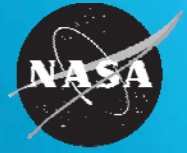


National Aeronautics and Space Administration



Numerical Computation of a Continuous-Thrust State Transition Matrix Incorporating Accurate Hardware and Ephemeris Models

Donald Ellison
Bruce Conway
Jacob Englander



NAVIGATION & MISSION DESIGN BRANCH
NASA GSFC

www.nasa.gov

code 595



Authors



- Donald Ellison – Ph. D. Student, Department of Aerospace Engineering
University of Illinois at Urbana-Champaign
- Bruce Conway – Professor, Department of Aerospace Engineering
University of Illinois at Urbana-Champaign
- Jacob Englander – Aerospace Engineer, Navigation & Mission Design
Branch, Code 595, NASA Goddard Space Flight Center

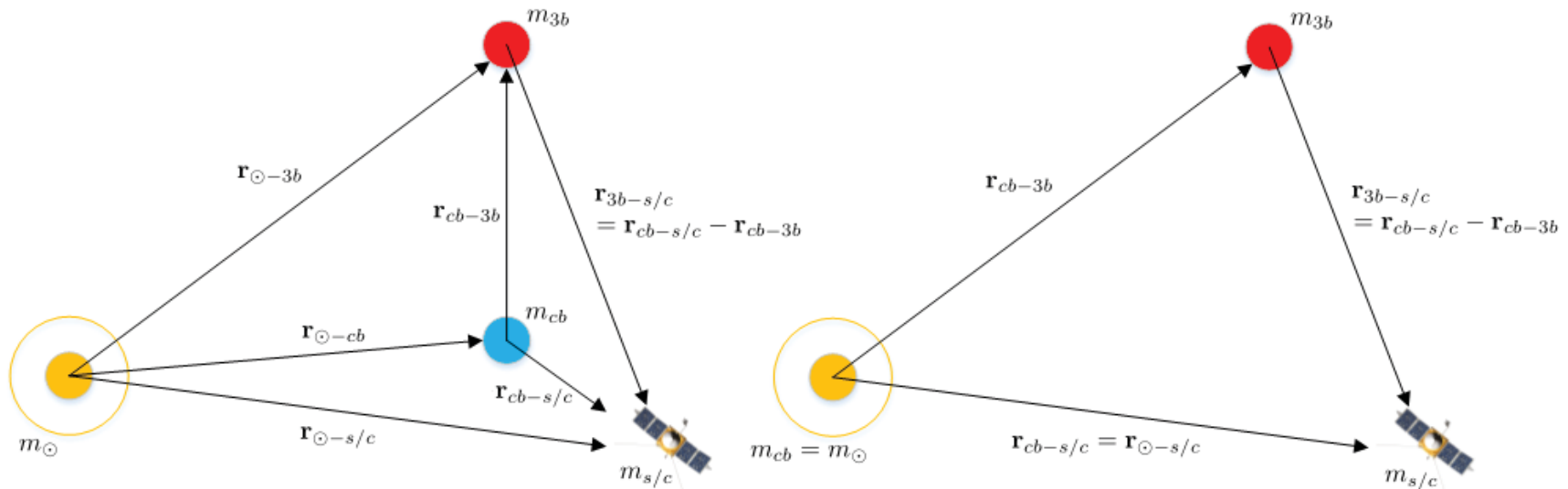
Outline

- Evolutionary Mission Trajectory Generator (EMTG)
- Finite-burn low-thrust (FBLT) transcription
- FBLT vs. multiple gravity-assist with low-thrust (MGALT) transcription
- FBLT match point constraint gradient calculation
- Continuous-thrust state transition matrix calculation
- Adaptive-step RK7(8)13M integrator
- FBLT match point time of flight gradient calculation
- Numerical Example
- Summary

The Evolutionary Mission Trajectory Generator (EMTG)

- Automated interplanetary low-thrust mission design tool
- Capabilities:
 - Global optimization of both the flyby sequence and the trajectory
 - No initial guess required, global search capability provided by monotonic basin hopping (MBH) algorithm
 - Works in multiple reference frames (interplanetary, moon tours, Earth orbit, etc)
 - Integration with SPICE ephemerides
 - Up-to-date models of launch vehicles, thrusters, power systems
 - Easy to use graphical user interface
- The purpose of EMTG is to automate almost all of the work so that an analyst can investigate a wide range of mission options in very little human time
- Computer time is CHEAP. Analyst time is EXPENSIVE

Finite-Burn Low-Thrust Transcription



$$\ddot{\mathbf{r}}_{cb-s/c} = -\frac{G(m_{cb} + m_{s/c}^0)}{r_{cb-s/c}^3} \mathbf{r}_{cb-s/c} - \sum_i G m_i \left[\frac{\mathbf{r}_{cb-s/c} - \mathbf{r}_{cb-3b}}{\|\mathbf{r}_{cb-s/c} - \mathbf{r}_{cb-3b}\|^3} \right]_i + \frac{\mathbf{u} D T}{m_{s/c}}$$

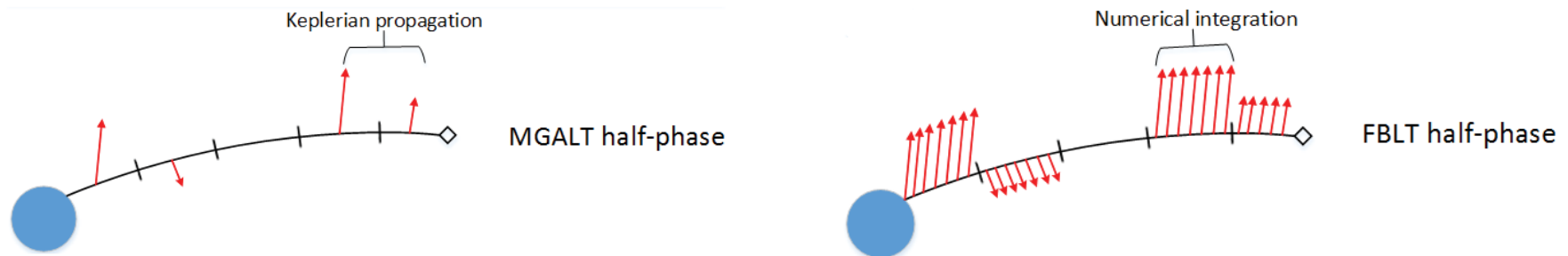
$$\dot{m} = -\|\mathbf{u}\| D \dot{m}_{max}$$

J. A. Englander, D. H. Ellison, and B. A. Conway, "Global Optimization of Low-Thrust, Multiple-Flyby Trajectories at Medium and Medium-High Fidelity," *AAS/AIAA Space Flight Mechanics Meeting, Santa Fe, NM*, January 26 - 30 2014.

FBLT vs. MGALT

\mathbf{p}	Description	Notes
t_0	Launch epoch	
v_∞	Magnitude of the outgoing velocity asymptote	first phase only
RA	Right ascension of launch asymptote	first phase only
DEC	Declination of launch asymptote	first phase only
TOF	Phase time of flight	N_p
m_f	Phase final mass	N_p
\mathbf{u}	Segment control vector	one per FBLT segment ($3N \times N_p$)
I_{sp}	Propulsion system specific impulse	only for VSI-capable engines ($N \times N_p$)
\mathbf{v}_{∞_0}	Phase initial excess velocity vector	all but the first phase ($3N_p - 1$)
\mathbf{v}_{∞_f}	Phase final excess velocity vector	all phases, except the final phase of a rendezvous (N_p)

Table 1: Typical decision vector for a low-thrust mission FBLT mission.



