BATCH LEAST SQUARES DIFFERENTIAL CORRECTION
OF A GEOCENTRIC ORBIT

PART 2 - MANUAL CORRECTION WORKSHEET

Roger L. Mansfield, September 26, 1998
http://astroger.com

(Updated to PTC's Mathcad Prime 10.0 on 2024 July 29)

In this worksheet we differentially correct (DC) the orbit of an artificial Earth satellite or space probe using a test case specified in worksheet GD1, or in a worksheet derived from GD1. You should open worksheet GD1, or your own worksheet derived from GD1, and click on "Calculate Worksheet" from the Math menu now, if you have not already done so.

The process that we will use in this worksheet is documented in Refs. [1] and [2] for the differential correction of Earth orbits using radar observations, but we will use both optical and radar observations in this worksheet. The batch equation of differential correction (BEDC) is:

$$X_o' = X_o + (A^T WA)^{-1} A^T W [Y - F(X_o)].$$

Here X_o is the initial estimate of the state vector, i.e., position and velocity, at epoch t_o. X_o' is the "improved" estimate of X_o at t_o, obtained by adding to X_o the correction $(A^T WA)^{-1} A^T W [Y - F(X_o)]$.

If we let N be the number of measurements, then Y is an N-by-1 column vector. We will permit, for our problem in geocentric motion, type 5 (optical) observations whose measurements consist of topocentric right ascension (RA, or α) and declination (DEC, or δ), and type 3 (radar) observations whose measurements consist of range, azimuth, elevation, and range rate.

$F(X_o)$ is thus an N-by-1column vector of "computed" measurements. What this means is that the measurements in each observation are computed via our UPM model of two-body motion, by propagating the current estimate, X_o, to the observation times t_i for i = 1, ... , n, and by then computing the measurements at each observation time, given the specified location of the observer. We say "current estimate, X_o" because we will find it necessary to iterate on the BEDC, testing for convergence at each iteration by means of a criterion we will define below. If we have convergence on a given iteration, then we stop and convert the solution to conic elements. But if we do not have convergence, then we replace X_o by X_o' and solve the BEDC again, i.e., iterate. (We could also implement an iteration counter and stop the DC if some maximum allowable number of iterations is reached without convergence, but that is not needed here because we iterate the BEDC manually by clicking on "Calculate Worksheet".)

$[Y - F(X_o)]$ is the N-by-1column vector of residuals, in the sense "observed minus computed". The BEDC is a form of the least squares normal equations, N equations in six unknowns, which result when one answers the question, "what is a necessary condition that the weighted sum of squares of the residuals be a minimum?" Note that the type 5 residuals are not actually $\Delta\alpha$ and $\Delta\delta$, but rather $\cos\delta\ \Delta\alpha$ and $\Delta\delta$; they are the projections of Δ**L** on **A** and **D** in turn. (The $\cos\delta$ factor can become quite important when the object passes near a celestial pole, where large changes in α accompany relatively small changes in arc length in the direction of motion.) Similarly, the type 3 angular residuals are cos El ΔAz and ΔEl.

A, the "A-matrix", is the N-by-6 array of partial derivatives of the N measurements with respect to the six components of the state vector X_o. We will compute the A-matrix from the O-matrix and the G-matrix, i.e., A = OG. O is the N-by-6 matrix of partials of the measurements with respect to the state vector at observation times t_i, for i = 1, ... , n. G is Goodyear's 6-by-6 state transition matrix, i.e., the 6-by-6 matrix of partials of the state components at times t_i with respect to the state components at t_o. G is therefore a 6-by-6 Jacobian matrix defined at each observation time t_i, for i = 1, ... , n.

W is the weight matrix. Under the assumption that the measurements are Gaussian random variables, and are not correlated (Danby [3] has a good discussion of this), W is a diagonal matrix and each diagonal entry is $1/\sigma_i^2$, where σ_i^2 is the variance of measurement i. (We implement W here only for completeness; we will take W as the N-by-N identity matrix in this worksheet.)

Here now is an outline of the steps we will follow:

1. Retrieve the test case values from disk, as specified by worksheet GD1, or as specified by your own worksheet that was derived from GD1 by duplication and modification.

   Retrieval includes obtaining the initial or current estimate of state, X, and the RMS history matrix. Each time you click on "Calculate Worksheet", GDC performs another iteration of weighted, batch least squares differential correction. At each iteration the corrected values of X are written to disk along with the RMS for that iteration. The corrected values of X thus become the current state estimate for the next iteration, and the RMS history is accumulated so that you can keep track of how the DC is going.

2. Define the procedural functions needed in the DC: **C**, **FG**, **GMAT**, and **FXA**.

3. Obtain the computed measurements, **FX**, and the A-matrix, A, by invoking **FXA**.

4. Compute the residuals, ΔY, the $A^T WA$ matrix ATWA, and the $A^T W\Delta Y$ matrix, ATWΔY.

5. Solve for and apply the corrections to state, ΔX. Compute the current RMS, display the RMS history, and test for convergence.

