APG4001S Determination of RMS between Precise and Broadcast Ephemerides for Satellite 12

Tim Marsh

10 August 2015

Contents

1 Introduction

1.1 Subject of Report

The subject if this report is to predict the ephemeris of a single GPS for one orbit. And to then compare our predicted ephemeris to the sp3 file with the correct precise one.

1.2 Background

Every satellite revives a navigation data from ground antenna, that navigation data is then relayed to the users as the navigation message. The navigation message provides all the necessary information to allow the user to perform the positioning service. It includes the Ephemeris parameters, needed to compute the satellite coordinates with enough accuracy, the Time parameters and Clock Corrections, to compute satellite clock offsets and time conversions, the Service Parameters with satellite health information.

This data is all used to refine the reviver coordinates for a specific time(in the case of refining coordinates immediately). The orbit can then be predicted using this information.

Once the orbits have been computed. they can be tested against and preexisting orbit prediction that is kept in the sp3 file.

1.3 Objectives

- Read the Navigation message file brdc2000.15n
- compute the predicted ephemeris
- compare the computed predicted ephemiris to the precise ephemiris in the sp3 file igs18540.sp3
- display the difference over the course of the orbit

1.4 Scope and Limitations

The program will be written in Python 3. all satellite data must come from the navigation message only and the sp3 file is only for comparing after the computation has happened.

2 Method

2.1 Algorithm

Broadcast ephemeris is attached as Appendix A

2.2 Python 3 Program

Attached as appendix B

3 Results

The results of the program are as follows:
(with meters difference on the Y-axis and time on the X-axis)

