A Parallel Computer Numerical Simulation Method Based on Coincident Coefficients

Wenhui Cui\textsuperscript{1}, Yonghua Li\textsuperscript{1}, Zhenyu Sun\textsuperscript{1}

\textsuperscript{1}State Key Laboratory of Astronautic Dynamics, Xi’an 710043, China

Abstract. Analyze the coefficients relationship of 3rd order Runge-Kutta method and 4th order Runge-Kutta method, a parallel 4th order Runge-Kutta method based on consistent coefficient is proposed, the new parallel method can simultaneously satisfy the coefficients relationship both the 3rd order Runge-Kutta method and 4th order Runge-Kutta method with absolute stability region, it can be further parallel to lower order Runge-Kutta method. The method can be implemented in the multi-process MPI framework based on clusters. The method is applied to the numerical integral simulation of spacecraft orbit prediction, and the results of STK software are used as reference. The experimental results show that the accuracy of the parallel method is consistent with the serial method, and the computation speed is significantly higher than the serial method.

1. Introduction

Runge-Kutta method is one of the most widely used computer numerical simulation methods, especially in the solution about ordinary differential equations (abbreviated to ODE) and differential-algebraic equations (abbreviated to DAEs). In general, people are used to simplifying the DAEs to the ODE and solve it with the standard Runge-Kutta method\textsuperscript{[1][2][3]}

The common form of Runge-Kutta method is:

\begin{align}
  y_{i+1} &= y_i + c_i K_i + c_2 K_2 + \cdots + c_p K_p \\
  K_i &= hf(x_i, y_i) \\
  K_2 &= hf(x_i + a_2 h, y_i + b_2 K_i) \\
  \vdots \\
  K_p &= hf(x_i + a_p h, y_i + b_p K_1 + \cdots + b_{p-1} K_{p-1})
\end{align}

The above equation is $p$ order Runge-Kutta method, where the $a_i, b_i, c_i$ are undetermined parameters. The values of $a_i, b_i, c_i$ are determined by Gauss and Radau quadratic, they show the nodes and the terms weight of Runge-Kutta method\textsuperscript{[1][4][5]}. Other methods are developed based on the standard Runge-Kutta method to ensure that the numerical solution satisfies the additional constraints of the equation. One of the main development of these methods is the Specialized Partitioned Additive Runge-Kutta method (SPARK) proposed by Jay\textsuperscript{[6][7]}. For general smoothing functions, the Runge-Kutta method is more accurate than the Euler method, Midpoint method, and other integral methods. However, its operation speed is slow in the high order because of the serial calculation process.

To solve the problem of low computing speed of Runge-Kutta method, Miranker and Liniger discussed the parallelism problems of 3rd order Runge-Kutta method\textsuperscript{[8]}, proposed the basic method and coefficients determination equation of 3rd order Runge-Kutta method, and analyzed the absolute stability region of parallel and serial integration methods. FeiJ proposed the 3rd order parallel Runge-
Kutta method with absolute stability region[9], and realized the parallelization of the 3rd order Runge-Kutta method. Bleszynski E. et al proposed an adaptive integration method AIM. The AIM provides a fast iterative integral equation solver significantly reduces the storage and solution time compared with the conventional method of moments[10]. Guo H. et al proposed a parallel solution of the geomagnetic field integral equation based on the OpenMP framework of the clusters[11]. Enenkel R. F. et al implemented the diagonal implicit parallel improvement of the Runge-Kutta method[12]. Michielssen E. et al proposed a multistage matrix decomposition algorithm to quickly solve ODE[13][14][15]. In this manuscript, a parallel 4th order Runge-Kutta method with absolute stability region is deduced based on the conceptual principle of parallel integration.

