Engineering Notes

High-Speed Solution of Spacecraft Trajectory Problems Using Taylor Series Integration

James R. Scott*
NASA John H. Glenn Research Center at Lewis Field, Cleveland, Ohio 44135
and
Michael C. Martini†
Analex Corporation, Cleveland, Ohio 44135

DOI: 10.2514/1.43459

Introduction

IT HAS been known for some time that Taylor series (TS) integration is among the most efficient and accurate numerical methods in solving differential equations. However, the full benefit of the method has yet to be realized in calculating spacecraft trajectories, for two main reasons. First, most applications of Taylor series to trajectory propagation have focused on relatively simple problems of orbital motion or on specific problems and have not provided general applicability. Second, applications that have been more general have required use of a preprocessor, which inevitably imposes constraints on computational efficiency. The latter approach includes the work of Berryman et al. [1], who solved the planetary *n*-body problem with relativistic effects. Their work specifically noted the computational inefficiencies arising from use of a preprocessor and pointed out the potential benefit of manually coding derivative routines.

Montenbruck's work [2] offers the only trajectory application in which Taylor series integration was implemented in a general approach without requiring a preprocessor. Limited to Earth-orbiting satellites, Montenbruck's work included oblateness effects of Earth, lunar and solar gravitation, solar radiation pressure, and atmospheric density.

In this Engineering Note, we report on a systematic effort to directly implement Taylor series integration in an operational trajectory propagation code: the Spacecraft N-Body Analysis Program (SNAP) [3]. The present Taylor series implementation is unique in that it applies to spacecraft virtually anywhere in the solar system and can be used interchangeably with another integration method.

SNAP is a high-fidelity trajectory propagator that includes force models for central body gravitation with $N \times N$ harmonics, other body gravitation with $N \times N$ harmonics, solar radiation pressure, atmospheric drag (for Earth orbits), and spacecraft thrusting (including shadowing). The governing equations are solved using an eighth-order Runge–Kutta Fehlberg (RKF) [4] single-step method with variable step size control.

In the present effort, TS is implemented by way of highly integrated subroutines that can be used interchangeably with RKF. This makes it possible to turn TS on or off during various phases of a mission. Current TS force models include central body gravitation with the J2 spherical harmonic, other body gravitation, thrust, constant atmospheric drag from Earth's atmosphere, and solar radiation pressure for a sphere under constant illumination.

The purpose of this Engineering Note is to demonstrate the performance of TS integration in an operational trajectory analysis code and to compare it with a standard method, eighth-order RKF. Results show that TS is 16.6 times faster on average and is more accurate in 87.5% of the cases presented.

Equations of Motion

Let $\boldsymbol{X} = (x_1, x_2, x_3, x_4, x_5, x_6, x_7)$ denote the spacecraft state vector, where $(x_1, x_2, x_3) = \boldsymbol{x}$ is the spacecraft position in Cartesian coordinates relative to an inertial frame centered at the central body, $(x_4, x_5, x_6) = \boldsymbol{v}$ is the spacecraft velocity relative to an inertial frame centered at the central body, and x_7 is the spacecraft mass. The equations of motion are

$$\boldsymbol{x}' = \boldsymbol{v} \quad \boldsymbol{v}' = \boldsymbol{a}(x_1, x_2, x_3, x_4, x_5, x_6, x_7, t) \quad x_7' = -\dot{m}(t) \tag{1}$$

where $\boldsymbol{a}$ is the acceleration and $\dot{m}$ is the mass flow rate. $\boldsymbol{a}$ is the sum of accelerations from the central body point mass, other body point masses, thrust, atmospheric drag, solar radiation pressure, oblateness effects of the central body, and oblateness effects of other bodies, so that

$$\boldsymbol{a} = \boldsymbol{a}_{cb} + \boldsymbol{a}_{ob} + \boldsymbol{a}_{th} + \boldsymbol{a}_{d} + \boldsymbol{a}_{srp} + \boldsymbol{a}_{obc} + \boldsymbol{a}_{obo} \tag{2}$$