FBLT as a Nonlinear Program: Constraints

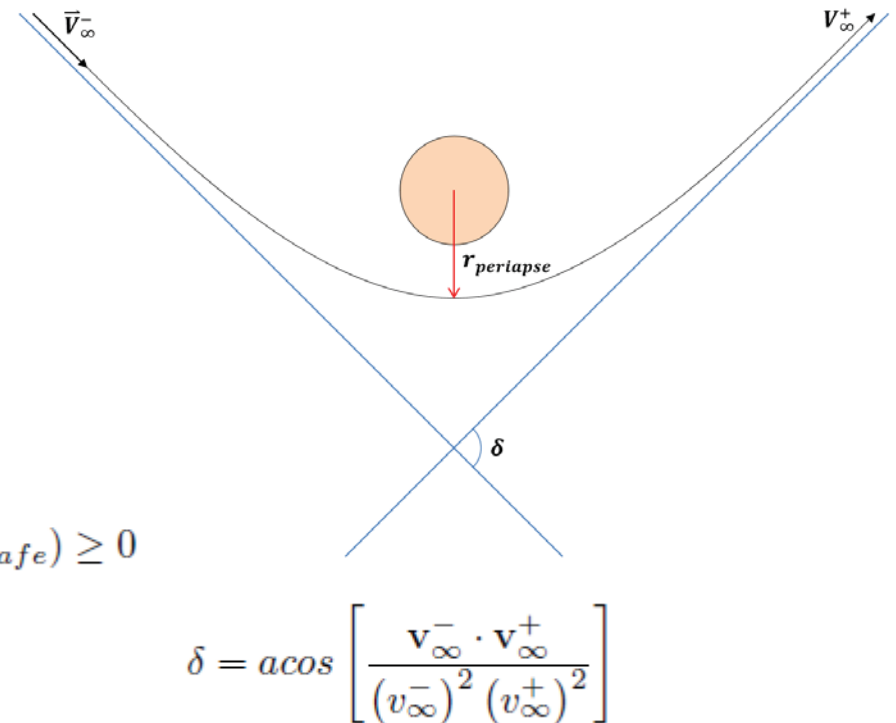
- MGALT and FBLT share identical decision and constraint vectors
- Control bounds, flyby model and TOF constraint gradients may be provided fully analytically

$$\|\mathbf{u}\| = \sqrt{u_{x_k}^2 + u_{y_k}^2 + u_{z_k}^2} \leq 1$$

$$c_{v_\infty} = v_\infty^+ - v_\infty^- = 0$$

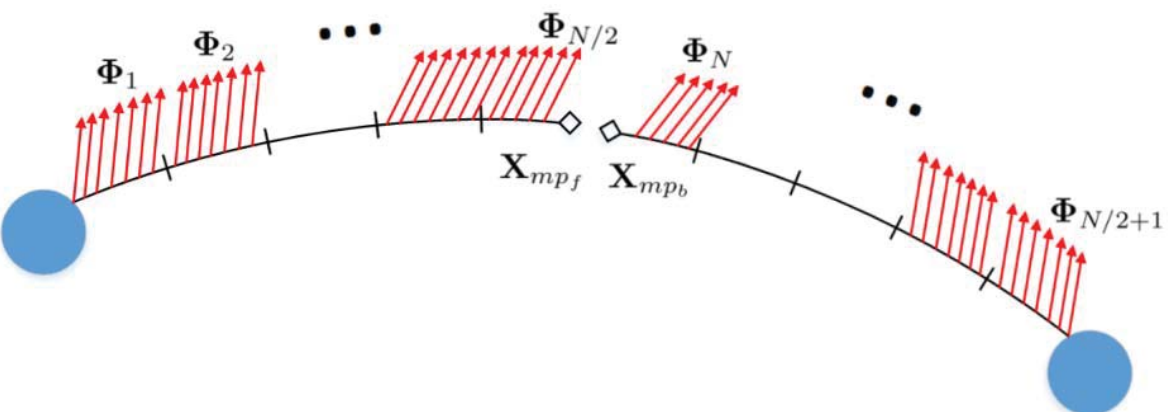
$$\begin{aligned} c_{\text{flyby-altitude}} &= r_{\text{periapse}} - (r_{\text{planet}} + h_{\text{safe}}) \geq 0 \\ &= \frac{\mu_{\text{planet}}}{v_\infty^2} \left[\frac{1}{\sin(\delta/2)} - 1 \right] - (r_{\text{planet}} + h_{\text{safe}}) \geq 0 \end{aligned}$$

$$c_{\text{TOF}} = t_{\min} \leq t_{\text{flight}} = \sum_{i=1}^{N_p} t_{\text{phase}_i} \leq t_{\max}$$



FBLT as a Nonlinear Program: Constraints

- FBLT transcription results in a large, sparse NLP
- We want to calculate the gradient of each NLP constraint w.r.t. each NLP parameter (KKT first order necessary conditions) → form the problem Jacobian
- SNOPT will do this with finite differencing...this is EXPENSIVE
- FBLT is already much slower than MGALT due to numerical integration vs. analytic Kepler, so it is even more important to supply user-defined derivatives
- The most challenging are the match point continuity constraints: state transition matrices

$$\mathbf{c}_{mp} = \mathbf{X}_{mp_b} - \mathbf{X}_{mp_f}$$


The diagram illustrates a curved trajectory between two blue circular endpoints. The trajectory is divided into segments by match points, marked with small diamonds. Red arrows represent state transition matrices $\Phi_1, \Phi_2, \dots, \Phi_{N/2}, \Phi_N, \dots, \Phi_{N/2+1}$ connecting the match points. The match points are labeled \mathbf{X}_{mp_f} and \mathbf{X}_{mp_b} .

$$\frac{\partial \mathbf{c}_{mp}}{\partial \mathbf{p}} = \frac{\partial \mathbf{X}_{mp_b}}{\partial \mathbf{p}} - \frac{\partial \mathbf{X}_{mp_f}}{\partial \mathbf{p}}$$

Match Point Constraint Gradient Calculation

- Traditional six-by-six STM is augmented to include mass and the current time step's control parameters u_x, u_y and u_z
- Match point constraint vector sensitivities w.r.t. controls make up the majority of the dense entries in the Jacobian
- An STM chain is constructed beginning with the segment of interest and proceeding using "stripped" STMs to the match point

$$\Phi(t, t_0) = \begin{bmatrix} \frac{\partial \mathbf{r}}{\partial \mathbf{r}_0} & \frac{\partial \mathbf{r}}{\partial \dot{\mathbf{r}}_0} & \frac{\partial \mathbf{r}}{\partial m_0} & \frac{\partial \mathbf{r}}{\partial \mathbf{u}_0} \\ \frac{\partial \dot{\mathbf{r}}}{\partial \mathbf{r}_0} & \frac{\partial \dot{\mathbf{r}}}{\partial \dot{\mathbf{r}}_0} & \frac{\partial \dot{\mathbf{r}}}{\partial m_0} & \frac{\partial \dot{\mathbf{r}}}{\partial \mathbf{u}_0} \\ \frac{\partial m}{\partial \mathbf{r}_0} & \frac{\partial m}{\partial \dot{\mathbf{r}}_0} & \frac{\partial m}{\partial m_0} & \frac{\partial m}{\partial \mathbf{u}_0} \\ \frac{\partial \mathbf{u}}{\partial \mathbf{r}_0} & \frac{\partial \mathbf{u}}{\partial \dot{\mathbf{r}}_0} & \frac{\partial \mathbf{u}}{\partial m_0} & \frac{\partial \mathbf{u}}{\partial \mathbf{u}_0} \end{bmatrix} \quad \Phi_s(t, t_0) = \begin{bmatrix} \frac{\partial \mathbf{r}}{\partial \mathbf{r}_0} & \frac{\partial \mathbf{r}}{\partial \dot{\mathbf{r}}_0} & \frac{\partial \mathbf{r}}{\partial m_0} & \mathbf{0}_{3 \times 3} \\ \frac{\partial \dot{\mathbf{r}}}{\partial \mathbf{r}_0} & \frac{\partial \dot{\mathbf{r}}}{\partial \dot{\mathbf{r}}_0} & \frac{\partial \dot{\mathbf{r}}}{\partial m_0} & \mathbf{0}_{3 \times 3} \\ \frac{\partial m}{\partial \mathbf{r}_0} & \frac{\partial m}{\partial \dot{\mathbf{r}}_0} & \frac{\partial m}{\partial m_0} & \mathbf{0}_{1 \times 3} \\ \frac{\partial \mathbf{u}}{\partial \mathbf{r}_0} & \frac{\partial \mathbf{u}}{\partial \dot{\mathbf{r}}_0} & \frac{\partial \mathbf{u}}{\partial m_0} & \mathbf{0}_{3 \times 3} \end{bmatrix}$$

$$\Phi(t_{mp}, t_i) = \Phi_s(t_{mp}, t_{mp-1}) \cdot \dots \cdot \Phi_s(t_{i+2}, t_{i+1}) \cdot \Phi(t_{i+1}, t_i)$$

