ASE 389P.4 Methods of Orbit Determination
Homework 3: The Batch and Sequential Processor

Junette Hsin
Masters Student, Aerospace Engineering and Engineering Mechanics, University of Texas, Austin, TX 78712

The numeric propagation of the state transition matrix (STM) and the batch processor were explored. A state transition matrix (STM) for a two-dimensional, two-body orbit was first generated and verified by comparing the mapped deviations to differences in the nonlinear solution. Finally, a nonlinear batch processor and the sequential processor for a simple problem were implemented.

Problem 1

Assume an orbit plane coordinate system with a gravitational parameter of 1, i.e., $\mu = 1$. The equations of motion are:

$$\begin{aligned} \ddot{x} &= -\frac{x}{r^3} \\ \ddot{y} &= -\frac{y}{r^3} \\ r^2 &= x^2 + y^2 \end{aligned}$$

Problem 1a

Generate a "true" solution by numerically integrating the equations of motion for the initial conditions:

$$\underline{X}(t_0) = \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{bmatrix}_{t=t_0} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Save the values of the state vector $\underline{X}(t_i)$ for $t_i = i \cdot 10$ time units (TU); $i = 0, \dots, 10$. Provide $\underline{X}(t_i)$ for t_1 and t_{10} in the writeup.

In your write-up, please indicate which integrator you used, what the tolerance was set to, and any other details necessary. Note, if you use a fixed time-step integrator, set the time-step to be smaller than 10 TU, but only save the data at 10 TU intervals.

Solution

$$\underline{X}(t_1) = [-0.839071529076614,\ -0.544021110889203,\ 0.544021110889159,\ -0.839071529076556]^T \tag{1}$$

$$\underline{X}(t_{10}) = [0.862318872276158,\ -0.506365641129975,\ 0.506365641129749,\ 0.862318872275784]^T \tag{2}$$

`ode45` was used to integrate with the relative tolerance set to 3e-14 and absolute tolerance set to 1e-16. The time step was 10 ms, or 100 Hz.

Problem 1b

Perturb the previous set of initial conditions by an amount

$$\underline{X}^*(t_0) = \underline{X}(t_0) - \delta\underline{X}(t_0)$$

(notice that the perturbation is subtracted!), where

$$\delta\underline{X}(t_0) = \begin{bmatrix} 1\times10^{-6} \\ -1\times10^{-6} \\ 1\times10^{-6} \\ 1\times10^{-6} \end{bmatrix}$$

Numerically integrate this "nominal" trajectory along with the associated state transition matrix to find $\underline{X}^*(t_i)$ and $\underline{\underline{\Phi}}(t_i, t_0)$ at $t_i = i \cdot 10$ TU; $i = 0, \ldots, 10$. Provide $\underline{X}^*(t_i)$ and $\underline{\underline{\Phi}}(t_i, t_0)$ at t_1 and t_{10} in the write-up. Be sure to use the same integrator with the same tolerance as in 1a. Compare to the online solutions before proceeding.

Solution

Note: For the solutions in this homework, Φ will be used to represent $\underline{\underline{\Phi}}$ to stay consistent with the convention used in the text, particularly for Problem 1c.

$$\underline{X}^*(t_1) = [-0.839031098038944,\ -0.544071486479261,\ 0.544076120415627,\ -0.839041244186316]^T \tag{3}$$

$$\Phi(t_1, t_0) = \begin{bmatrix} -19.2963174705313 & -1.00059195284582 & -1.54462409476518 & -20.592274677963 \\ 24.5395368984514 & 2.54304003750491 & 3.38202243902567 & 24.9959638292643 \\ -26.628448580308 & -1.24704108018372 & -2.0860289935347 & -27.5413748340174 \\ -15.0754226453647 & -1.45709728481153 & -2.00114420643928 & -14.6674122499641 \end{bmatrix} \tag{4}$$

$$\underline{X}^*(t_{10}) = [0.862623359653583,\ -0.505843963221603,\ 0.50584568923193,\ 0.862623303038346]^T \tag{5}$$

$$\Phi(t_{10}, t_0) = \begin{bmatrix} -151.284032326374 & -0.0696433460453671 & -0.575183991319675 & -152.539455288431 \\ -260.234514431179 & 0.881235606618973 & 0.0191322894654827 & -260.670088444015 \\ 259.154447538201 & 0.374643452779477 & 1.23674843709451 & 260.026380249705 \\ -152.127910765101 & 0.366712857375205 & -0.138829570274937 & -151.639213163376 \end{bmatrix} \tag{6}$$