Taylor Series Formulation

Let the state vector $\mathbf{X}$ have the initial condition $\boldsymbol{X}_0$. Within the radius of convergence, the system variables $x_n(t)$ can be expanded in a Taylor series:

$$x_n(t) = \sum_{k=0}^{\infty} \frac{x_n^{(k)}(t_0)}{k!}(t - t_0)^k, \qquad n = 1, \ldots, 7 \tag{3}$$

where the reduced derivatives $\frac{x_n^{(k)}(t_0)}{k!} := X_n(k)$ are obtained by successively differentiating the right-hand side of Eqs. (1). The reduced derivatives can be inexpensively obtained by reformulating the right-hand side and making use of highly efficient differentiation arithmetic. For example, for a function $w(t) = f(t)g(t)$, one uses [5]

$$W(k) = \sum_{j=0}^{k} F(j)G(k - j) \tag{4}$$

where W, F, and G are reduced derivatives. For $w(t) = f(t)/g(t)$, one uses [6]

$$W(k) = \frac{1}{g}\left[F(k) - \sum_{j=1}^{k} G(j)W(k - j)\right] \tag{5}$$

The system variables may then be expanded in a series of arbitrary degree K

$$x_n(t) = \sum_{k=0}^{K} X_n(k)(t - t_0)^k \tag{6}$$

for $n = 1, 2, \ldots, N$, where N is the number of equations in the reformulated system. From t_0, the solution is advanced to t_1, where

Presented as Paper 6957 at the AIAA/AAS Astrodynamics Specialist Conference and Exhibit, Honolulu, HI, 18–21 August 2008; received 27 January 2009; revision received 11 August 2009; accepted for publication 16 August 2009.

*Senior Research Scientist/Systems Engineer, Systems Engineering and Analysis Division, 21000 Brookpark Road, Mail Stop 86-1. Senior Member AIAA.

†Aerospace Engineer, Systems Engineering and Analysis Division, 21000 Brookpark Road, Mail Stop 500-103.

Table 1 Trajectory problems

Problem	Description	Central body	Other bodies
1	A 10,000 kg satellite orbits Earth for 10 days at an inclination of 28.45 deg.	Earth	Sun, moon
2	Problem 1 with constant atmospheric drag.	Earth	Sun, moon
3	Problem 2 including the J2 spherical harmonic.	Earth	Sun, moon
4	A 10,000 kg spacecraft spirals out of Earth's gravity well in a low-thrust trajectory. Calculation stops when the semimajor axis of trajectory equals 40,000 km. Includes J2 spherical harmonic.	Earth	Sun, moon
5	A 3580 kg spacecraft 400 km above Earth has been propelled with sufficient energy to reach the moon. Spacecraft coasts to moon, performs insertion burn, propagates to apolune, and performs final burn to achieve 500 km by 10,000 km polar lunar orbit with an argument of perilune equal to 90 deg.	Moon	Earth, sun
6	Spacecraft with 2848.56 kg mass coasts for 10 days in 500 km by 10,000 km polar lunar orbit with an argument of perilune equal to 90 deg.	Moon	Earth, sun
7	A 585 kg spacecraft near Earth thrusts for 38.45 days to achieve sufficient energy to coast to Mars. Includes constant solar radiation pressure for a sphere.	Sun	Earth, moon, Venus, Mars, Jupiter barycenter, Saturn barycenter
8	A 555.66 kg spacecraft coasts to Mars flyby for 161.55 days. Includes constant solar radiation pressure for a sphere.	Sun	Earth, moon, Venus, Mars, Jupiter barycenter, Saturn barycenter
9	A 10,000 kg spacecraft in Europa orbit thrusts tangentially to spiral out until the semimajor axis equals 10,000 km.	Europa	Jupiter, Sun, Ganymede, Io, Callisto
10	A 9800.49 kg spacecraft coasts for 1 day after spiraling out of Europa orbit.	Europa	Jupiter, Sun, Ganymede, Io, Callisto

the step size $h = t_1 - t_0$ is determined to meet the error tolerance. From t_1, the variables are expanded to t_2, and so forth. Thus, by a process of "analytic continuation," one obtains a set of overlapping series solutions that cover the integration domain.