6. Write the corrected state vector to disk and convert to conic elements.

7. Repeat Steps 1-6, by clicking on "Calculate Worksheet", until convergence is obtained.

As a preliminary, we define some constants that we will need, and set the Mathcad worksheet ORIGIN to 1 so that subscripts start at unity rather than at zero.

$DegPerRad := \frac{180}{\pi}$ **ORIGIN** $\equiv 1$

$a_e := 6378.135$ Earth's mean equatorial radius in km:

1. Retrieve the test case values from disk, as specified by worksheet GD1, or as specified by your own worksheet that was derived from GD1 by duplication and modification.

$n := \mathrm{READPRN}(\text{"NOBS.prn"})_1$	Number of observations.
$t := \mathrm{READPRN}(\text{"TVALS.prn"})$	Observation times.
$OBTYPE := \mathrm{READPRN}(\text{"OBTYPES.prn"})$	Observation types.
$N := \mathrm{READPRN}(\text{"NMEAS.prn"})_1$	Number of measurements.
$W := \mathrm{READPRN}(\text{"WEIGHTS.prn"})$	Measurement weights matrix.
$R := \mathrm{READPRN}(\text{"RVALS.prn"})$	Values of **R**.
$V := \mathrm{READPRN}(\text{"VVALS.prn"})$	Values of **V**.
$ASEZ := \mathrm{READPRN}(\text{"SEZMATS.prn"})$	Array of **SEZ** matrices.
$Y := \mathrm{READPRN}(\text{"YVALS.prn"})$	Values of Y.
$X := \mathrm{READPRN}(\text{"STATE.prn"})$	State vector (corrected by GDC).
$Epoch := \mathrm{READPRN}(\text{"EPOCH.prn"})_1$	Epoch of state vector.
$RMS := \mathrm{READPRN}(\text{"RMS.prn"})$	RMS history for state corrections by GDC (one entry for each iteration).
$k := 0.074366916133$	Set Gaussian constant for geocentric orbital motion.
$\mu := 1$	Assume that mass of secondary (artificial Earth satellite or space probe) is negligible relative to mass of primary (Earth).

$$K := k \cdot \sqrt{\mu}$$

2. Define the procedural functions needed in the DC: **C**, **FG**, **GMAT**, and **FXA**.

For path propagation one needs to calculate only c_0 through c_3, but for the state transition matrix, G, one needs c_0 through c_5. To keep down the length of this worksheet we define one version of **C**, the one that calculates c_0 through c_5. (Remember that since the ORIGIN = 1, the subscripts of the c-functions that we will use outside of the function **C** will range from 1 through 6, rather than from 0 through 5.)

$$C(x) := \begin{array}{l} N \leftarrow 0 \\ \text{while } |x| \ge 0.1 \\ \quad x \leftarrow \frac{x}{4} \\ \quad N \leftarrow N+1 \\ c_5 \leftarrow \frac{\left(1-\frac{x}{42}\cdot\left(1-\frac{x}{72}\cdot\left(1-\frac{x}{110}\cdot\left(1-\frac{x}{156}\cdot\left(1-\frac{x}{210}\cdot\left(1-\frac{x}{272}\right)\right)\right)\right)\right)\right)}{120} \\ c_4 \leftarrow \frac{\left(1-\frac{x}{30}\cdot\left(1-\frac{x}{56}\cdot\left(1-\frac{x}{090}\cdot\left(1-\frac{x}{132}\cdot\left(1-\frac{x}{182}\cdot\left(1-\frac{x}{240}\right)\right)\right)\right)\right)\right)}{24} \\ c_3 \leftarrow \frac{1}{6} - c_5 \cdot x \\ c_2 \leftarrow \frac{1}{2} - c_4 \cdot x \\ c_1 \leftarrow 1 - c_3 \cdot x \\ c_0 \leftarrow 1 - c_2 \cdot x \\ \text{while } N > 0 \\ \quad N \leftarrow N-1 \\ \quad c_5 \leftarrow \frac{\left(c_2 \cdot c_3 + c_4 + c_5\right)}{16} \\ \quad c_4 \leftarrow \frac{\left(c_2 \cdot c_2 + c_4 + c_4\right)}{8} \\ \quad c_3 \leftarrow \frac{\left(c_1 \cdot c_2 + c_3\right)}{4} \\ \quad c_2 \leftarrow \frac{c_1 \cdot c_1}{2} \\ \quad c_1 \leftarrow c_1 \cdot c_0 \\ \quad c_0 \leftarrow 2 \cdot c_0 \cdot c_0 - 1 \\ \left[c_0 \;\; c_1 \;\; c_2 \;\; c_3 \;\; c_4 \;\; c_5\right]^{T} \end{array}$$

Function UKEP solves the uniform Kepler equation for function **FG**. **FG,** in turn, propagates position and velocity for function **FXA**.