FBLT State Transition Matrix Calculation

- Unlike the two-body problem (see Battin for example), analytic expressions for the perturbation STM entries don't exist for true continuous thrust
- We can solve for them numerically
- Differential equations for the STM entries are appended to the physics EOM's and integrated alongside them
- We must calculate the state propagation matrix (**A**) at each integration time step
- **A** also referred to as the fundamental solution/set

$$\mathbf{X} = \begin{bmatrix} \mathbf{r} \\ \dot{\mathbf{r}} \\ m \end{bmatrix} = \left[x \quad y \quad z \quad v_x \quad v_y \quad v_z \quad m \right]^T$$

$$\begin{aligned} \dot{\Phi} &= \mathbf{A}\Phi \\ \mathbf{A} &= \frac{\partial \dot{\mathbf{X}}}{\partial \mathbf{X}} \end{aligned}$$

$$\dot{\mathbf{X}} = \mathbf{f} = \left[\dot{\mathbf{r}} \quad \ddot{\mathbf{r}} \quad \dot{m} \quad \dot{u}_x \quad \dot{u}_y \quad \dot{u}_z \quad \dot{s}_{11} \quad \dot{s}_{12} \quad \dots \quad \dot{s}_{1010} \right]^T$$

S. P. Hughes, D. S. Cooley, and J. J. Guzman, "A Direct Method for Fuel Optimal Maneuvers of Distributed Spacecraft in Multiple Flight Regimes," *Proceedings of the AAS/AIAA Space Flight Mechanics Meeting, Copper Mountain, Colorado*, No. AAS-05-158, January 23-27 2005.

S. P. Hughes and E. Dove, "Optimal Control and Near Optimal Guidance for the Magnetospheric Multi-Scale Mission (MMS)," *Proceedings of the AAS/AIAA Astrodynamics Specialist Conference, Pittsburgh, Pennsylvania*, No. AAS-09-330, August 9-13 2009.

FBLT State Transition Matrix Calculation

$$\dot{\Phi} = \mathbf{A}\Phi \quad \mathbf{A} = \frac{\partial \dot{\mathbf{X}}}{\partial \mathbf{X}}$$

$$\Phi = \begin{bmatrix} s_{11} & s_{12} & s_{13} & s_{14} & s_{15} & s_{16} & s_{17} & s_{18} & s_{19} & s_{110} \\ s_{21} & s_{22} & s_{23} & s_{24} & s_{25} & s_{26} & s_{27} & s_{28} & s_{29} & s_{210} \\ s_{31} & s_{32} & s_{33} & s_{34} & s_{35} & s_{36} & s_{37} & s_{38} & s_{39} & s_{310} \\ s_{41} & s_{42} & s_{43} & s_{44} & s_{45} & s_{46} & s_{47} & s_{48} & s_{49} & s_{410} \\ s_{51} & s_{52} & s_{53} & s_{54} & s_{55} & s_{56} & s_{57} & s_{58} & s_{59} & s_{510} \\ s_{61} & s_{62} & s_{63} & s_{64} & s_{65} & s_{66} & s_{67} & s_{68} & s_{69} & s_{610} \\ s_{71} & s_{72} & s_{73} & s_{74} & s_{75} & s_{76} & s_{77} & s_{78} & s_{79} & s_{710} \\ s_{81} & s_{82} & s_{83} & s_{84} & s_{85} & s_{86} & s_{87} & s_{88} & s_{89} & s_{810} \\ s_{91} & s_{92} & s_{93} & s_{94} & s_{95} & s_{96} & s_{97} & s_{98} & s_{99} & s_{910} \\ s_{101} & s_{102} & s_{103} & s_{104} & s_{105} & s_{106} & s_{107} & s_{108} & s_{109} & s_{1010} \end{bmatrix}$$

FBLT State Transition Matrix Calculation

$$\dot{\Phi} = \mathbf{A}\Phi \quad \mathbf{A} = \frac{\partial \dot{\mathbf{X}}}{\partial \mathbf{X}}$$

$$\mathbf{A} = \begin{bmatrix} \frac{\partial v_x}{\partial x} & \frac{\partial v_x}{\partial y} & \frac{\partial v_x}{\partial z} & \frac{\partial v_x}{\partial v_x} & \frac{\partial v_x}{\partial v_y} & \frac{\partial v_x}{\partial v_z} & \frac{\partial v_x}{\partial m} & \frac{\partial v_x}{\partial u_x} & \frac{\partial v_x}{\partial u_y} & \frac{\partial v_x}{\partial u_z} \\ \frac{\partial v_y}{\partial x} & \frac{\partial v_y}{\partial y} & \frac{\partial v_y}{\partial z} & \frac{\partial v_y}{\partial v_x} & \frac{\partial v_y}{\partial v_y} & \frac{\partial v_y}{\partial v_z} & \frac{\partial v_y}{\partial m} & \frac{\partial v_y}{\partial u_x} & \frac{\partial v_y}{\partial u_y} & \frac{\partial v_y}{\partial u_z} \\ \frac{\partial v_z}{\partial x} & \frac{\partial v_z}{\partial y} & \frac{\partial v_z}{\partial z} & \frac{\partial v_z}{\partial v_x} & \frac{\partial v_z}{\partial v_y} & \frac{\partial v_z}{\partial v_z} & \frac{\partial v_z}{\partial m} & \frac{\partial v_z}{\partial u_x} & \frac{\partial v_z}{\partial u_y} & \frac{\partial v_z}{\partial u_z} \\ \frac{\partial a_x}{\partial x} & \frac{\partial a_x}{\partial y} & \frac{\partial a_x}{\partial z} & \frac{\partial a_x}{\partial v_x} & \frac{\partial a_x}{\partial v_y} & \frac{\partial a_x}{\partial v_z} & \frac{\partial a_x}{\partial m} & \frac{\partial a_x}{\partial u_x} & \frac{\partial a_x}{\partial u_y} & \frac{\partial a_x}{\partial u_z} \\ \frac{\partial a_y}{\partial x} & \frac{\partial a_y}{\partial y} & \frac{\partial a_y}{\partial z} & \frac{\partial a_y}{\partial v_x} & \frac{\partial a_y}{\partial v_y} & \frac{\partial a_y}{\partial v_z} & \frac{\partial a_y}{\partial m} & \frac{\partial a_y}{\partial u_x} & \frac{\partial a_y}{\partial u_y} & \frac{\partial a_y}{\partial u_z} \\ \frac{\partial a_z}{\partial x} & \frac{\partial a_z}{\partial y} & \frac{\partial a_z}{\partial z} & \frac{\partial a_z}{\partial v_x} & \frac{\partial a_z}{\partial v_y} & \frac{\partial a_z}{\partial v_z} & \frac{\partial a_z}{\partial m} & \frac{\partial a_z}{\partial u_x} & \frac{\partial a_z}{\partial u_y} & \frac{\partial a_z}{\partial u_z} \\ \frac{\partial \dot{m}}{\partial x} & \frac{\partial \dot{m}}{\partial y} & \frac{\partial \dot{m}}{\partial z} & \frac{\partial \dot{m}}{\partial v_x} & \frac{\partial \dot{m}}{\partial v_y} & \frac{\partial \dot{m}}{\partial v_z} & \frac{\partial \dot{m}}{\partial m} & \frac{\partial \dot{m}}{\partial u_x} & \frac{\partial \dot{m}}{\partial u_y} & \frac{\partial \dot{m}}{\partial u_z} \\ \frac{\partial \dot{u}_x}{\partial x} & \frac{\partial \dot{u}_x}{\partial y} & \frac{\partial \dot{u}_x}{\partial z} & \frac{\partial \dot{u}_x}{\partial v_x} & \frac{\partial \dot{u}_x}{\partial v_y} & \frac{\partial \dot{u}_x}{\partial v_z} & \frac{\partial \dot{u}_x}{\partial m} & \frac{\partial \dot{u}_x}{\partial u_x} & \frac{\partial \dot{u}_x}{\partial u_y} & \frac{\partial \dot{u}_x}{\partial u_z} \\ \frac{\partial \dot{u}_y}{\partial x} & \frac{\partial \dot{u}_y}{\partial y} & \frac{\partial \dot{u}_y}{\partial z} & \frac{\partial \dot{u}_y}{\partial v_x} & \frac{\partial \dot{u}_y}{\partial v_y} & \frac{\partial \dot{u}_y}{\partial v_z} & \frac{\partial \dot{u}_y}{\partial m} & \frac{\partial \dot{u}_y}{\partial u_x} & \frac{\partial \dot{u}_y}{\partial u_y} & \frac{\partial \dot{u}_y}{\partial u_z} \\ \frac{\partial \dot{u}_z}{\partial x} & \frac{\partial \dot{u}_z}{\partial y} & \frac{\partial \dot{u}_z}{\partial z} & \frac{\partial \dot{u}_z}{\partial v_x} & \frac{\partial \dot{u}_z}{\partial v_y} & \frac{\partial \dot{u}_z}{\partial v_z} & \frac{\partial \dot{u}_z}{\partial m} & \frac{\partial \dot{u}_z}{\partial u_x} & \frac{\partial \dot{u}_z}{\partial u_y} & \frac{\partial \dot{u}_z}{\partial u_z} \end{bmatrix}$$