Here h is chosen so that each state vector component x_n satisfies

$$|X_n(K-1)|h^{K-1} + |X_n(K)|h^K \leq |x_n|\tau + \tau \tag{7}$$

where τ is the error tolerance parameter.

Results

We consider the trajectory problems in Table 1. All calculations were run on a Dell PowerEdge 2600 with two 3.066 GHz processors and 4 GB of RAM. The source code was compiled using the Absoft Fortran 90 compiler without optimization. SNAP was run with all intermediate print and stop options turned off. All TS calculations used a series with 20 terms.

Problems 1–10 were solved for five values of τ from 10^{-10} to 10^{-14}. RKF results were obtained using the error tolerance $\tau_n = |x_n|\tau + \tau$ for each state vector component.

Table 2 RKF CPU times (seconds)

	Problem									
τ	1	2	3	4	5	6	7	8	9	10
1.E − 10	9.03	9.10	9.25	44.63	0.44	1.99	0.16	0.19	0.79	0.12
1.E − 11	11.94	12.01	12.18	58.44	0.51	2.64	0.18	0.20	1.00	0.14
1.E − 12	15.88	16.18	16.00	78.37	0.59	3.49	0.23	0.22	1.33	0.17
1.E − 13	21.25	21.17	21.70	103.49	0.71	4.62	0.29	0.26	1.74	0.22
1.E − 14	28.16	27.91	28.19	135.53	0.87	6.15	0.38	0.32	2.25	0.27

Table 3 TS CPU times (seconds)

	Problem									
τ	1	2	3	4	5	6	7	8	9	10
1.E − 10	0.25	0.32	0.87	3.69	0.19	0.21	0.09	0.10	0.10	0.03
1.E − 11	0.27	0.32	0.99	4.13	0.19	0.23	0.10	0.11	0.12	0.03
1.E − 12	0.31	0.36	1.10	4.68	0.18	0.26	0.10	0.13	0.13	0.04
1.E − 13	0.35	0.41	1.25	5.23	0.18	0.29	0.12	0.15	0.14	0.04
1.E − 14	0.39	0.46	1.40	5.90	0.20	0.33	0.12	0.16	0.15	0.05

Table 4 CPU ratios RKF/TS

	Problem									
τ	1	2	3	4	5	6	7	8	9	10
1.E − 10	36.12	28.44	10.63	12.09	2.32	9.48	1.78	1.90	7.90	4.00
1.E − 11	44.22	37.53	12.30	14.15	2.68	11.48	1.80	1.82	8.33	4.67
1.E − 12	51.23	44.94	14.55	16.75	3.28	13.42	2.30	1.69	10.23	4.25
1.E − 13	60.71	51.63	17.36	19.79	3.94	15.93	2.42	1.73	12.43	5.50
1.E − 14	72.21	60.67	20.14	22.97	4.35	18.64	3.17	2.00	15.00	5.40

Table 5 RKF convergence results

	Problem									
τ	1	2	3	4	5	6	7	8	9	10
1.E − 10	0.43E − 06	0.43E − 06	0.42E − 06	0.13E − 04	0.86E − 07	0.11E − 05	0.39E − 11	0.66E − 10	0.10E − 08	0.97E − 10
1.E − 11	0.35E − 07	0.35E − 07	0.34E − 07	0.11E − 05	0.23E − 08	0.87E − 07	0.12E − 11	0.38E − 11	0.94E − 10	0.14E − 10
1.E − 12	0.27E − 08	0.27E − 08	0.27E − 08	0.82E − 07	0.99E − 09	0.70E − 08	0.61E − 12	0.40E − 12	0.85E − 11	0.11E − 11
1.E − 13	0.19E − 09	0.20E − 09	0.21E − 09	0.56E − 08	0.50E − 08	0.55E − 09	0.69E − 12	0.12E − 12	0.94E − 12	0.18E − 12