$$UKEP\left(\tau, rmag_o, \sigma_o, \alpha\right) :=$$

```
s ← τ / rmag_o
Δs ← s
while |Δs| ≥ 0.00000001
    c ← C(α·s²)
    F ← rmag_o·s·c_2 + σ_o·s²·c_3 + s³·c_4 − τ
    DF ← rmag_o·c_1 + σ_o·s·c_2 + s²·c_3
    DDF ← σ_o·c_1 + (1 − rmag_o·α)·s·c_2
    if DF ≥ 0
        m ← 1
    else
        m ← −1
    Δs ← −5·F / (DF + m·sqrt(|(4·DF)² − 20·F·DDF|))
    s ← s + Δs
s
```

$$FG\left(K, r_o, v_o, \Delta t\right) :=$$

```
τ ← K·Δt
rmag_o ← sqrt(r_o·r_o)
σ_o ← r_o·v_o
α ← 2/rmag_o − v_o·v_o
s ← UKEP(τ, rmag_o, σ_o, α)
c ← C(α·s²)
f_r ← 1 − s²·c_3·rmag_o^(−1)
g_r ← τ − s³·c_4
rmag ← rmag_o·c_1 + σ_o·s·c_2 + s²·c_3
f_v ← −s·c_2·(rmag·rmag_o)^(−1)
g_v ← 1 − s²·c_3·rmag^(−1)
[ K  α  rmag_o  f_r  f_v ]
[ τ  s  rmag    g_r  g_v ]
```

Function **GMAT** provides the state transition matrix for function **FXA**.

The state transition matrix formulation that we use below is based upon the seminal works of Goodyear [4], [5]. See also Shepperd [6], Battin [7], and Der [8] for more recent expositions.

Before defining **GMAT**, we define functions $\mathbf{S}_{11}$, $\mathbf{S}_{12}$, $\mathbf{S}_{21}$, and $\mathbf{S}_{22}$ just to make **GMAT** fit horizontally and vertically within the margins of a single Mathcad page.

$$S_{11}\left(rmag_o, rmag, f_r, g_r, f_v, g_v, s\right) := \begin{bmatrix} -\dfrac{f_v \cdot s_2 + \dfrac{f_r - 1}{rmag_o}}{rmag_o} & -f_v \cdot s_3 \\ \dfrac{\left(f_r - 1\right) \cdot s_2}{rmag_o} & \left(f_r - 1\right) \cdot s_3 \end{bmatrix}$$

$$S_{12}\left(rmag_o, rmag, f_r, g_r, f_v, g_v, s\right) := \begin{bmatrix} -f_v \cdot s_3 & -\left(g_v - 1\right) \cdot s_3 \\ \left(f_r - 1\right) \cdot s_3 & g_r \cdot s_3 \end{bmatrix}$$

$$S_{21}\left(rmag_o, rmag, f_r, g_r, f_v, g_v, s\right) := \begin{bmatrix} -f_v \cdot \left(\dfrac{s_1}{rmag_o \cdot rmag} + \dfrac{1}{rmag^2} + \dfrac{1}{{rmag_o}^2}\right) & -\dfrac{f_v \cdot s_2 + \dfrac{g_v - 1}{rmag}}{rmag} \\ \dfrac{f_v \cdot s_2 + \dfrac{\left(f_r - 1\right)}{rmag_o}}{rmag_o} & f_v \cdot s_3 \end{bmatrix}$$

$$S_{22}\left(rmag_o, rmag, f_r, g_r, f_v, g_v, s\right) := \begin{bmatrix} -\dfrac{f_v \cdot s_2 + \dfrac{g_v - 1}{rmag}}{rmag} & \dfrac{-\left(g_v - 1\right) \cdot s_2}{rmag} \\ f_v \cdot s_3 & \left(g_v - 1\right) \cdot s_3 \end{bmatrix}$$

(Note that because ORIGIN = 1, the subscripts of the c-functions and Goodyear's s-functions range from 1 to 6 rather than from 0 to 5. It is especially important to note this difference when checking the **GMAT** formulas against Goodyear's original works.)