Calculation of the State Propagation Matrix

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbb{I}_{3 \times 3} & \mathbf{0} & \mathbf{0} \\ \mathbf{A}_{21} & \mathbf{0} & \mathbf{A}_{23} & \mathbf{A}_{24} \\ \mathbf{A}_{31} & \mathbf{0} & \mathbf{0} & \mathbf{A}_{34} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}$$

$$\begin{aligned} \mathbf{A}_{21} &= \frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial \mathbf{r}_{cb-s/c}} \\ &= \frac{3\mu_{cb} \mathbf{r}_{cb-s/c} \mathbf{r}_{cb-s/c}^T}{r_{cb-s/c}^5} - \frac{\mu_{cb}}{r_{cb-s/c}^3} \mathbb{I}_{3 \times 3} + \frac{3\mu_{3b} \mathbf{r}_{3b-s/c} \mathbf{r}_{3b-s/c}^T}{r_{3b-s/c}^5} - \frac{\mu_{3b}}{r_{3b-s/c}^3} \mathbb{I}_{3 \times 3} \\ &\quad + \frac{\mathbf{u} D}{m_{s/c}} \cdot \frac{\partial T}{\partial P} \cdot \frac{\partial P}{\partial r_{\odot-s/c}} \cdot \frac{\partial r_{\odot-s/c}}{\partial \mathbf{r}_{cb-s/c}} \end{aligned}$$

Calculation of the State Propagation Matrix

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbb{I}_{3 \times 3} & \mathbf{0} & \mathbf{0} \\ \mathbf{A}_{21} & \mathbf{0} & \mathbf{A}_{23} & \mathbf{A}_{24} \\ \mathbf{A}_{31} & \mathbf{0} & \mathbf{0} & \mathbf{A}_{34} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}$$

$$\begin{aligned} \mathbf{A}_{21} &= \frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial \mathbf{r}_{cb-s/c}} \\ &= \frac{3\mu_{cb} \mathbf{r}_{cb-s/c} \mathbf{r}_{cb-s/c}^T}{r_{cb-s/c}^5} - \frac{\mu_{cb}}{r_{cb-s/c}^3} \mathbb{I}_{3 \times 3} + \frac{3\mu_{3b} \mathbf{r}_{3b-s/c} \mathbf{r}_{3b-s/c}^T}{r_{3b-s/c}^5} - \frac{\mu_{3b}}{r_{3b-s/c}^3} \mathbb{I}_{3 \times 3} \\ &\quad + \frac{\mathbf{u} D}{m_{s/c}} \cdot \frac{\partial T}{\partial P} \cdot \frac{\partial P}{\partial r_{\odot-s/c}} \cdot \frac{\partial r_{\odot-s/c}}{\partial \mathbf{r}_{cb-s/c}} \end{aligned}$$

Launch Vehicle, Thruster, and Power Modeling

- Launch vehicles are modeled using a polynomial fit

$$m_{delivered} = (1 - \sigma_{LV}) (f_{LV} C_3^5 + e_{LV} C_3^4 + d_{LV} C_3^3 + c_{LV} C_3^2 + b_{LV} C_3 + a_{LV})$$

where σ_{LV} is launch vehicle margin and C_3 is hyperbolic excess velocity

- Thrusters are modeled using either a polynomial fit to published thrust and mass flow rate data

$$\begin{aligned} \dot{m}_{max} &= e_F P^4 + d_F P^3 + c_F P^2 + b_F P + a_F \\ T_{max} &= e_T P^4 + d_T P^3 + c_T P^2 + b_T P + a_T \end{aligned}$$

or, when detailed performance data is unavailable

$$T_{max} = \frac{2 \eta P}{I_{sp} g_0} \qquad \dot{m}_{max} = \frac{T_{max}}{I_{sp} g_0}$$

- Power is modeled by a standard polynomial model

$$\frac{P_0}{r^2} \left(\frac{\gamma_0 + \frac{\gamma_1}{r} + \frac{\gamma_2}{r^2}}{1 + \gamma_3 r + \gamma_4 r^2} \right) (1 - \tau)^t$$

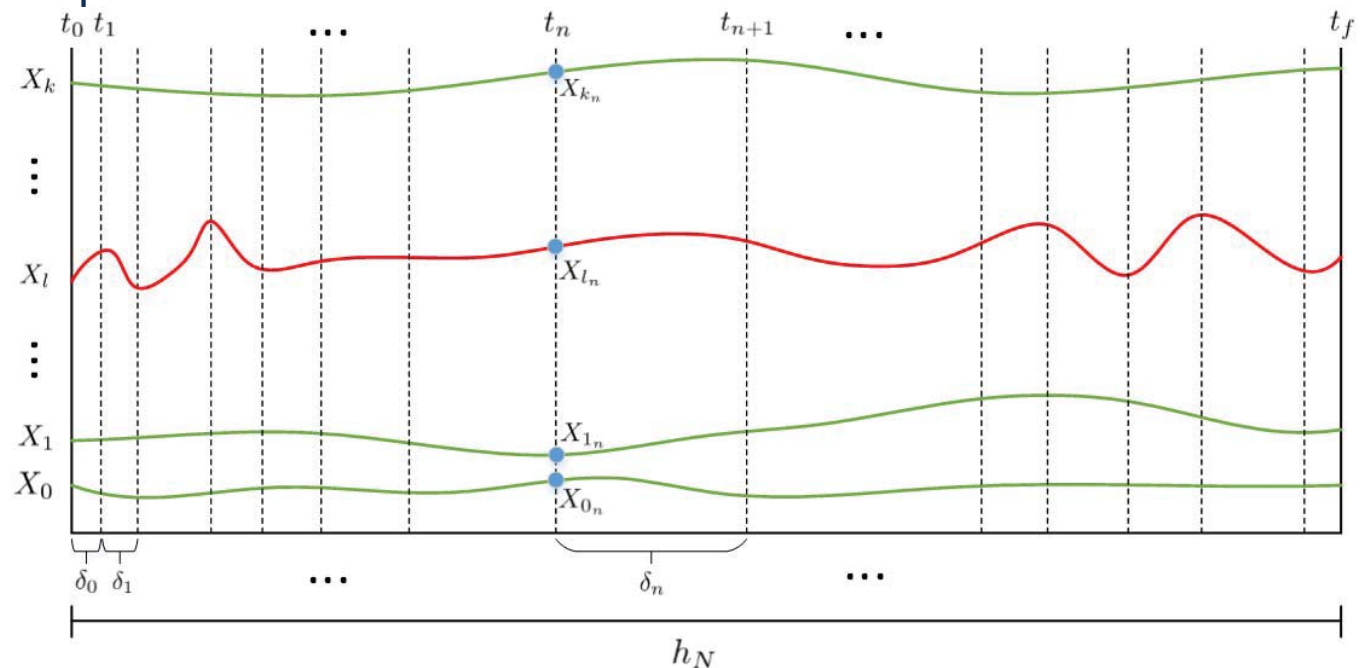
P_0 is the power at beginning of life at 1 AU, τ is the solar array degradation constant and t is the time since launch in years

EMTG's Integrator: Adaptive RK7(8)13M

- Two main components
 - Adaptive step sizing algorithm
 - 8th order embedded explicit Runge-Kutta step calculation method
- Local truncation error between 7th and 8th order solutions is used to adaptively size the integration step

- Each FBLT segment h_N is broken into δ_i sub-steps

$$h_N = \sum_i \delta_i$$



Adaptive Step Size Computation

- After each sub-step, the local truncation error between the 7th and 8th order solutions is calculated (Δe)
- If $\Delta e > \Delta T$ then $S = 0.98$, the time step will be reduced and the sub-step is reintegrated
- else $S = 1.01$, and the time step length is increased, potentially saving computation time
- q is the order of the relative error of the method, $q = 8$ for RK7(8)13M
- The integration proceeds until integration across the full FBLT time step has been completed to within the user-specified error tolerance

$$\delta_{n+1} = \delta_n \cdot S \cdot \left(\frac{\Delta T}{\Delta e} \right)^{\frac{1}{q}}$$

A. M. Ghosh, *Multi-Cubesat Mission Planning Enabled Through Parallel Computing*. PhD thesis, University of Illinois at Urbana-Champaign, May 2013.