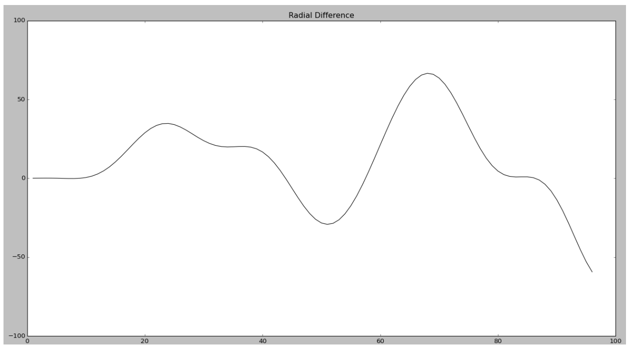

Figure 1: Sat 12, Epoch 00-00-00

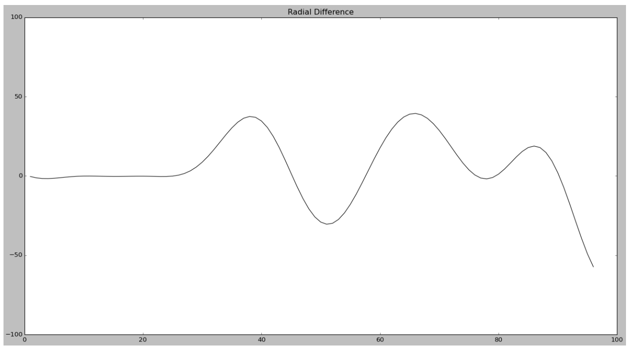

Figure 2: Sat 12, Epoch 04-00-00

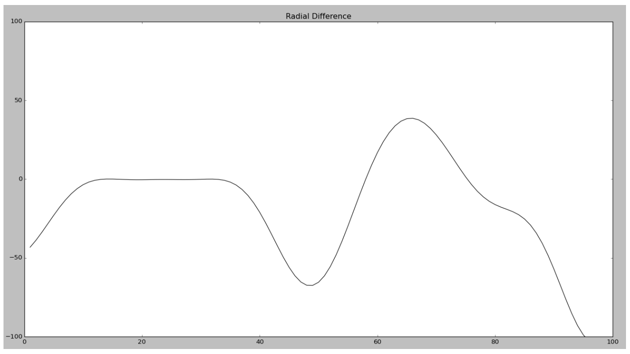

Figure 3: Sat 12, Epoch 06-00-00

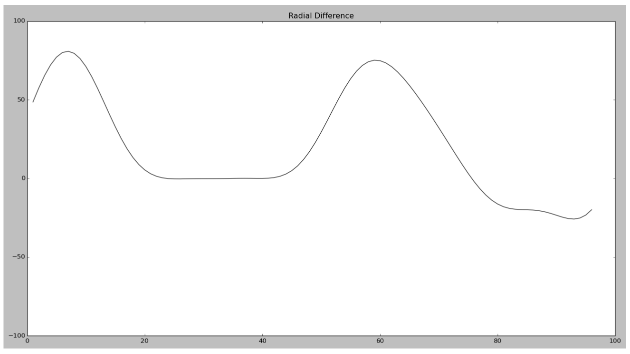

Figure 4: Sat 12, Epoch 08-00-00

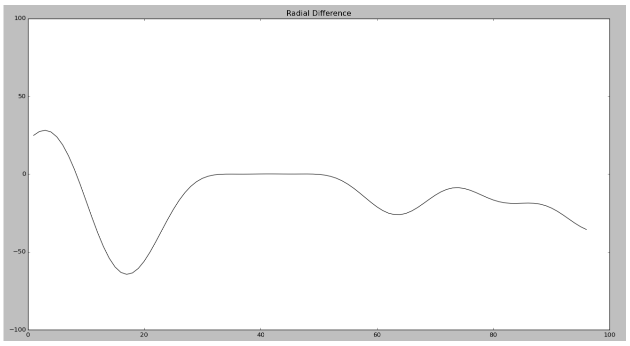

Figure 5: Sat 12, Epoch 10-00-00

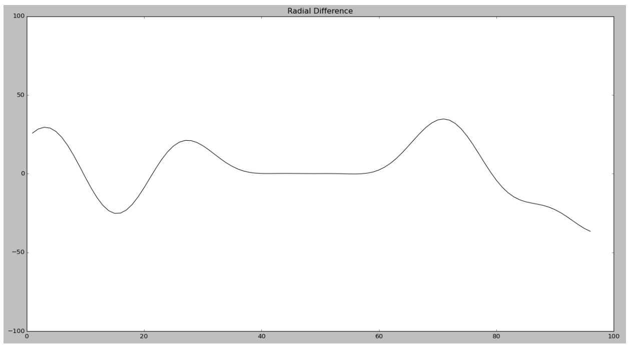

Figure 6: Sat 12, Epoch 12-00-00

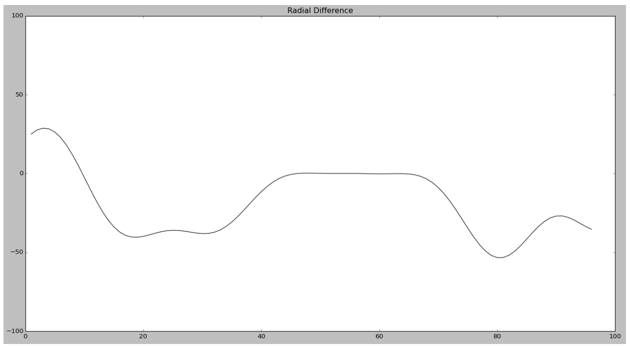

Figure 7: Sat 12, Epoch 14-00-00

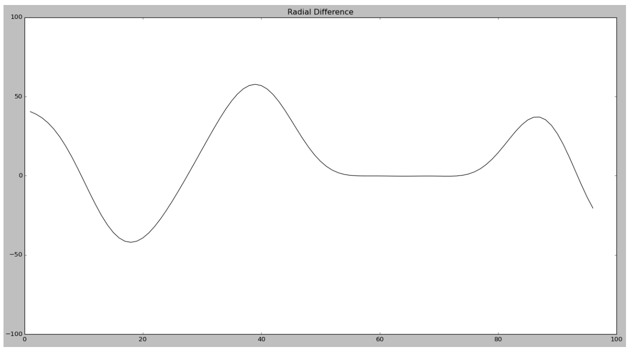

Figure 8: Sat 12, Epoch 16-00-00

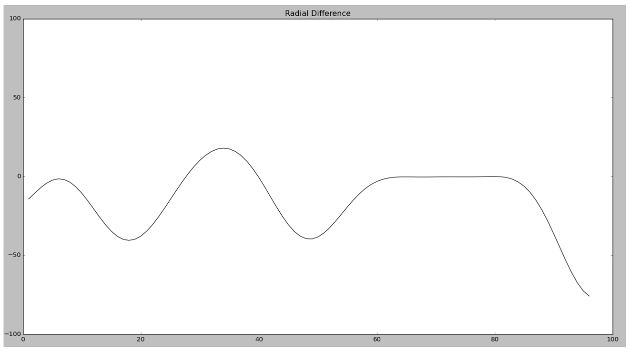

Figure 9: Sat 12, Epoch 18-00-00

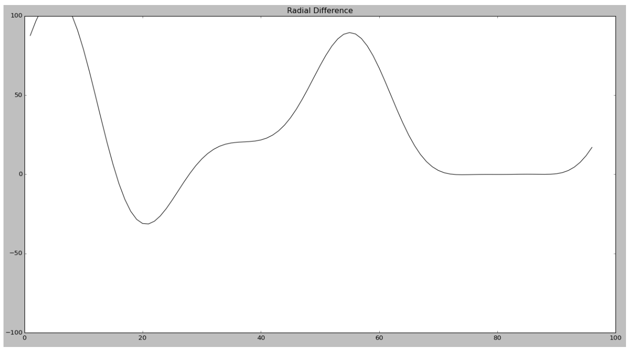

Figure 10: Sat 12, Epoch 19-59-44

The Epoch show in the description of the images is the time that the prediction as started

A Appendix A

Table 2-3. Subframe 1 Reserved Data Fields

Word	Bits
3	11-12
4	1-24
5	1-24
6	1-24
7	1-16

2.4.4 Subframes 2 and 3 - Satellite Ephemeris Data

Subframes 2 and 3 contain the ephemeris representation parameters of the transmitting satellite.

2.4.4.1 Ephemeris Parameters

Table 2-4 gives the definition of the orbital parameters using terminology typical of Keplerian or bital parameters; it is noted, however, that the transmitted parameter values are expressed in a coordinate system which allows the best trajectory fit in Earth fixed coordinates for each specific fit interval. The user will not interpret intermediate coordinate values as pertaining to any conven tional or stable coordinate system.

For each parameter contained in subframe 2 and 3, the number of bits, the scale factor of the LSB (which is the last bit received), the range, and the units are as specified in Table 2-5.

Table 2-4. Ephemeris Data Definitions

M_0	Mean Anomaly at Reference Time
Δn	Mean Motion Difference from Computed Value
e	Eccentricity
$(A)^{1/2}$	Square Root of the Semi-Major Axis
$(OMEGA)_0$	Longitude of Ascending Node of Orbit Plane at Weekly Epoch
i_0	Inclination Angle at Reference Time
ω	Argument of Perigee
OMEGADOT	Rate of Right Ascension
IDOT	Rate of Inclination Angle