$$GMAT(M, r_o, v_o, r, v) := \left\| \begin{array}{l} \tau \leftarrow M_{2,1} \\ \alpha \leftarrow M_{1,2} \\ s \leftarrow M_{2,2} \\ rmag_o \leftarrow M_{1,3} \\ rmag \leftarrow M_{2,3} \\ f_r \leftarrow M_{1,4} \\ g_r \leftarrow M_{2,4} \\ f_v \leftarrow M_{1,5} \\ g_v \leftarrow M_{2,5} \\ c \leftarrow C(\alpha \cdot s^2) \\ svec \leftarrow \begin{bmatrix} c_1 & s \cdot c_2 & s^2 \cdot c_3 & s^3 \cdot c_4 & s^4 \cdot c_5 & s^5 \cdot c_6 \end{bmatrix}^T \\ U \leftarrow svec_3 \cdot \tau + s \cdot svec_5 - 3 \cdot svec_6 \\ A \leftarrow \text{augment}(r, v) \\ B \leftarrow \text{augment}(r_o, v_o)^T \\ a_o \leftarrow \dfrac{-r_o}{rmag_o^{\,3}} \\ a \leftarrow \dfrac{-r}{rmag^3} \\ I \leftarrow \text{identity}(3) \\ G_{11} \leftarrow f_r \cdot I + U \cdot v \cdot a_o^{\,T} + A \cdot S_{11}(rmag_o, rmag, f_r, g_r, f_v, g_v, svec) \cdot B \\ G_{12} \leftarrow g_r \cdot I - U \cdot v \cdot v_o^{\,T} + A \cdot S_{12}(rmag_o, rmag, f_r, g_r, f_v, g_v, svec) \cdot B \\ G_{21} \leftarrow f_v \cdot I + U \cdot a \cdot a_o^{\,T} + A \cdot S_{21}(rmag_o, rmag, f_r, g_r, f_v, g_v, svec) \cdot B \\ G_{22} \leftarrow g_v \cdot I - U \cdot a \cdot v_o^{\,T} + A \cdot S_{22}(rmag_o, rmag, f_r, g_r, f_v, g_v, svec) \cdot B \\ \text{stack}(\text{augment}(G_{11}, G_{12}), \text{augment}(G_{21}, G_{22})) \end{array} \right.$$

In **FXA** we will need to calculate the computed measurements (**FX**) and the partials of the measurements at time $\mathbf{t}_i$ with respect to the state at time $\mathbf{t}_i$ (the O matrix) for observation i, for i=1, ... , n observations. The calculations depend upon whether the observation is type 5 or 3. Therefore, we define two functions, **TYPE5** and **TYPE3**, to handle the calculations for **FXA**.

$$TYPE5(i,j,r) := \left\| \begin{array}{l} \rho \leftarrow r + R^{\langle i \rangle} \\ \rho mag \leftarrow \sqrt{\rho \cdot \rho} \\ RA \leftarrow \text{angle}\left(\rho_1, \rho_2\right) \\ DEC \leftarrow \text{asin}\left(\dfrac{\rho_3}{\rho mag}\right) \\ FX \leftarrow \begin{bmatrix} \cos\left(Y_{j+2}\right) \cdot RA \\ DEC \end{bmatrix} \\ O \leftarrow \begin{bmatrix} \dfrac{-\sin(RA)}{\rho mag} & \dfrac{\cos(RA)}{\rho mag} & 0 & 0 & 0 & 0 \\ \dfrac{-\sin(DEC)\cdot\cos(RA)}{\rho mag} & \dfrac{-\sin(DEC)\cdot\sin(RA)}{\rho mag} & \dfrac{\cos(DEC)}{\rho mag} & 0 & 0 & 0 \end{bmatrix} \\ \text{augment}(FX, O) \end{array} \right.$$

Note that in **TYPE5** i is the observation index and j is the measurement index. We only need to input the position vector **r** at time t_i since the topocentric R.A. and declination measurements depend upon the space object's and sensor's positions (the sensor's position, **R**, is obtained as a "global"), but not on their velocities.

<u>Mathcad 15 vs. Mathcad Prime 10</u>

The two Mathcad 15 worksheets for this "Tracking Galileo's Earth 1 Flyby" application were hard to convert to Mathcad Prime 10, especially Gdc.xmcd.

Problem is that the Mathcad Prime 10 converter (from Mathcad 15 to Mathcad Prime) does not distinguish between vectors, say **r** = $\begin{bmatrix} x \\ y \\ z \end{bmatrix}$ and the scalar representation *r* = $\sqrt{x+y+z}$.

In Mathcad 15, one can create a Math Style "Vectors & Matrices" that is simply the style "Variables" with boldface. So Mathcad 15 treats **r** (r-bold) as a vector and *r* (r-italic) by the rules of vector calculation. But when Mathcad Prime 10 converts such a worksheet, it unbolds **r** and treats it the same as *r*. I have been able to "work around" this, but it is cumbersome.

The permanent solution, I think, would be to have in Mathcad Prime, in the Mathcad Formatting tab, an additional choice, "Vectors" that allows one to specify boldface to distinguish vectors from their scalar representation.

$$dot(A,B) := A_1 \cdot B_1 + A_2 \cdot B_2 + A_3 \cdot B_3$$

I defined this scalar multiplication function "dot" during my debugging of this worksheet, and left it in as a reminder to say something about the Mathcad Prime 10 worksheet converter treating vectors **r** and their scalar representations *r* the same.