EMTG's Integrator: Adaptive RK8(7)13M

1. Calculate the gradient and step at the left hand side of the RK sub-step

$$\mathbf{k}_1 = \delta_n \cdot \mathbf{f} \left(t_n, \hat{\mathbf{X}}_n \right)$$

2. At each of the 13 stages, calculate the state sample point and the gradient/step at that point...store the gradient information

$$\hat{\mathbf{X}}_{n(i)} = \hat{\mathbf{X}}_n + \sum_{j=1}^{i-1} a_{ij} \mathbf{k}_j \quad i > j \quad i, j = 1, 2, 3, \dots, s$$
$$\mathbf{k}_i = \delta_n \cdot \mathbf{f} \left(t_n + c_i \delta_n, \hat{\mathbf{X}}_{n(i)} \right)$$

3. At the right hand side of the sub-step, compute the 7th and 8th order solutions

$$\hat{\mathbf{X}}_{n+1} = \hat{\mathbf{X}}_n + \sum_{i=1}^s \hat{b}_i \mathbf{k}_i \quad \mathbf{X}_{n+1} = \hat{\mathbf{X}}_n + \sum_{i=1}^s b_i \mathbf{k}_i \quad i = 2, 3, \dots, s$$

4. Update time and adjust the step size $t_{n+1} = t_n + \delta_n$

Dormand-Prince RK7(8)13M Butcher Tableau

c_i	a_{ij}										\hat{b}_i	b_i	
0											$\frac{14005451}{335480064}$	$\frac{13451932}{455176623}$	
$\frac{1}{18}$	$\frac{1}{18}$										0	0	
$\frac{1}{12}$	$\frac{1}{48}$	$\frac{1}{16}$									0	0	
$\frac{1}{8}$	$\frac{1}{32}$	0	$\frac{3}{32}$								0	0	
$\frac{5}{16}$	$\frac{5}{16}$	0	$\frac{-75}{64}$	$\frac{75}{64}$							0	0	
$\frac{3}{8}$	$\frac{3}{80}$	0	0	$\frac{3}{16}$	$\frac{3}{20}$						$\frac{-59238493}{1068277825}$	$\frac{-808719846}{976000145}$	
$\frac{59}{400}$	$\frac{29443841}{614563906}$	0	0	$\frac{77736538}{692538347}$	$\frac{-28693883}{1125000000}$	$\frac{23124283}{1800000000}$					$\frac{181606767}{758867731}$	$\frac{1757004468}{5645159321}$	
$\frac{93}{200}$	$\frac{16016141}{946692911}$	0	0	$\frac{61564180}{158732637}$	$\frac{22789713}{633445777}$	$\frac{545815736}{2771057229}$	$\frac{-180193667}{1043307555}$				$\frac{561292985}{797845732}$	$\frac{656045339}{265891186}$	
$\frac{5490023248}{9719169821}$	$\frac{39632708}{573591083}$	0	0	$\frac{-433636366}{683701615}$	$\frac{-421739975}{2616292301}$	$\frac{100302831}{723423059}$	$\frac{790204164}{839813087}$	$\frac{800635310}{3783071287}$			$\frac{-1041891430}{1371343529}$	$\frac{-3867574721}{1518517206}$	
$\frac{13}{20}$	$\frac{246121993}{1340847787}$	0	0	$\frac{-37695042795}{15268766246}$	$\frac{-309121744}{1061227803}$	$\frac{-12992083}{490766935}$	$\frac{6005943493}{2108947869}$	$\frac{393006217}{1396673457}$	$\frac{123872331}{1001029789}$		$\frac{760417239}{1151165299}$	$\frac{465885868}{322736535}$	
$\frac{1201146811}{1299019798}$	$\frac{-1028468189}{846180014}$	0	0	$\frac{8478235783}{508512852}$	$\frac{1311729495}{1432422823}$	$\frac{-10304129995}{1701304382}$	$\frac{-48777925059}{3047939560}$	$\frac{15336726248}{1032824649}$	$\frac{-45442868181}{3398467696}$	$\frac{3065993473}{597172653}$	$\frac{118820643}{751138087}$	$\frac{53011238}{667516719}$	
1	$\frac{185892177}{718116043}$	0	0	$\frac{-3185094517}{667107341}$	$\frac{-477755414}{1098053517}$	$\frac{-703635378}{230739211}$	$\frac{5731566787}{1027545527}$	$\frac{5232866602}{850066563}$	$\frac{-4093664535}{808688257}$	$\frac{3962137247}{1805957418}$	$\frac{65686358}{487910083}$	$\frac{-528747749}{2220607170}$	$\frac{2}{45}$
1	$\frac{403863854}{491063109}$	0	0	$\frac{-5068492393}{434740067}$	$\frac{-411421997}{543043805}$	$\frac{652783627}{914296604}$	$\frac{11173962825}{925320556}$	$\frac{-13158990841}{6184727034}$	$\frac{3936647629}{1978049680}$	$\frac{-160528059}{685178525}$	$\frac{248638103}{1413531060}$	0	$\frac{1}{4}$

P. J. Prince and J. R. Dormand, "High order embedded Runge-Kutta formulae," *Journal of Computational and Applied Mathematics*, Vol. 7, No. 1, 1981, pp. 67-75.