C_{uc}	Amplitude of the Cosine Harmonic Correction Term to the Argument of Latitude
C_{us}	Amplitude of the Sine Harmonic Correction Term to the Argument of Latitude
C_{rc}	Amplitude of the Cosine Harmonic Correction Term to the Orbit Radius
C_{rs}	Amplitude of the Sine Harmonic Correction Term to the Orbit Radius
C_{ic}	Amplitude of the Cosine Harmonic Correction Term to the Angle of Inclination
C_{is}	Amplitude of the Sine Harmonic Correction Term to the Angle of Inclination
t_{oe}	Reference Time Ephemeris
IODE	Issue of Data (Ephemeris)

2.4.4.2 Issue of Data, Ephemeris

The Issue of Data, Ephemeris (IODE) is an 8 bit number equal to the 8 LSBs of the 10 bit IODC of the same data set. The issue of ephemeris data (IODE) term will provide the user with a convenient means for detecting any change in the ephemeris representation parameters. The IODE is provided in both subframes 2 and 3 for the purpose of comparison with the 8 LSBs of the IODC term in subframe 1. Whenever these three terms do not match, a data set cutover has occurred and new data must be collected. The transmitted IODE will be different from any value transmitted by the satellite during the preceding six hours.

Table 2-5. Ephemeris Parameters

Parameter	No. of Bits	Scale Factor (LSB)	Effective Range***	Units
IODE	8			(see text)
C_{rs}	16*	2^{-5}		meters
Δn	16*	2^{-43}		semi-circles/sec
M_0	32*	2^{-31}		semi-circles
C_{uc}	16*	2^{-29}		radians
e	32	2^{-33}	0.03	dimensionless
C_{us}	16*	2^{-29}		radians
$(A)^{1/2}$	32	2^{-19}		meters$^{1/2}$
t_{oe}	16	2^4	604,784	seconds
C_{ic}	16*	2^{-29}		radians
$(OMEGA)_0$	32*	2^{-31}		semi-circles
C_{is}	16*	2^{-29}		radians
i_0	32*	2^{-31}		semi-circles
C_{rc}	16*	2^{-5}		meters
ω	32*	2^{-31}		semi-circles
OMEGADOT	24*	2^{-43}		semi-circles/sec
IDOT	14*	2^{-43}		semi-circles/sec

* Parameters so indicated are two's complement, with the sign bit (+ or -) occupying the MSB;
** See Figure 2-8 for complete bit allocation in subframe;
*** Unless otherwise indicated in this column, effective range is the maximum range attainable with indicated bit allocation and scale factor.