2. The coefficient conditions of serial Runge-Kutta methods

The common form of 3rd order Runge-Kutta method is:

\[
\begin{align*}
y_{n+1} &= y_n + h(c_1k_1 + c_2k_2 + c_3k_3) \\
k_1 &= f(t_n, y_n) \\
k_2 &= f(t_n + a_2h, y_n + b_2hk_1) \\
k_3 &= f(t_n + a_3h, y_n + h(b_3k_1 + b_3k_2)) \\
a_i &= \sum_{j=1}^{i-1} b_j, \quad i = 2, 3
\end{align*}
\]

The common form of 4th order Runge-Kutta method is:

\[
\begin{align*}
y_{n+1} &= y_n + h(c_1k_1 + c_2k_2 + c_3k_3 + c_4k_4) \\
k_1 &= f(t_n, y_n) \\
k_2 &= f(t_n + a_2h, y_n + b_2hk_1) \\
k_3 &= f(t_n + a_3h, y_n + h(b_3k_1 + b_3k_2)) \\
k_4 &= f(t_n + a_4h, y_n + h(b_4k_1 + b_4k_2 + b_4k_3)) \\
a_i &= \sum_{j=1}^{i-1} b_j, \quad i = 2, 3, 4
\end{align*}
\]

According to the definition of truncation error in numerical analysis, the coefficients relationship can be obtained by the following steps:

1. \(y(x_{n+1})\) is expanded by Taylor series for unary function, it is the true value of \(y\) at point \(x_{n+1}\);
2. The right function of \(k_1, k_2, k_3\) and \(k_4\) is expanded by Taylor series for binary function, the expanded expression of \(y_{n+1}\) is obtained, it is the approximate value of \(y\) at point \(x_{n+1}\) by using the Runge-Kutta method;
3. Take the truncation error \(E = y(x_{n+1}) - y_{n+1}\), so the \(a_i, b_j, c_i\) that minimize \(E\) are the optimal coefficients of the Runge-Kutta method.

According to the above method, the coefficients relationship of equation (2) is as follows:

\[
\begin{align*}
c_1 + c_2 + c_3 &= 1 \\
c_2k_2 + c_3k_3 &= \frac{1}{2} \quad (4) \\
c_2k_2^2 + c_3k_3^2 &= \frac{1}{3} \\
c_3k_3^3 &= \frac{1}{6}
\end{align*}
\]

the coefficient relation of equation (3) is as follows:
\[
\begin{align*}
\begin{cases}
c_1 + c_2 + c_3 + c_4 &= 1 \\
c_2 \lambda_2 + c_1 \lambda_3 + c_4 \lambda_4 &= \frac{1}{2} \\
c_2 \lambda_2^2 + c_1 \lambda_3^2 + c_4 \lambda_4^2 &= \frac{1}{3} \\
c_2 \lambda_2^3 + c_1 \lambda_3^3 + c_4 \lambda_4^3 &= \frac{1}{4} \\
c_2 \mu_{32} + c_1 (\mu_{43} \lambda_3 + \mu_{42} \lambda_2) &= \frac{1}{6} \\
c_2 \lambda_2^2 \mu_{32} + c_1 (\mu_{43}^2 \lambda_3^2 + \mu_{42} \lambda_2^2) &= \frac{1}{12} \\
c_2 \lambda_2 \lambda_3 \mu_{52} + c_1 (\mu_{43} \lambda_3 \lambda_4 + \mu_{42} \lambda_2 \lambda_4) &= \frac{1}{8} \\
c_2 \mu_{52} \mu_{43} \lambda_2 &= \frac{1}{24}
\end{cases}
\end{align*}
\]

These coefficients are all undetermined coefficients for \( n \) order Runge-Kutta method, include \( c_i (i = 1, \cdots, n) \), \( \lambda_i (i = 2, \cdots, n) \) and \( \mu_i (i = 3, \cdots, n; j = 2, \cdots, i - 1) \), the value of these coefficients is mainly determined by the improved Euler method.