Table 6 TS convergence results

	Problem									
τ	1	2	3	4	5	6	7	8	9	10
1.E − 10	0.25E − 07	0.26E − 07	0.23E − 07	0.56E − 06	0.71E − 08	0.40E − 07	0.15E − 11	0.66E − 12	0.12E − 09	0.25E − 10
1.E − 11	0.28E − 08	0.29E − 08	0.13E − 07	0.14E − 06	0.18E − 10	0.24E − 08	0.27E − 12	0.21E − 12	0.81E − 11	0.30E − 11
1.E − 12	0.27E − 09	0.31E − 09	0.90E − 09	0.68E − 06	0.15E − 10	0.22E − 09	0.22E − 13	0.27E − 12	0.45E − 12	0.11E − 12
1.E − 13	0.25E − 10	0.43E − 10	0.22E − 07	0.16E − 06	0.12E − 10	0.20E − 10	0.22E − 14	0.25E − 12	0.27E − 11	0.46E − 13

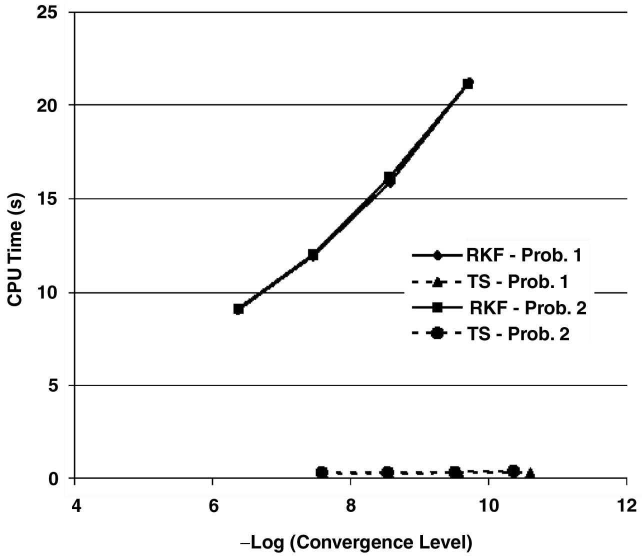

Fig. 1 CPU time vs valid digits for problems 1 and 2.

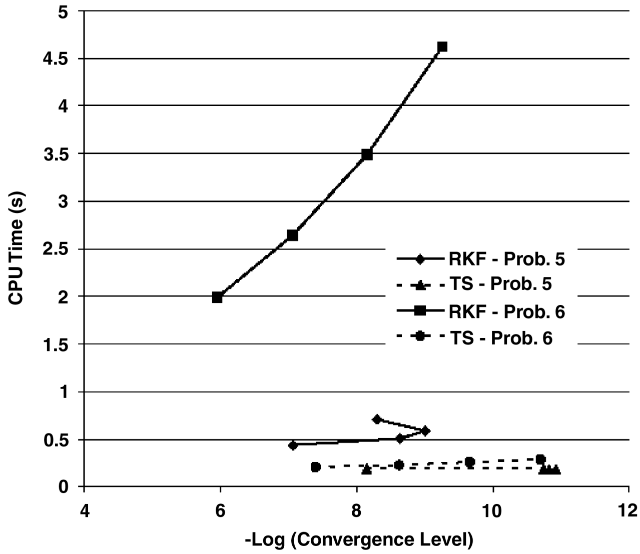

Fig. 3 CPU time vs valid digits for problems 5 and 6.