Any change in the subframe 2 and 3 data will be accomplished in concert with a change in both IODE words. Cutovers to new data sets will occur only on hour boundaries except for the first data set of a new upload. The first data set may be cut-in (reference paragraph 2.4.1.1) at any time during the hour and therefore may be transmitted by the satellite for less than one hour. Additionally, the t_{oe} value, for at least the first data set transmitted by an satellite after an upload, will be different from that transmitted prior to the cutover.

2.4.4.3 Spare and Reserved Data Fields

Table 2-6 provides the locations of spare and reserved data fields within subframe 2. All spare and reserved data fields support valid parity within their respective words. Contents of spare data fields are alternating ones and zeros until they are allocated for a new function. Users are cautioned that the contents of spare data fields can change without warning.

Table 2-6. Subframe 2 Spare and Reserved Data Fields

Word	Bits	Status
10	17	Reserved
10	18-22	Spare

2.4.5 Subframes 4 and 5 - Support Data

Both subframes 4 and 5 are subcommutated 25 times each; the 25 versions of these subframes are referred to as pages 1 through 25 of each subframe. With the possible exception of "spare" pages and explicit repeats, each page contains different data in words three through ten. As shown in Figure 2-8, the pages of subframe 4 use six different formats, while those of subframe 5 use two.

A brief summary of the various data contained in each page of subframes 4 and 5 is as follows:

Table 2-15. Elements of Coordinate Systems

$A = \left(\sqrt{A}\right)^2$	Semi-major axis
$n_0 = \sqrt{\frac{\mu}{A^3}}$	Computed mean motion - rad/sec
$t_k = t - t_{oe}$*	Time from ephemeris reference epoch
$n = n_0 + \Delta n$	Corrected mean motion
$M_k = M_0 + nt_k$	Mean anomaly
$M_k = E_k - e \sin E_k$	Kepler's equation for eccentric anomaly (may be solved by iteration) - radians
$\nu_k = \tan^{-1}\left\{\frac{\sin\nu_k}{\cos\nu_k}\right\} = \tan^{-1}\left\{\frac{\sqrt{1 - e^2}\sin E_k \,/\, (1 - e\cos E_k)}{(\cos E_k - e) \,/\, (1 - e\cos E_k)}\right\}$	True anomaly
$E_k = \cos^{-1}\left\{\frac{e + \cos\nu_k}{1 + e\cos\nu_k}\right\}$	Eccentric anomaly
$\Phi_k = \nu_k + \omega$	Argument of latitude
	<u>Second Harmonic Perturbations</u>
$\delta u_k = C_{us}\sin 2\Phi_k + C_{uc}\cos 2\Phi_k$	Argument of latitude correction
$\delta r_k = C_{rc}\cos 2\Phi_k + C_{rs}\sin 2\Phi_k$	Radius correction
$\delta i_k = C_{ic}\cos 2\Phi_k + C_{is}\sin 2\Phi_k$	Correction to inclination
$u_k = \Phi_k + \delta u_k$	Corrected argument of latitude
$r_k = A(1 - e\cos E_k) + \delta r_k$	Corrected radius
$i_k = i_0 + \delta i_k + (\mathrm{IDOT})\, t_k$	Corrected inclination
$\left.\begin{aligned} x_k' &= r_k \cos u_k \\ y_k' &= r_k \sin u_k \end{aligned}\right\}$	Positions in orbital plane
$\Omega_k = \Omega_0 + \left(\dot{\Omega} - \dot{\Omega}_e\right) t_k - \dot{\Omega}_e t_{oe}$	Corrected longitude of ascending node
$\left.\begin{aligned} x_k &= x_k' \cos\Omega_k - y_k' \cos i_k \sin\Omega_k \\ y_k &= x_k' \sin\Omega_k + y_k' \cos i_k \cos\Omega_k \\ z_k &= y_k' \sin i_k \end{aligned}\right\}$	Earth-Centered, Earth-Fixed coordinates

* t is GPS system time at time of transmission, i.e., GPS time corrected for transit time (range/speed of light). Furthermore, t_k shall be the actual total time difference between the time t and the epoch time t_{oe}, and must account for beginning or end of week crossovers. That is, if t_k is greater than 302,400 seconds, subtract 604,800 seconds from t_k. If t_k is less than -302,400 seconds, add 604,800 seconds to t_k.