$$TYPE3(i,j,r,v) := \begin{array}{l}
\rho \leftarrow r + R^{\langle i \rangle} \\
\rho mag \leftarrow \sqrt{\rho \cdot \rho} \\
\rho_{dot} \leftarrow v \cdot K + V^{\langle i \rangle} \\
L \leftarrow \dfrac{\rho}{\rho mag} \\
rrate_vec \leftarrow \rho_{dot} \cdot L \\
rrate \leftarrow dot\left(\rho_{dot}, L\right) \\
SEZ \leftarrow \text{submatrix}\left(ASEZ, 3 \cdot i - 2, 3 \cdot i, 1, 3\right) \\
\rho_h \leftarrow SEZ^{T} \cdot \rho \\
AZ \leftarrow \text{mod}\left(3 \cdot \pi - \text{angle}\left(\rho_{h_1}, \rho_{h_2}\right), 2 \cdot \pi\right) \\
EL \leftarrow \text{asin}\left(\dfrac{\rho_{h_3}}{\rho mag}\right) \\
FX \leftarrow \left[\rho mag \;\; \cos\left(Y_{j+3}\right) \cdot AZ \;\; EL \;\; rrate\right]^{T} \\
A \leftarrow SEZ \cdot \left[\sin(AZ) \;\; \cos(AZ) \;\; 0\right]^{T} \\
D \leftarrow SEZ \cdot \left[\sin(EL) \cdot \cos(AZ) \;\; -\sin(EL) \cdot \sin(AZ) \;\; \cos(EL)\right]^{T} \\
P \leftarrow \rho_{dot} - rrate_vec \\
O \leftarrow \begin{bmatrix}
L_1 & L_2 & L_3 & 0 & 0 & 0 \\
\dfrac{A_1}{\rho mag} & \dfrac{A_2}{\rho mag} & \dfrac{A_3}{\rho mag} & 0 & 0 & 0 \\
\dfrac{D_1}{\rho mag} & \dfrac{D_2}{\rho mag} & \dfrac{D_3}{\rho mag} & 0 & 0 & 0 \\
\dfrac{P_1}{\rho mag} & \dfrac{P_2}{\rho mag} & \dfrac{P_3}{\rho mag} & L_1 \cdot K & L_2 \cdot K & L_3 \cdot K
\end{bmatrix} \\
\text{augment}\left(FX, O\right)
\end{array}$$

Note that in **TYPE3** we need the type 3 observation count, i3, and the space object's velocity, since the computed range rate measurement depends upon velocity. We use i3 to index into the sensor velocity array, **V**, for type 3 observations, because we only calculated sensor velocities for the type 3 observations.

Function **FXA** calculates **FX**, the N-by-1computed measurements vector, and A, the N-by-6 A-matrix of partials of the measurements at time t_i with respect to the state at time t_o. (Note that in the call to procedural function **FG**, the time since epoch is converted from days to minutes by multiplying by 1440 minutes per day.)

$$FXA(K, r_o, v_o) :=$$

```
j ← 0
for i ∈ 1 .. n
    M ← FG(K, r_o, v_o, (t_i − Epoch) · 1440)
    τ ← M_{2,1}
    α ← M_{1,2}
    s ← M_{2,2}
    rmag_o ← M_{1,3}
    rmag ← M_{2,3}
    f_r ← M_{1,4}
    g_r ← M_{2,4}
    f_v ← M_{1,5}
    g_v ← M_{2,5}
    r ← f_r · r_o + g_r · v_o
    v ← f_v · r_o + g_v · v_o
    G ← GMAT(M, r_o, v_o, r, v)
    if OBTYPE_i = 5
        FXO ← TYPE5(i, j, r)
        for k ∈ 1 .. 2
            FX_{j+k} ← FXO_{k,1}
        O ← submatrix(FXO, 1, 2, 2, 7)
        j ← j + 2
    else
        FXO ← TYPE3(i, j, r, v)
        for k ∈ 1 .. 4
            FX_{j+k} ← FXO_{k,1}
        O ← submatrix(FXO, 1, 4, 2, 7)
        j ← j + 4
    if i = 1
        A ← O · G
    else
        A ← stack(A, O · G)
augment(FX, A)
```

Obtain the computed measurements, **FX**, and the A-matrix, A, by invoking **FXA**.

$$r_o := \begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix}$$

$$v_o := \begin{bmatrix} X_4 \\ X_5 \\ X_6 \end{bmatrix} \cdot \frac{1}{K}$$

$$M := FXA\left(K, r_o, v_o\right)$$

$$FX := M^{\langle 1 \rangle}$$

$$A := \text{submatrix}\left(M, 1, N, 2, 7\right)$$

(Click on the **FX** column vector and scroll down to see all N entries.)

(Click on the A matrix and scroll down to see all N rows. Scroll right to see all 6 columns.)