Dormand-Prince RK7(8)13M Butcher Tableau

8th order

c_i	a_{ij}										\hat{b}_i	b_i	
0											<u>14005451</u>	<u>13451932</u>	
											<u>335480064</u>	<u>455176623</u>	
$\frac{1}{18}$	$\frac{1}{18}$										0	0	
$\frac{1}{12}$	$\frac{1}{48}$	$\frac{1}{16}$									0	0	
$\frac{1}{8}$	$\frac{1}{32}$	0	$\frac{3}{32}$								0	0	
$\frac{5}{16}$	$\frac{5}{16}$	0	$-\frac{75}{64}$	$\frac{75}{64}$							0	0	
$\frac{3}{8}$	$\frac{3}{80}$	0	0	$\frac{3}{16}$	$\frac{3}{20}$						<u>-59238493</u>	<u>-808719846</u>	
$\frac{59}{400}$	<u>$\frac{29443841}{614563906}$</u>	0	0	<u>$\frac{77736538}{692538347}$</u>	<u>$\frac{-28693883}{1125000000}$</u>	<u>$\frac{23124283}{1800000000}$</u>					<u>1068277825</u>	<u>976000145</u>	
$\frac{93}{200}$	<u>$\frac{16016141}{946692911}$</u>	0	0	<u>$\frac{61564180}{158732637}$</u>	<u>$\frac{22789713}{633445777}$</u>	<u>$\frac{545815736}{2771057229}$</u>	<u>$\frac{-180193667}{1043307555}$</u>				<u>181606767</u>	<u>1757004468</u>	
$\frac{5490023248}{9719169821}$	<u>$\frac{39632708}{573591083}$</u>	0	0	<u>$\frac{-433636366}{683701615}$</u>	<u>$\frac{-421739975}{2616292301}$</u>	<u>$\frac{100302831}{723423059}$</u>	<u>$\frac{790204164}{839813087}$</u>	<u>$\frac{800635310}{3783071287}$</u>			<u>758867731</u>	<u>5645159321</u>	
$\frac{13}{20}$	<u>$\frac{246121993}{1340847787}$</u>	0	0	<u>$\frac{-37695042795}{15268766246}$</u>	<u>$\frac{-309121744}{1061227803}$</u>	<u>$\frac{-12992083}{490766935}$</u>	<u>$\frac{6005943493}{2108947869}$</u>	<u>$\frac{393006217}{1396673457}$</u>	<u>$\frac{123872331}{1001029789}$</u>		<u>561292985</u>	<u>656045339</u>	
$\frac{1201146811}{1299019798}$	<u>$\frac{-1028468189}{846180014}$</u>	0	0	<u>$\frac{8478235783}{508512852}$</u>	<u>$\frac{1311729495}{1432422823}$</u>	<u>$\frac{-10304129995}{1701304382}$</u>	<u>$\frac{-48777925059}{3047939560}$</u>	<u>$\frac{15336726248}{1032824649}$</u>	<u>$\frac{-45442868181}{3398467696}$</u>	<u>$\frac{3065993473}{597172653}$</u>	<u>797845732</u>	<u>265891186</u>	
1	<u>$\frac{185892177}{718116043}$</u>	0	0	<u>$\frac{-3185094517}{667107341}$</u>	<u>$\frac{-477755414}{1098053517}$</u>	<u>$\frac{-703635378}{230739211}$</u>	<u>$\frac{5731566787}{1027545527}$</u>	<u>$\frac{5232866602}{850066563}$</u>	<u>$\frac{-4093664535}{808688257}$</u>	<u>$\frac{3962137247}{1805957418}$</u>	<u>$\frac{65686358}{487910083}$</u>	<u>-1041891430</u>	<u>-3867574721</u>
1	<u>$\frac{403863854}{491063109}$</u>	0	0	<u>$\frac{-5068492393}{434740067}$</u>	<u>$\frac{-411421997}{543043805}$</u>	<u>$\frac{652783627}{914296604}$</u>	<u>$\frac{11173962825}{925320556}$</u>	<u>$\frac{-13158990841}{6184727034}$</u>	<u>$\frac{3936647629}{1978049680}$</u>	<u>$\frac{-160528059}{685178525}$</u>	<u>$\frac{248638103}{1413531060}$</u>	<u>1371343529</u>	<u>1518517206</u>
											<u>760417239</u>	<u>465885868</u>	
											<u>1151165299</u>	<u>322736535</u>	
											<u>118820643</u>	<u>53011238</u>	
											<u>751138087</u>	<u>667516719</u>	
											<u>-528747749</u>	<u>2</u>	
											<u>2220607170</u>	<u>45</u>	
											<u>$\frac{1}{4}$</u>	<u>0</u>	

P. J. Prince and J. R. Dormand, "High order embedded Runge-Kutta formulae," *Journal of Computational and Applied Mathematics*, Vol. 7, No. 1, 1981, pp. 67-75.

7th order

Match Point Time of Flight Derivatives

- If we had an analytic equations for propagating the trajectory, TOF derivatives would be fairly simple (to first order)

$$\delta \mathbf{r}_m = \mathbf{v}_m \cdot \delta TOF$$

- Unfortunately, we have a numerical approximation, so computing TOF derivatives involves taking derivatives of the components of the integrator itself
- Derivatives of both the adaptive step routine as well as the RK step calculation must be accumulated as the integration is performed

$$\frac{\partial \hat{\mathbf{X}}_{n+1}}{\partial TOF} = \frac{\partial \hat{\mathbf{X}}_n}{\partial TOF} + \sum_{i=1}^s \hat{b}_i \frac{\partial \mathbf{k}_i}{\partial TOF} \quad \dot{\mathbf{X}} = \mathbf{f} = \begin{bmatrix} \dot{\mathbf{r}} \\ \ddot{\mathbf{r}} \\ \dot{m} \end{bmatrix}$$

$$\frac{\partial \mathbf{k}_i}{\partial TOF} = \frac{\partial \delta_n}{\partial TOF} \mathbf{f} + \delta_n \frac{\partial \mathbf{f}}{\partial TOF}$$

$$\frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial TOF} = \frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial \mathbf{r}_{cb-s/c}} \cdot \frac{\partial \mathbf{r}_{cb-s/c}}{\partial TOF} + \frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial \mathbf{r}_{cb-3b}} \cdot \frac{\partial \mathbf{r}_{cb-3b}}{\partial TOF} + \frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial m_{s/c}} \cdot \frac{\partial m_{s/c}}{\partial TOF} + \frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial T} \cdot \frac{\partial T}{\partial TOF}$$

TOF Derivative Challenges

- State gradient TOF derivatives for the spacecraft in turn require sensitivity of 3rd body position to changes in TOF
- EMTG uses the SPICE kernels to calculate solar system body states
- Analytical derivatives are only possible if you can access SPICE internally
- SPICE uses Chebyshev polynomial interpolation for planetary positions
- Fortunately, for planets only, the velocity polynomial is the time derivative of the position polynomial (we have direct access to the position curve derivative)
- This is not true for planetary satellites, velocity curve is fit separately
 - Another ephemeris reader (FIRE, Arora and Russell, 2008) would solve this
- EMTG's power solar electric hardware models are TOF dependent

$$\frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial TOF} = \frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial \mathbf{r}_{cb-s/c}} \cdot \frac{\partial \mathbf{r}_{cb-s/c}}{\partial TOF} + \frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial \mathbf{r}_{cb-3b}} \cdot \frac{\partial \mathbf{r}_{cb-3b}}{\partial TOF} + \frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial m_{s/c}} \cdot \frac{\partial m_{s/c}}{\partial TOF} + \frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial T} \cdot \frac{\partial T}{\partial TOF}$$

TOF Derivative Challenges

- State gradient TOF derivatives for the spacecraft in turn require sensitivity of 3rd body position to changes in TOF
- EMTG uses the SPICE kernels to calculate solar system body states
- Analytical derivatives are only possible if you can access SPICE internally
- SPICE uses Chebyshev polynomial interpolation for planetary positions
- Fortunately, for planets only, the velocity polynomial is the time derivative of the position polynomial (we have direct access to the position curve derivative)
- This is not true for planetary satellites, velocity curve is fit separately
 - Another ephemeris reader (FIRE, Arora and Russell, 2008) would solve this
- EMTG's power solar electric hardware models are TOF dependent

$$\frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial TOF} = \frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial \mathbf{r}_{cb-s/c}} \cdot \frac{\partial \mathbf{r}_{cb-s/c}}{\partial TOF} + \frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial \mathbf{r}_{cb-3b}} \cdot \frac{\partial \mathbf{r}_{cb-3b}}{\partial TOF} + \frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial m_{s/c}} \cdot \frac{\partial m_{s/c}}{\partial TOF} + \frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial T} \cdot \frac{\partial T}{\partial TOF}$$

$$\frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial \mathbf{r}_{cb-3b}} = -\mu_{3b} \left[\frac{3\mathbf{r}_{3b-s/c}\mathbf{r}_{3b-s/c}^T}{r_{3b-s/c}^5} - \frac{\mathbb{I}_{3x3}}{r_{3b-s/c}^3} \right]$$

TOF Derivative Challenges

- State gradient TOF derivatives for the spacecraft in turn require sensitivity of 3rd body position to changes in TOF
- EMTG uses the SPICE kernels to calculate solar system body states
- Analytical derivatives are only possible if you can access SPICE internally
- SPICE uses Chebyshev polynomial interpolation for planetary positions
- Fortunately, for planets only, the velocity polynomial is the time derivative of the position polynomial (we have direct access to the position curve derivative)
- This is not true for planetary satellites, velocity curve is fit separately
 - Another ephemeris reader (FIRE, Arora and Russell, 2008) would solve this
- EMTG's power solar electric hardware models are TOF dependent

$$\frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial TOF} = \frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial \mathbf{r}_{cb-s/c}} \cdot \frac{\partial \mathbf{r}_{cb-s/c}}{\partial TOF} + \frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial \mathbf{r}_{cb-3b}} \cdot \frac{\partial \mathbf{r}_{cb-3b}}{\partial TOF} + \frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial m_{s/c}} \cdot \frac{\partial m_{s/c}}{\partial TOF} + \frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial T} \cdot \frac{\partial T}{\partial TOF}$$