The most common form of the 4th order Runge-Kutta method is as follows:

\[
\begin{align*}
k_1 &= f(t_n, y_n) \\
k_2 &= f(t_n + \frac{1}{2} h, y_n + \frac{1}{2} hk_1) \\
k_3 &= f(t_n + \frac{1}{2} h, y_n + \frac{1}{2} hk_2) \\
k_4 &= f(t_n + h, y_n + hk_3) \\
y_{n+1} &= y_n + \frac{1}{6} h(k_1 + 2k_2 + 2k_3 + k_4)
\end{align*}
\]

There are infinite number of coefficients that adapt to the \( n \) order Runge-Kutta method in fact, so the coefficients suitable for the Runge-Kutta method are a collection. The first three coefficients \( k_1, k_2 \) and \( k_3 \) of the 4th order Runge-Kutta method are identical with the corresponding parameters of the 3rd order Runge-Kutta method in form. The form consistency is the basis of the parallelization of the highorder Runge-Kutta method by using the loworder Runge-Kutta method.

3. The Parallel 4th order Runge-Kutta method

The basic idea of Parallel Runge-Kutta method is: Based on the form consistency of \( n \) order Runge-Kutta equation and \( n+1 \) order Runge-Kutta equation, the \( n \) order Runge-Kutta equation is used to calculate the first \( n \) coefficients of \( n+1 \) order Runge-Kutta equation approximately, and calculate the \( k_{n+1} \) by \( n+1 \) order Runge-Kutta equation, so the complete \( n+1 \) order Runge-Kutta equation is obtained. In this course, the first \( n \) coefficients of \( n \) order Runge-Kutta equation \( y^{(n)} \) and the \( n+1 \) order Runge-Kutta equation \( y^{(n+1)} \) is the same[9].

The Runge-Kutta method is an approximate solution of ODE obtained by truncating the higher order differential term of the Taylor series. For ODE, the solutions of \( n \) order and \( n+1 \) order Runge-Kutta method are different in accuracy, but the same in essence. If the truncation error of \( n \) order is required, the Runge-Kutta method of \( n \) order and larger order is applicable to the operation process, and the results can meet the requirements properly. After parallel processing of the integration of \( n+1 \) order, the accuracy of its solution is higher than that of \( n \) order Runge-Kutta method but lower than that of \( n+1 \) order Runge-Kutta method, and the truncation error of \( n \) order method is still satisfied. When the truncation error order of the final result required by the calculation is
\(n - m (m \leq n - 2)\) order, for the \(n + 1\) order Runge-Kutta method, the parallel decomposition process can be carried out for \(m + 1\) times, that is, the \(n\) order Runge-Kutta method can be parallel to the operation process of \(n - m\) order at most.

The expression of the parallel 4th order Runge-Kutta method is as follows:

\[
\begin{align*}
    k_{1,n+1} &= f(t_{n+1}, y_{n+1}^{(3)}) \\
    k_{2,n+1} &= f(t_{n+1} + a_2h, y_{n+1}^{(3)} + b_2hk_{1,n+1}) \\
    k_{3,n+1} &= f(t_{n+1} + a_3h, y_{n+1}^{(3)} + h(b_3k_{1,n+1} + b_2k_{2,n+1})), \\
    y_{n+1}^{(3)} &= y_{n+1} + h(c_1k_{1,n+1} + c_2k_{2,n+1} + c_3k_{3,n+1}) \\
    k_{4,n+1} &= f(t_{n+1} + a_4h, y_{n+1}^{(4)} + h(b_4k_{1,n} + b_2k_{2,n} + b_3k_{3,n})). \\
    y_{n+1}^{(4)} &= y_{n+1} + h(c_1k_{1,n} + c_2k_{2,n} + c_3k_{3,n} + c_4k_{4,n}).
\end{align*}
\]

All parameters in the above equation meet all requirements for parameters in (4) and (5).