Problem 1c

For this problem, $\underline{\underline{\Phi}}(t_i, t_0)$ is symplectic. Demonstrate this for $\underline{\underline{\Phi}}(t_{10}, t_0)$ by multiplying it by $\underline{\underline{\Phi}}^{-1}(t_{10}, t_0)$, given by Eq. 4.2.22 in the text. Provide $\underline{\underline{\Phi}}^{-1}(t_{10}, t_0)$ and show that the product with $\underline{\underline{\Phi}}(t_{10}, t_0)$ is the identity matrix.

Solution

From Eq. 4.2.22 in the text[1]:

$$\Phi^{-1}(t, t_k) = \begin{bmatrix} \Phi_4^T & -\Phi_2^T \\ -\Phi_3^T & \Phi_1^T \end{bmatrix} \tag{7}$$

From $\Phi(t_{10}, t_0)$ in Problem 1b, Φ_4^T, Φ_2^T, Φ_3^T and Φ_1^T are as follows:

$$\Phi_1^T = \begin{bmatrix} -151.284032326374 & -0.0696433460453671 \\ -260.234514431179 & 0.881235606618973 \end{bmatrix} \tag{8}$$

$$\Phi_2^T = \begin{bmatrix} -0.575183991319675 & -152.539455288431 \\ 0.0191322894654827 & -260.670088444015 \end{bmatrix} \tag{9}$$

$$\Phi_3^T = \begin{bmatrix} 259.154447538201 & 0.374643452779477 \\ -152.127910765101 & 0.366712857375205 \end{bmatrix} \tag{10}$$

$$\Phi_4^T = \begin{bmatrix} 1.23674843709451 & 260.026380249705 \\ -0.138829570274937 & -151.639213163376 \end{bmatrix} \tag{11}$$

The inverse of $\Phi(t_{10}, t_0)$ evaluates to:

$$\Phi^{-1}(t_{10}, t_0) = \begin{bmatrix} 1.23674843709451 & -0.138829570274937 & 0.575183991319675 & -0.0191322894654827 \\ 260.026380249705 & -151.639213163376 & 152.539455288431 & 260.670088444015 \\ -259.154447538201 & 152.127910765101 & -151.284032326374 & -260.234514431179 \\ -0.374643452779477 & -0.366712857375205 & -0.0696433460453671 & 0.881235606618973 \end{bmatrix} \tag{12}$$

The product of $\Phi^{-1}(t_{10}, t_0)$ and $\Phi(t_{10}, t_0)$ is the identity matrix, which shows that $\Phi(t_{10}, t_0)$ is symplectic. The format is short so that the significant figures may fit on the page:

$$\Phi^{-1}(t_{10}, t_0)\Phi(t_{10}, t_0) = \begin{bmatrix} 1.0000 & -0.0000 & 0.0000 & -0.0000 \\ 0.0000 & 1.0000 & 0.0000 & 0 \\ 0 & -0.0000 & 1.0000 & 0.0000 \\ 0.0000 & 0 & -0.0000 & 1.0000 \end{bmatrix} \tag{13}$$

Problem 1d

Calculate the perturbation vector, $\delta\underline{X}(t_i)$, by the following methods:

$$\begin{aligned}(1)\quad \delta\underline{X}(t_i) &= \underline{X}(t_i) - \underline{X}^*(t_i)\\ (2)\quad \delta\underline{X}(t_i) &= \underline{\underline{\Phi}}(t_i,t_0)\delta\underline{X}(t_0)\end{aligned}$$

and compare the results of (1) and (2). Provide the numeric results of (1) and (2) at t_1 and t_{10} in the write-up, along with $\delta\underline{X}(t_i) - \underline{\underline{\Phi}}(t_i,t_0)\delta\underline{X}(t_0)$. How closely do they compare?