B Appendix B

```
###################################################################
### Read brdc file for specific satelite and compute
### ephemiris for rest of orbit (96) every 900s (15min)
###
### then check this against the sp3 file for the same
### satelite
###
### For this assignment i will be using satelite 12
###
###
###################################################################
from NavMsg import ReadNavMessage
import matplotlib as mpl
import matplotlib.pyplot as plt
import math as mt
import numpy as np

SatPRN = '12'

with open("NAVandSP3/igs18540.sp3", 'r') as f:
        Lr1 = []
        for line in f:
                if line[2:4] == SatPRN:
                        x = float(line[6:18])
                        y = float(line[18:32])
                        z = float(line[32:46])
                        rad1 = np.array([x, y, z])
                        rad1_sq = np.sqrt((rad1**2).sum())
                        Lr1 += [rad1_sq]
                        precise = np.array(Lr1) * 1000

def getEK (mk,e):
    k = 0
    while k < 4:
        if k == 0:
            ek = mk + e*np.sin(mk)
        else:
            ek = mk + e*np.sin(ek)
        k += 1
        return ek

def calcXkYkZk(CurrSat,time):


        return Xk,Yk,Zk

if __name__ == "__main__":

        sqrt = np.sqrt
        #step 1: read nav message
        Sat = ReadNavMessage(int(SatPRN))
        Epochs = []
        for i,j in Sat.items():
                Epochs.append(i)
        Epochs.sort()
        #done reading message

        print (Epochs)

        for Ep in Epochs:
                if Ep != '18-00-0':
                        continue

                lr2 = []
                count = []


                #step 2 calculate ephemeris
                CurrSat = Sat[Ep]
                e = CurrSat.e

                m = 3.986008*10**14
                omegaDot_e = 7.292115167*10**-5

                A = CurrSat.sqrtA **2

                n0 = sqrt(m/A**3)
                n = n0 + CurrSat.DeltaN
```

```
time = 0
for num in range(96):
        time = num * 900
        print(CurrSat.Toe)
        tk = time - CurrSat.Toe

        if tk > 302400.0:
                tk = tk - 604800
        if tk < -302400.0:
                tk = tk + 604800

        Mk = CurrSat.M0 + n * tk

        Ek = getEK(Mk,e)

        neum = (sqrt(1-e**2) * np.sin(Ek))/(1-e*np.cos
        denom = (np.cos(Ek) - e)/ (1 - e * np.cos(Ek))

        Vk = np.arctan( neum / denom )
        phiK = Vk + CurrSat.omega

        sigUk = CurrSat.Cus * np.sin(2*phiK) + CurrSat
        sigRk = CurrSat.Crc * np.sin(2*phiK) + CurrSat
        sigIk = CurrSat.Cic * np.sin(2*phiK) + CurrSat

        Uk = phiK + sigUk
        Rk = A * ( 1 - e * np.cos(Ek) ) + sigRk
        Ik = CurrSat.i0 +sigIk + (CurrSat.IDOT)*tk

        xk_ = Rk * np.cos(Uk)
        yk_ = Rk * np.sin(Uk)

        OHMk = CurrSat.OMEGA + (CurrSat.OMEGADOT - ome
        Xk = xk_ * np.cos(OHMk) - yk_ * np.cos(Ik) * n
        Yk = xk_ * np.sin(OHMk) + yk_ * np.cos(Ik) * n
        Zk = yk_ * np.sin(Ik)

        rad2 = np.array([Xk,Yk,Zk])
        # print(Xk,Yk,Zk)
        rad2_sqr = sqrt((rad2**2).sum())
        lr2 += [rad2_sqr]
```

```
          broadcast = np.array(lr2)


count += [CurrSat.Epoch]
length = len(count)
llr1 = length * Lr1
vllr1 = np.array(lr2)

r = precise - broadcast

counter = []
for c in range(1,97):
          counter += [c]
t = r[:96]

fig, ax = plt.subplots()
plt.plot(counter, t, '-', color='black')
plt.ylim([-100, 100])
plt.title("Difference")
plt.plot()
plt.show()
```