$$FX = \begin{bmatrix} 0.53509441 \\ 1.72538124 \\ 0.16172659 \\ -0.09129529 \\ 0.51001924 \\ 1.77174893 \\ 0.17444278 \\ -0.08706324 \\ 0.4937401 \\ \vdots \end{bmatrix}$$

$$A = \begin{bmatrix} 0.86804784 & 0.47879998 & -0.00617182 & -0.2161988 & -0.12289927 & 0.00349622 \\ 0.62435882 & -1.11510465 & -1.31778422 & -0.15494362 & 0.27990112 & 0.33466998 \\ 0.75868462 & -1.18189345 & 1.30905907 & -0.17891191 & 0.29055028 & -0.32284142 \\ 0.09578695 & -0.03276068 & 0.0743451 & 0.04099315 & 0.04643557 & -0.02053131 \\ 0.887664 & 0.44414926 & -0.03508495 & -0.20255143 & -0.10414898 & 0.00963509 \\ 0.57631235 & -1.22825559 & -1.3742512 & -0.13094094 & 0.28225292 & 0.31874691 \\ 0.76018643 & -1.23770905 & 1.3821127 & -0.16537521 & 0.27933022 & -0.31293719 \\ 0.09008717 & -0.04306445 & 0.06885445 & 0.04565066 & 0.04514879 & -0.0197498 \\ 0.90053959 & 0.41775929 & -0.05638722 & -0.19274334 & -0.09171823 & 0.01348779 \\ & & & & & \vdots \end{bmatrix}$$

4. Compute the residuals, ΔY, the $A^T WA$ matrix ATWA, and the $A^T W\Delta Y$ matrix, ATWΔY.

$$\Delta Y := Y - FX$$

$$ATWA := A^T \cdot W \cdot A$$

$$AT W\Delta Y := A^T \cdot W \cdot \Delta Y$$

$$\Delta Y = \begin{bmatrix} -0.00006341 \\ -0.00018637 \\ -0.0022384 \\ -0.00000269 \\ -0.00006327 \\ -0.00017612 \\ -0.00076507 \\ -0.00000171 \\ -0.00006287 \\ -0.00013063 \\ -0.00026644 \\ 0.00000018 \\ -0.00006332 \\ -0.0001298 \\ -0.00028689 \\ 0.00000274 \\ -0.00006217 \\ -0.00015231 \\ -0.00032036 \\ 0.00000743 \\ -0.00006038 \\ -0.00002926 \\ \vdots \end{bmatrix}$$

(Click on the ΔY column vector and scroll down to see all N entries.)

$$ATWA = \begin{bmatrix} 100.73242292 & -26.52813647 & 43.91107588 & -4.88953748 & -64.9870653 & -27.39583008 \\ -26.52813647 & 199.70148133 & -33.31644608 & 20.79665186 & 128.8128128 & 51.13137564 \\ 43.91107588 & -33.31644608 & 132.58865033 & -5.14029743 & -32.69634822 & -14.34080037 \\ -4.88953748 & 20.79665186 & -5.14029743 & 23.59807951 & 48.37234078 & 24.94057678 \\ -64.9870653 & 128.8128128 & -32.69634822 & 48.37234078 & 267.48905266 & 114.50625751 \\ -27.39583008 & 51.13137564 & -14.34080037 & 24.94057678 & 114.50625751 & 63.36510271 \end{bmatrix}$$

$$ATW\Delta Y = \begin{bmatrix} -0.0000004 \\ -0.00000095 \\ -0.00000046 \\ -0.00000036 \\ -0.00000092 \\ -0.00000057 \end{bmatrix}$$

5. Solve for and apply the corrections to state, ΔX. Compute the current RMS, display the RMS history, and test for convergence.

$$\Delta X := ATWA^{-1} \cdot ATW\Delta Y$$

$$\Delta X = \begin{bmatrix} -4.74114971 \cdot 10^{-9} \\ -4.45461098 \cdot 10^{-9} \\ -3.72644534 \cdot 10^{-9} \\ -8.3371872 \cdot 10^{-9} \\ 3.39887074 \cdot 10^{-9} \\ -0.00000001 \end{bmatrix}$$

$$X := \text{stack}\left(r_o, v_o\right) + \Delta X$$

$$WSS := \sum_{i=1}^{N} \left(W_{i,i} \cdot \Delta Y_i\right)^2$$

Weighted sum of squares of residuals.

$$WSS = 0.00009916$$

$$WRMS := \sqrt{\frac{WSS}{N}} \cdot a_e$$

Weighted RMS in km.

$WRMS = 4.58373841$

$$PWSS := \sum_{i=1}^{6} \left(ATW\Delta Y_i \cdot \Delta X_i \right)$$

Predicted weighted sum of squares of residuals for next iteration.

$PWSS = 1.41375378 \cdot 10^{-14}$

$$PWRMS := \sqrt{\frac{|WSS - PWSS|}{N}} \cdot a_e$$

Predicted weighted RMS for next iteration, in km.