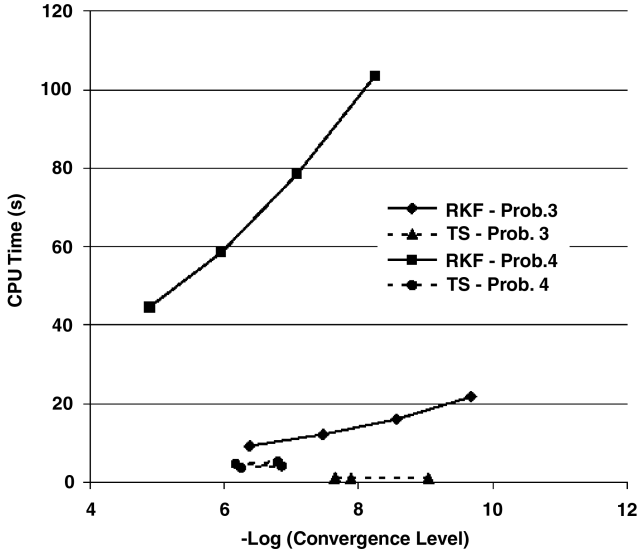

Fig. 2 CPU time vs valid digits for problems 3 and 4.

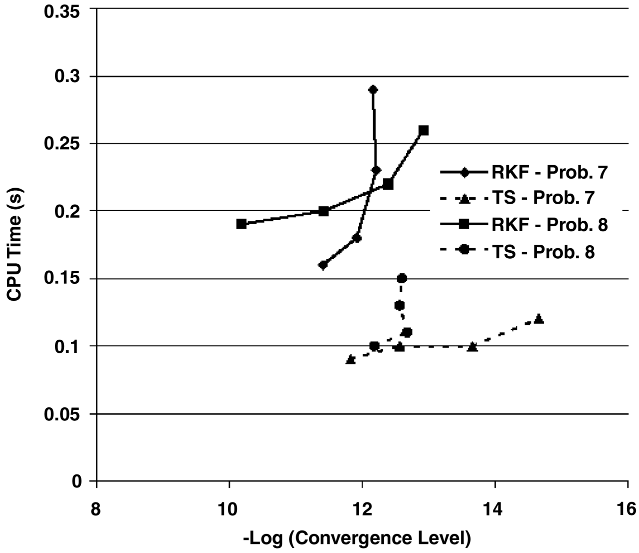

Fig. 4 CPU time vs valid digits for problems 7 and 8.

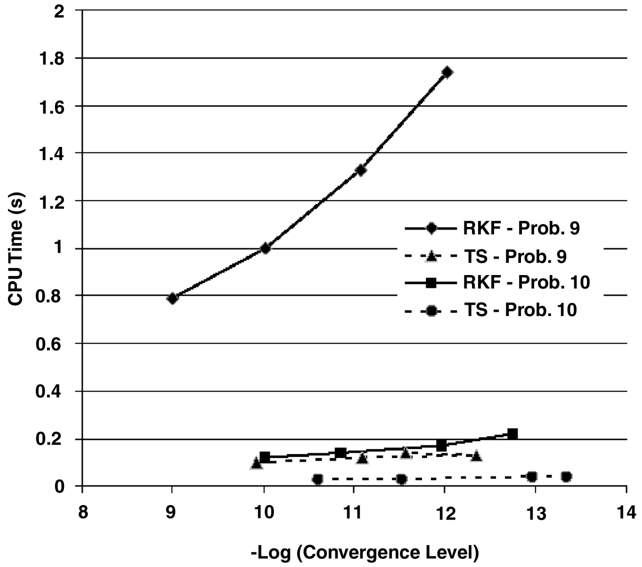

Fig. 5 CPU time vs valid digits for problems 9 and 10.