TOF Derivative Challenges

- State gradient TOF derivatives for the spacecraft in turn require sensitivity of 3rd body position to changes in TOF
- EMTG uses the SPICE kernels to calculate solar system body states
- Analytical derivatives are only possible if you can access SPICE internally
- SPICE uses Chebyshev polynomial interpolation for planetary positions
- Fortunately, for planets only, the velocity polynomial is the time derivative of the position polynomial (we have direct access to the position curve derivative)
- This is not true for planetary satellites, velocity curve is fit separately
 - Another ephemeris reader (FIRE, Arora and Russell, 2008) would solve this
- EMTG's power solar electric hardware models are TOF dependent

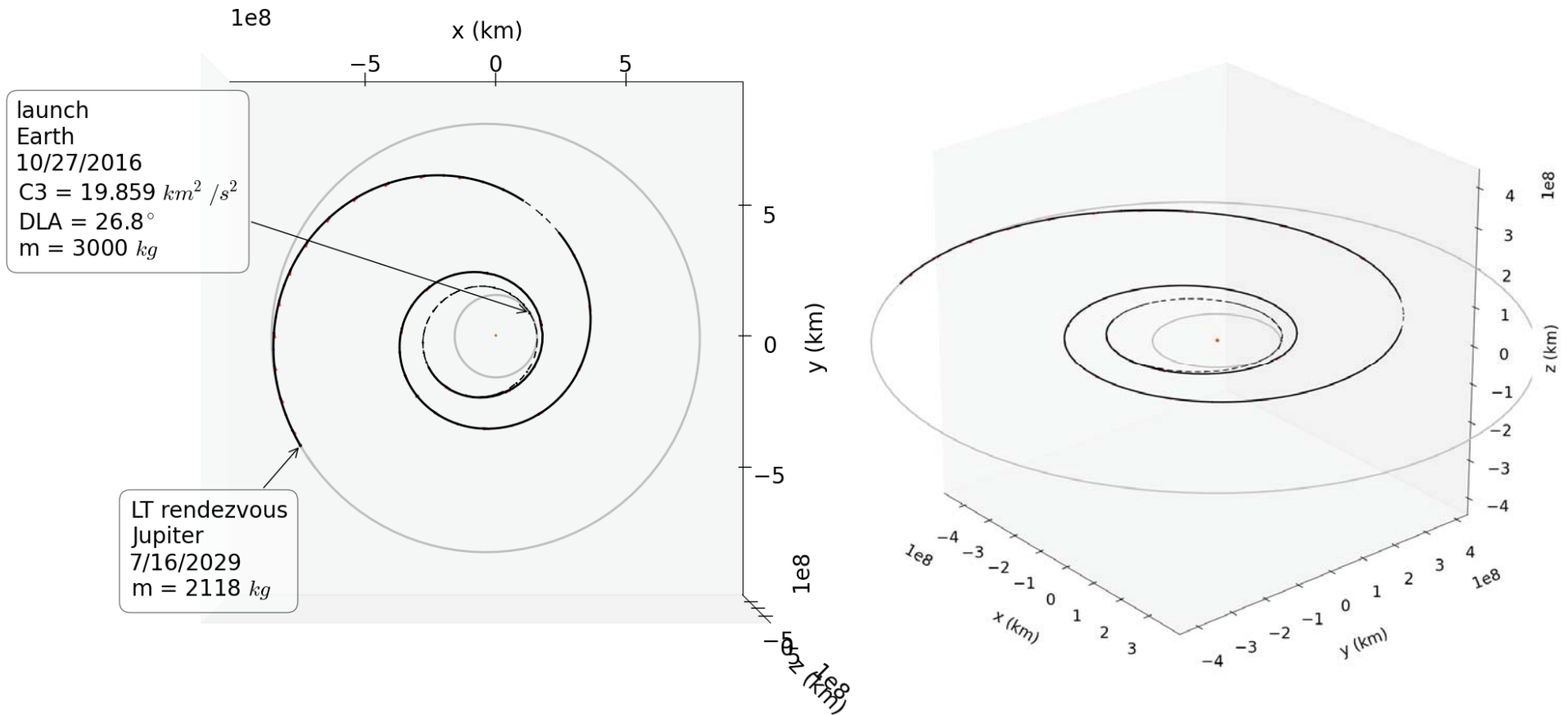
$$\frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial TOF} = \frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial \mathbf{r}_{cb-s/c}} \cdot \frac{\partial \mathbf{r}_{cb-s/c}}{\partial TOF} + \frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial \mathbf{r}_{cb-3b}} \cdot \frac{\partial \mathbf{r}_{cb-3b}}{\partial TOF} + \frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial m_{s/c}} \cdot \frac{\partial m_{s/c}}{\partial TOF} + \frac{\partial \ddot{\mathbf{r}}_{cb-s/c}}{\partial T} \cdot \frac{\partial T}{\partial TOF}$$

Example: Solar Electric Earth to Jupiter

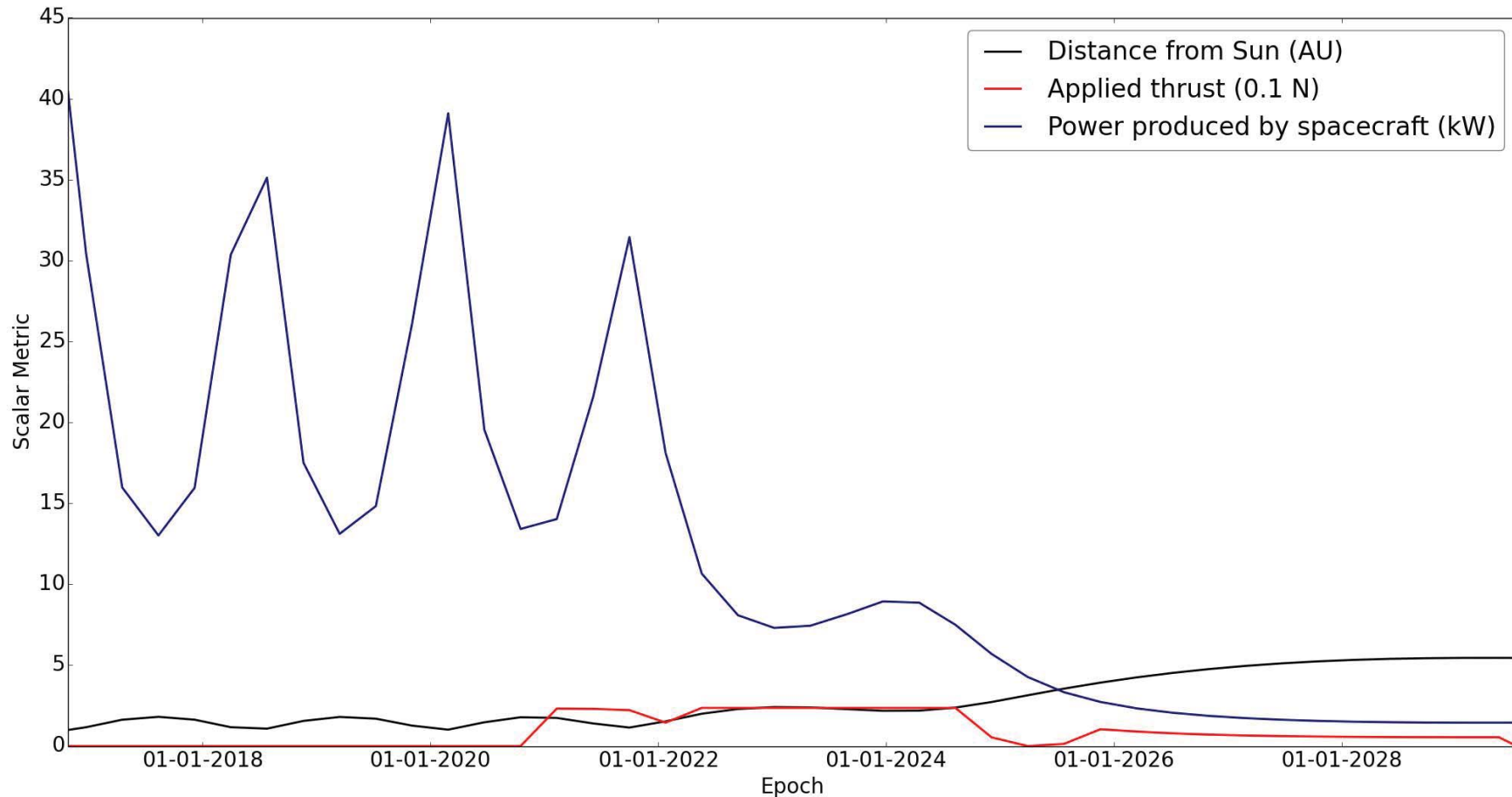
Parameter	Value
Initial mass	3000 kg
Earliest allowed launch date	January 1st, 2015
Latest allowed launch date	unbounded
Initial v_∞	up to 6.97 km/s
Maximum flight time	unbounded
Arrival type	low-thrust rendezvous
Thruster	NEXT - high thrust variant
Thrust coefficients	$e_T = 0.09591$ $d_T = -1.98537$ $c_T = 11.47980$ $b_T = 15.06977$ $a_T = 14.51552$
Thruster duty cycle	0.9
Mass flow rate coefficients	$e_F = 0.01492$ $d_F = -0.27539$ $c_F = 1.60966$ $b_F = -2.53056$ $a_F = 3.22089$
Solar power coefficients	$\gamma_0 = 1.32077$ $\gamma_1 = -0.10848$ $\gamma_2 = -0.11665$ $\gamma_3 = 0.10843$ $\gamma_4 = -0.01279$
Solver parameters	
Flyby sequence	Earth-Jupiter direct (E-J)
Number of time-steps	40
Ephemeris	SPICE
SNOPT feasibility tolerance	1.0e-5
Objective function	maximize final mass
Number of NLP parameters	126
Number of constraints (including objective function)	48
Dense Jacobian entries	1002

