & Turyshev, S. G. 2002, Phys. Rev. D, 65, 082004

Niell, A. E. 1996, J. Geophys. Res., 101, 3227

Olsen, Ø. 2007, A&A, 463, 393