Solution

$\delta\underline{X}(t_1)$ from method 1:

$$\delta\underline{X}(t_1) = \begin{bmatrix} -4.04310376698191e-05 \\ 5.03755900577163e-05 \\ -5.5009526467753e-05 \\ -3.02848902397068e-05 \end{bmatrix} \tag{14}$$

$\delta\underline{X}(t_1)$ from method 2:

$$\delta\underline{X}(t_1) = \begin{bmatrix} -4.04326242904136e-05 \\ 5.03744831292364e-05 \\ -5.50088113276763e-05 \\ -3.02868818169565e-05 \end{bmatrix} \tag{15}$$

The difference between the two methods for $\delta\underline{X}(t_1)$ is the following:

$$\begin{bmatrix} 1.58662059446919e-09 \\ 1.10692847990051e-09 \\ -7.15140076693879e-10 \\ 1.99157724972097e-09 \end{bmatrix} \tag{16}$$

$\delta\underline{X}(t_{10})$ from method 1:

$$\delta\underline{X}(t_{10}) = \begin{bmatrix} -0.000304487377425611 \\ -0.000521677908372098 \\ 0.000519951897818727 \\ -0.000304430762562036 \end{bmatrix} \tag{17}$$

$\delta\underline{X}(t_{10})$ from method 2:

$$\delta\underline{X}(t_{10}) = \begin{bmatrix} -0.000304329028260079 \\ -0.000521766706192347 \\ 0.000520042932772221 \\ -0.000304272666356127 \end{bmatrix} \tag{18}$$

The difference between the two methods for $\delta\underline{X}(t_{10})$ is the following:

$$\begin{bmatrix} -1.58349165531931e-07 \\ 8.8797820249403e-08 \\ -9.10349534938987e-08 \\ -1.5809620590901e-07 \end{bmatrix} \tag{19}$$

t_{10} is 10 TU larger than t_1, but the order of magnitude of the difference between both methods increases by 2 from t_1 to t_{10}. The results from both methods are quite close, but gradually diverge as time goes on.

Problem 2

Given the observation state relation $\underline{y} = \underline{\underline{H}}\ x + \underline{\epsilon}$, where x is a scalar and

$$\underline{y} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$$

$$\underline{\underline{W}} = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

and

$$\underline{\underline{H}} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^T$$

with *a priori* information $\overline{x} = 2$ and $\overline{W} = 2$:

Problem 2a

Using the batch processing algorithm, what is $\hat{x}$? In the write-up, outline the method employed in the code.

Solution

$$\hat{x} = 1.5 \tag{20}$$

The method employed in the code is the following:

1. Given $\underline{y}$, $\underline{\underline{W}}$, and $\underline{\underline{H}}$, and *a priori* information $\bar{x}$ and $\bar{W}$,
2. Initialize $\Lambda = \bar{W}$
3. Initialize $N = \bar{W}\bar{x}$
4. Accumulate $\Lambda = \Lambda + \underline{\underline{H}}^T \underline{\underline{W}}\ \underline{\underline{H}}$
5. Accumulate $N = N + \underline{\underline{H}}\ \underline{\underline{W}}\ \underline{y}$
6. Use the normal equation to find $\hat{x}$ (multiply both sides by inverted Λ): $\Lambda\hat{x} = N$

Problem 2b

What is the best estimate of the observation error, $\hat{\epsilon}$?

Solution

The error can be found from the observation state relation:

$$e = \underline{y} - \underline{\underline{H}}\hat{x} \tag{21}$$

The best estimate of the observation error is:

$$e = [-0.5, 0.5, -0.5]^T \tag{22}$$

Appendix

HW3 MATLAB code

```
% ASE 389 Orbit Determination
% HW 2
% Junette Hsin

clear;

%% Problem 1a:

global mu
mu = 1;

% set ode45 params
rel_tol = 3e-14;          % 1e-14 accurate; 1e-6 coarse
abs_tol = 1e-16;
options = odeset('reltol', rel_tol, 'abstol', abs_tol );

rv0 = [1; 0; 0; 1];
dt = 0.01;

% integrate
[t, rv] = ode45(@fn.TwoBod_4states, [0:dt:100], [rv0], options);

%% Problem 1b:

drv0 = [1e-6; -1e-6; 1e-6; 1e-6];
STM0 = eye(4);
STM0 = reshape(STM0, [16 1]);

rvSTM0 = [rv0 - drv0; STM0];

% integrate
[tstar, rvstar] = ode45(@fn.TwoBod_4states_STM, [0:dt:100], [rvSTM0], options);

STMf = rvstar(end, 5:20);
STMf = reshape(STMf, [4 4]);

%% Problem 1c:

STMf1 = STMf(1:2, 1:2);
STMf2 = STMf(1:2, 3:4);
STMf3 = STMf(3:4, 1:2);
STMf4 = STMf(3:4, 3:4);

STMfinv = [ STMf4',  -STMf2'; ...
-STMf3', STMf1' ];

STMfinv * STMf

%% Problem 1d:

% t1 = 10 TU
i = 10 / dt + 1;
STMi = rvstar(i, 5:20);
STMi = reshape(STMi, [4 4]);

drv1 = rv(i,:) - rvstar(i,1:4);
drv2 = STMi * drv0;

ddrvt1 = drv1' - drv2;

% t10 = 100 TU
i = 100 / dt + 1;
STMi = rvstar(i, 5:20);
STMi = reshape(STMi, [4 4]);
```

```
drv1 = rv(i,:) - rvstar(i,1:4);
drv2 = STMi * drv0;

ddrvt10 = drv1' - drv2;

%% Problem 2: Given the observation state relation y = H x + eps, where x is a scalar and
% y = [1; 2; 1]
% W = [2 0 0; 0 1 0; 0 0 1];
% H = [1; 1; 1];
% with a priori information xbar = 2 and Wbar = 2:
%
% Problem 2a: Using the batch processing algorithm, what is x^? In the write-up, outline the
    method
% employed in the code.

% W matrix is inv(R)
% Wbar = inv(P)

% observation states
y = [1; 2; 1];
W = [2 0 0; 0 1 0; 0 0 1];
H = [1; 1; 1];

x0 = 2;
W0 = 2;

% Lambda = inv(P) = W0
P0       = inv(W0);
Lambda   = W0;
R        = inv(W);

% N = inv(P) * x0 = W0 * x0
N = W0 * x0;

% accumulate
Lambda   = Lambda + H' * W * H;
N        = N + H' * W * y;

% normal equation
xhat = inv(Lambda) * N;

e = y - H*xhat;

%% Problem 2b: What is the best estimate of the observation error, eps?

e = y - H*xhat;
```

Two Body EOM

```
function dx = TwoBod_4states(t, x)
% ----------------------------------------------------------------------
% Inputs
%   t = [Nx1] time vector (orbit is Keplerian, doesn't matter)
%   x = [4x1] state vector
%
% Outputs
%   dx = [4x1] derivative of state vector
% ----------------------------------------------------------------------

global mu

dx = zeros(4, 1);   % force column vector

% dx1 = x3
% dx2 = x4
% dx3 = (-u/r^3) * x1
```

```
% dx4 = (-u/r^3) * x2

dx(1:2) = x(3:4);
r_norm  = norm(x(1:2));
dx(3:4) = ( - mu / r_norm^3 ) * x(1:2);

end
```

Two Body EOM with STM

```
function drvSTM = TwoBod_4states_STM(t, rvSTM)
% ----------------------------------------------------------------------
% Inputs
%   t = [Nx1] time vector (orbit is Keplerian, doesn't matter)
%   x = [4x1] state vector
%
% Outputs
%   dx = [4x1] derivative of state vector
% ----------------------------------------------------------------------

global mu

% initialize
drv     = zeros(4, 1);        % force column vector
drvSTM  = zeros(20, 1);    % STM is 4-by-4 --> 16

STM     = rvSTM(5:20);
STM     = reshape(STM, [4 4]);

x  = rvSTM(1);
y  = rvSTM(2);
dx = rvSTM(3);
dy = rvSTM(4);

% dx1 = x3
% dx2 = x4
% dx3 = (-u/r^3) * x1
% dx4 = (-u/r^3) * x2

drv(1:2) = [dx; dy];
r        = norm([x, y]);
drv(3:4) = ( - mu / r^3 ) * [x; y];

%% STM stuff

G = [ -mu/r^3 + 3*mu*x^2/r^5 , 3*mu*x*y/r^5             ; ...
3*mu*x*y/r^5              , -mu/r^3 + 3*mu*y^2/r^5   ];

K = zeros(2,2);

A = [ zeros(2), eye(2); ...
G,          K ];

dSTM = A * STM;
dSTM = reshape(dSTM, [16 1]);

drvSTM = [drv; dSTM];

end
```

References

[1] Bob Schutz, G. H. B., Byron Tapley, *Statistical Orbit Determination*, Academic Press, 2004.