Table 7 Difference between RKF and TS solution

Problem	Relative difference	Absolute difference, km
1	0.300E − 10	0.203E − 06
2	0.193E − 10	0.131E − 06
3	0.383E − 07	0.259E − 03
4	0.112E − 07	0.447E − 03
5	0.212E − 04	0.248E + 00
6	0.793E − 06	0.185E − 02
7	0.664E − 08	0.103E + 01
8	0.321E − 06	0.693E + 02
9	0.516E − 04	0.395E + 00
10	0.837E − 04	0.197E + 01

Tables 2 and 3 present the CPU times for RKF and TS, respectively. Table 4 shows the ratio of RKF/TS CPU times. The average speed up for TS is 16.6.

Tables 5 and 6 show the relative L_2 convergence of the final spacecraft position, that is, $|\boldsymbol{x} - \boldsymbol{x}_f|/|\boldsymbol{x}_f|$ where $\boldsymbol{x}_f$ is the final spacecraft position corresponding to $\tau = 10^{-14}$ and $\boldsymbol{x}$ is the final spacecraft position corresponding to larger τ. TS is more converged than RKF in 35 out of 40 cases. Figures 1–5 plot CPU times versus convergence levels; the results are presented in pairs to save space. It should be noted that the impressive results in Figs. 1–5 would be somewhat less dramatic if TS were compared with a state-of-the-art Runge–Kutta solver such as DOPRI8 [7].

Table 7 shows the difference between RKF and TS in predicting final spacecraft position. The differences are primarily due to the fact that TS integrates the motion of other bodies, whereas RKF uses Jet Propulsion Laboratory DE405 Planetary and Lunar Ephemerides ephemeris files. The relatively small differences show that the TS approach of integrating other body motion is a viable alternative to using ephemeris files.

Conclusions

Direct implementation of Taylor series integration in an operational trajectory analysis code leads to major improvements in both computational speed and accuracy over a conventional eighth order Runge–Kutta method. Speed improvements are demonstrated on a wide variety of problems, including those with oblateness effects and solar radiation pressure. The main drawback is the need to extract derivative information from atmospheric density models. This is an area that requires further study. Overall, Taylor series offers the potential for very large reductions in computational time while simultaneously improving accuracy in trajectory propagation problems.

Acknowledgment

The authors would like to thank Glen Horvat and Kurt Hack for their support of this work.

References

[1] Berryman, K. W., Stanford, R. H., and Breckheimer, P. J., "The ATOMFT Integrator: Using Taylor Series to Solve Ordinary Differential Equations," *AIAA/American Astronautical Society Astrodynamics Conference*, AIAA Paper 88-4217-CP, Aug. 1988.

[2] Montenbruck, O., "Numerical Integration of Orbital Motion Using Taylor Series," *American Astronautical Society/AIAA Spaceflight Mechanics Meeting*, American Astronautical Society Paper 92-195, Feb. 1992.

[3] Martini, M., "S.N.A.P. 2.3 User's Manual, Spacecraft N-Body Analysis Program," NASA, Washington, DC, Dec. 2005; also https://technology.grc.nasa.gov/software/ [retrieved 2 Sept. 2009].

[4] Fehlberg, E., "Classical Fifth-, Sixth-, Seventh-, and Eighth-Order Runge–Kutta Formulas with Stepsize Control," NASA TR R-287, Oct. 1968; also http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19680027281_1968027281.pdf [retrieved 2 Sept. 2009].

[5] Corliss, G., and Chang, Y. F., "Solving Ordinary Differential Equations Using Taylor Series," *ACM Transactions on Mathematical Software*, Vol. 8, No. 2, 1982, pp. 114–144.
doi:10.1145/355993.355995

[6] Barton, D., Willers, I. M., and Zahar, R. V. M., "Taylor Series Methods for Ordinary Differential Equations—An Evaluation," *Mathematical Software*, edited by J. Rice, Academic Press, New York, 1971, pp. 372.

[7] Dormand, J. R., and Prince, P. J., "A Family of Embedded Runge–Kutta Formulae," *Journal of Computational and Applied Mathematics*, Vol. 6, March 1980, pp. 19–26.

B. Marchand
Associate Editor