$PWRMS = 4.58373841$

$$Converged := \begin{array}{l} \text{if } |WRMS - PWRMS| < 0.01 \cdot WRMS \\ \quad \| 1 \\ \text{else} \\ \quad \| 0 \end{array}$$

$Converged = 1$

$$\text{APPENDPRN}(\text{"RMS.prn"}, [WRMS \;\; Converged]) = \begin{bmatrix} 0 & 0 \\ 5.0433802 & 0 \\ 4.5837376 & 1 \\ 4.58373841 & 1 \end{bmatrix}$$

$RMS := \text{READPRN}(\text{"RMS.prn"})$

RMS History:

$$RMS = \begin{bmatrix} 0 & 0 \\ 5.043 & 0 \\ 4.584 & 1 \\ 4.584 & 1 \end{bmatrix}$$

Number of iterations:

$Iterations := \text{rows}(RMS) - 1$

$Iterations = 3$

6. Write the corrected state vector to disk and convert to conic elements.

$$\text{WRITEPRN}\left(\text{“STATE.prn”}, \text{stack}\left(\begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix}, \begin{bmatrix} X_4 \\ X_5 \\ X_6 \end{bmatrix} \cdot K\right)\right) = \begin{bmatrix} 0.82564645 \\ -0.6324892 \\ 0.49067332 \\ -0.04889401 \\ -0.10631182 \\ -0.05486373 \end{bmatrix}$$

Compute and display the conic elements by calling function **PVCO** to transform position and velocity to conic elements.

$$r_1 := \begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix} \qquad r_1 \cdot a_e = \begin{bmatrix} 5266.08454 \\ -4034.10149 \\ 3129.58065 \end{bmatrix}$$

$$v_1 := \begin{bmatrix} X_4 \\ X_5 \\ X_6 \end{bmatrix} \cdot K \qquad v_1 \cdot \frac{a_e}{60} = \begin{bmatrix} -5.19754366 \\ -11.30118540 \\ -5.83213765 \end{bmatrix}$$

PVCO invokes function SCAL1, which we define now.

$$SCAL1(K, \alpha, q, e, v) := \begin{array}{l} \text{if } \alpha > 0 \\ \quad E \leftarrow v - 2 \cdot \text{atan}\left(\dfrac{e \cdot \sin(v)}{1 + \sqrt{1 - e^2} + e \cdot \cos(v)}\right) \\ \quad s \leftarrow \dfrac{E}{\sqrt{\alpha}} \\ \text{else} \\ \quad w \leftarrow \dfrac{1}{K} \cdot \sqrt{\dfrac{q}{1 + e}} \cdot \tan\left(\dfrac{v}{2}\right) \\ \quad \text{if } \alpha = 0 \\ \quad\quad s \leftarrow 2 \cdot w \\ \quad \text{else} \\ \quad\quad E \leftarrow 2 \cdot \text{atanh}\left(\sqrt{-\alpha} \cdot w\right) \\ \quad\quad s \leftarrow \dfrac{E}{\sqrt{-\alpha}} \\ s \end{array}$$

Finally, now, we define function **PVCO**.

(Note that in **PVCO**, as defined in this document, the subscripts of the **P**, **Q**, and **W** vectors range from 1 through 3 rather than from 0 through 2. Also, the subscripts of **c** range from 1 through 4 rather than from 0 through 3.)

$$
PVCO(K, r, v) := \left\|
\begin{array}{l}
rmag \leftarrow \sqrt{r \cdot r} \\
h \leftarrow r \times v \\
hmag \leftarrow \sqrt{h \cdot h} \\
W \leftarrow \dfrac{h}{hmag} \\
E \leftarrow \dfrac{v \cdot v}{2} - \dfrac{K^2}{rmag} \\
\alpha \leftarrow -2 \cdot E \\
p \leftarrow \dfrac{hmag^2}{K^2} \\
e \leftarrow \sqrt{1.0 - \alpha \cdot p \cdot K^{-2}} \\
q \leftarrow \dfrac{p}{1+e} \\
U \leftarrow \dfrac{r}{rmag} \\
V \leftarrow W \times U \\
\upsilon \leftarrow \text{angle}\left(\dfrac{hmag}{K^2} \cdot v \cdot V - 1.0, \dfrac{hmag}{K^2} \cdot v \cdot U\right) \\
P \leftarrow \cos(\upsilon) \cdot U - \sin(\upsilon) \cdot V \\
Q \leftarrow \sin(\upsilon) \cdot U + \cos(\upsilon) \cdot V \\
i \leftarrow \text{acos}\left(W_3\right) \\
\Omega \leftarrow \text{angle}\left(-W_2, W_1\right) \\
\omega \leftarrow \text{angle}\left(Q_3, P_3\right) \\
s \leftarrow SCAL1(K, \alpha, q, e, \upsilon) \\
c \leftarrow C\left(\alpha \cdot s^2\right) \\
\Delta t \leftarrow q \cdot s + K^2 \cdot e \cdot s^3 \cdot c_4 \\
\begin{bmatrix} q \\ e \\ i \cdot DegPerRad \\ \Omega \cdot DegPerRad \\ \omega \cdot DegPerRad \\ \Delta t \end{bmatrix}
\end{array}
\right.
$$

We now invoke **PVCO** and place its output into array **CONIC**.

$$CONIC := PVCO\left(K, r_1, v_1\right)$$

$$CONIC = \begin{bmatrix} 1.14999772 \\ 2.47318712 \\ 143.00229017 \\ 103.78192276 \\ 134.87129494 \\ -0.00405756 \end{bmatrix}$$

We should note that the position vector input to **PVCO** must have units of E.R. and the velocity vector must have units of E.R. per minute. We summarize the weighted batch least squares orbital solution as follows.