Table 2: Earth to Jupiter low-thrust direct transfer problem assumptions

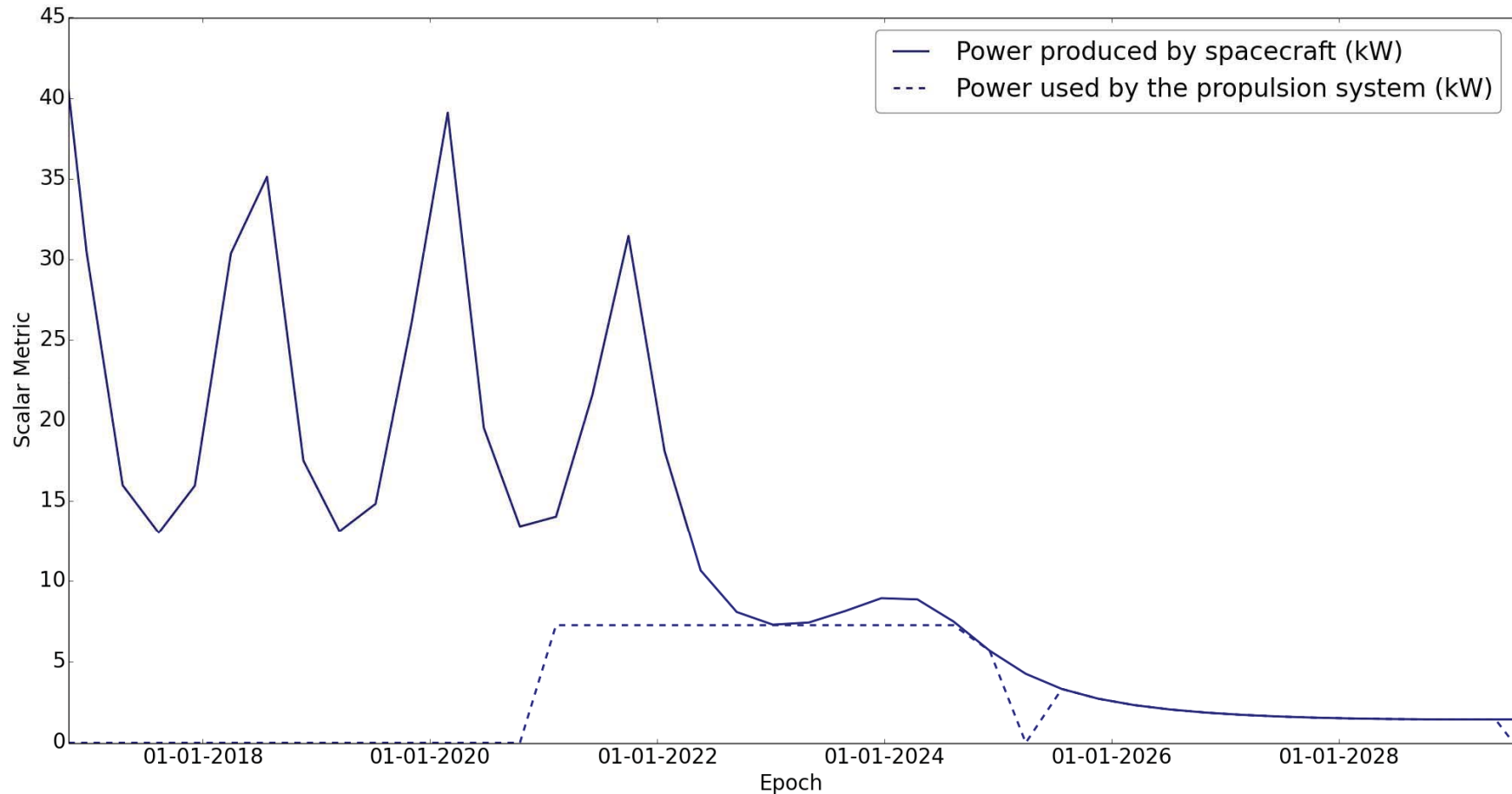
Example: Solar Electric Earth to Jupiter



Example: Solar Electric Earth to Jupiter



Example: Solar Electric Earth to Jupiter



Example: Solar Electric Earth to Jupiter

- SNOPT was allowed to execute for 100 major iterations
- STM-based Jacobian run converged after 80 majors
- Finite differenced Jacobian run hit the 100 major iteration limit
- Both are equally effective at solving the problem from an MGALT initial guess
- Numerically computed STMs afford 13-15 times execution speed increase

Metric	STM computed Jacobian	Finite Differenced Jacobian
Final mass delivered to Jupiter	2117.68 kg	2117.53 kg
SNOPT execution time	44.04 s	672.91 s

Table 3: STM vs. finite differenced Jacobian SNOPT performance metrics

Next Steps

- Low-thrust preliminary design cadence has been drastically increased
- EMTG-General Mission Analysis Tool (GMAT) work flow has been improved
- Although GMAT can converge from an MGALT or FBLT initial guess, for complex problems sometimes even an FBLT cannot converge from MGALT
- For cases when MGALT is insufficient, it is crucial that initial guesses for GMAT be generated as fast as possible using FBLT
- Launch vehicle, power and thruster models are currently being integrated with GMAT
- It is recommended that the STM described in this presentation be incorporated into GMAT's solvers as it can handle arbitrarily complex dynamics (i.e. SRP, gravitational harmonics could be added relatively easily)
- EMTG and GMAT native integrators are also quite similar



Conclusions

- The state transition matrix increases the efficiency of the finite-burn low-thrust transcription and allows for the tempo at which low-thrust preliminary design is performed to be increased, saving analysts' time.
- It importantly includes accurate hardware models that can have a major influence on mission feasibility even at the preliminary design stage
- This work enables FBLT to be used in the framework of a global optimizer for many problems
- With EMTG's performance increases serving as an example, the logical next step would be to repeat the same process inside the GMAT high-fidelity tool
- This work has extensions in other areas of interest:
 - Spacecraft guidance (method of adjoints and the guidance matrix)

Thank you!

This work is supported by Rich Burns and was completed under contract with NASA GSFC NNX14AD27G

EMTG is available open-source at:

<https://sourceforge.net/projects/emtg/>