$\left(CONIC_1 - 1\right) \cdot a_e = 956.70571$	Perigee height in km, relative to spherical Earth figure.
$CONIC_2 = 2.47318712$	Path eccentricity.
$CONIC_3 = 143.00229$	Path inclination, in degrees.
$CONIC_4 = 103.78192$	Right ascension of ascending node, in degrees.
$CONIC_5 = 134.87129$	Argument of perigee, in degrees.
$CONIC_6 = -0.00406$	Time of flight from perigee to epoch, in minutes.

We have the height of perigee above a spherical Earth figure, but for a closest approach determination, it would be more accurate to have the actual height of the spacecraft above its subpoint on a oblate spheroidal Earth at the instant of perigee. We calculate this now.

$f := \frac{1}{298.26}$ Earth's polar vs. equatorial flattening factor.

$e_e := \sqrt{2 \cdot f - f^2}$ Eccentricity of Earth's meridional reference ellipse.

We define procedural function **GRT**, which inputs an artificial Earth satellite's position vector and outputs the geodetic latitude of the subsatellite point (subpoint), and the satellite's height above the subpoint.

$$
GRT(r) := \left\|
\begin{array}{l}
rmag \leftarrow \sqrt{r \cdot r} \\
\delta \leftarrow \operatorname{asin}\left(\frac{r_3}{rmag}\right) \\
\phi_c \leftarrow \delta \\
\text{for } j \in 1 .. 4 \\
\quad \left\|
\begin{array}{l}
r_s \leftarrow \frac{\sqrt{1 - {e_e}^2}}{\sqrt{1 - \left(e_e \cdot \cos\left(\phi_c\right)\right)^2}} \\
\phi_s \leftarrow \operatorname{atan}\left(\frac{\tan\left(\phi_c\right)}{1 - {e_e}^2}\right) \\
H_s \leftarrow \sqrt{rmag^2 - \left(r_s \cdot \sin\left(\phi_s - \phi_c\right)\right)^2} - r_s \cdot \cos\left(\phi_s - \phi_c\right) \\
\phi_c \leftarrow \delta - \operatorname{asin}\left(\frac{H_s \cdot \sin\left(\phi_s - \phi_c\right)}{rmag}\right)
\end{array}
\right. \\
\begin{bmatrix} \phi_s \\ H_s \end{bmatrix}
\end{array}
\right.
$$

$\Delta t := -CONIC_6$ $M := FG\left(K, r_1, \frac{v_1}{K}, \Delta t\right)$

$f_r := M_{1,4}$ $g_r := M_{2,4}$

$r := f_r \cdot r_1 + g_r \cdot \frac{v_1}{K}$ $LatHt := GRT(r)$

Geodetic latitude, ϕ_s, and height above spheroid, H_s, at time of perigee passage:

$LatHt_1 \cdot DegPerRad = 25.37357$ (degrees)

$LatHt_2 \cdot a_e = 960.60847$ (km)

7. Repeat Steps 1-6, by clicking on "Calculate Worksheet", until convergence is obtained.

REFERENCES

[1] Mansfield, Roger L., "Algorithms for Reducing Radar Observations of a Hyperbolic Near Earth Flyby," *Journal of the Astronautical Sciences* (April-June 1993), pp. 249-259.

[2] Mansfield, Roger L., "Tracking Data Reduction for the Geotail, Mars Observer, and Galileo Missions," *Eleventh Space Surveillance Workshop,* MIT Lincoln Laboratory (March 31, 1993).

[Relevant material was distributed in a 19-page presentation handout, separately from the published conference proceedings, and was distributed only to attendees of the author's actual presentation. The published conference proceedings contain a six-page extended abstract of the presentation only (no equations).]

[3] Danby, J.M.A., *Fundamentals of Celestial Mechanics,* Willmann-Bell (2nd Ed. 1988), Appendix I.

[4] Goodyear, William H., "Completely General Closed-Form Solution for the Coordinates and Partial Derivatives of the Two-Body Problem," *Astronomical Journal,* Vol. 70, No. 3 (1965), pp. 189-192.

[5] Goodyear, William H., "A General Method for the Computation of Cartesian Coordinates and Partial Derivatives of the Two-Body Problem," NASA Report CR-522 (1966).

[This report, 97 pages in length, was prepared under Contract No. NAS 5-9762 by IBM for NASA's Goddard Space Flight Center, Greenbelt, Maryland.]

[6] Shepperd, Stanley W., "Universal Keplerian State Transition Matrix," *Celestial Mechanics,* Vol. 35, No. 2 (February 1985), pp. 129-144.

[7] Battin, Richard H., *An Introduction to the Mathematics and Methods of Astrodynamics,* AIAA Education Series (1987), Section 9.5.

[8] Der, Gim J., "An Elegant State Transition Matrix," *Journal of the Astronautical Sciences* (October - December 1997), pp. 371-390.