The steps of parallel Runge-Kutta method is as follows:

1. Select the proper values of coefficients \(a_i, b_j, c_j\), construct the parallel 4th order Runge-Kutta equation;
2. In equation (7), constitute the standard 4th order Runge-Kutta equation with the first three lines and the last two lines of equation (7), calculate the \(k_{1,n}, k_{2,n}, k_{3,n}\) and \(y_{n+1}^{(3)}\);
3. Calculate the value of \(k_{1,n+1}, k_{2,n+1}, k_{3,n+1}\) and \(y_{n+2}^{(3)}\) based on \(t_{n+1}\) and \(y_{n+1}^{(3)}\);
4. Calculate \(k_{4,n}\) based on \(t_n\) and \(y_n^{(4)}\);
5. Calculate \(y_{n+1}^{(4)}\) based on \(k_{1,n}, k_{2,n}, k_{3,n}\) and \(k_{4,n}\);
6. Let \(n = n + 1\), repeat (3)-(5), calculate the \(y_{n+2}^{(4)}\).

The ergodic calculation is carried out within the interval [-1,2] to find the parallel algorithm coefficients that satisfy equations (4) and (5) simultaneously. There is no exact solution to the equations, but the approximate solution can be obtained within a certain range of error. When the calculation errors \(\delta < 0.03\) of each equation of (4) and (5) are satisfied, the parallel 4th order Runge-Kutta method can be obtained as follows:

\[
\begin{align*}
    k_{1,n+1} &= f(t_{n+1}, y_{n+1}^{(3)}) \\
    k_{2,n+1} &= f(t_{n+1} + 0.4h, y_{n+1}^{(3)} + 0.4hk_{1,n+1}) \\
    k_{3,n+1} &= f(t_{n+1} + 0.3h, y_{n+1}^{(3)} - 1.78375h k_{1,n+1} + 2.08375hk_{2,n+1}) \\
    y_{n+1}^{(3)} &= y_{n+1}^{(3)} + h(-k_{1,n+1} + 1.8k_{2,n+1} + 0.2k_{3,n+1}) \\
    k_{4,n+1} &= f(t_{n+1} + h, y_{n+1}^{(4)} - 0.5hk_{1,n} + 1.1hk_{2,n} + 0.4hk_{3,n}) \\
    y_{n+1}^{(4)} &= y_{n+1}^{(4)} + h(0.7k_{2,n} + 0.1k_{3,n} + 0.2k_{4,n}).
\end{align*}
\]

In the study of stability problems, the model equation \(\dot{y} = \lambda y\) is generally used[9]. To guarantee the stability of the differential equation, it is assumed that \(\lambda < 0\). For an integral algorithm with fixed parameters, the calculation process of \(y_n\) integral to \(y_{n+1}\) and \(y_{n+1}\) integral to \(y_{n+2}\) is exactly equivalent, and its absolute stable region is exactly equivalent. To simplify the analysis of the problem, \(y_{n+1}^{(3)}\) and \(y_{n+2}^{(3)}\) can be replaced by \(y_{n}^{(3)}\) and \(y_{n+1}^{(3)}\). So the equation can be obtained as follows:

\[
\begin{align*}
    y_{n+1}^{(3)} &= (1 + h\lambda + 0.78h^2\lambda^2 + 0.1667h^3\lambda^3)y_{n}^{(3)} \\
    y_{n+2}^{(4)} &= (1 + 0.2h\lambda)y_{n}^{(4)} + (0.8\lambda h + 0.5h^2\lambda^2 + 0.32735h^3\lambda^3 + 0.01667h^4\lambda^4)y_{n}^{(3)}.
\end{align*}
\]

The equations can be expressed as matrix:
\[
\begin{bmatrix}
y_{n+1}^{(3)} \\
y_{n+1}^{(4)}
\end{bmatrix} =
\begin{bmatrix}
1 + h\lambda + 0.78h^2\lambda^2 + 0.1667h^3\lambda^3 & 0 \\
0.8\lambda h + 0.5h^2\lambda^2 + 0.32735h^3\lambda^2 + 0.01667h^4\lambda^3 & 1 + 0.2\lambda h
\end{bmatrix}
\begin{bmatrix}
y_{n}^{(3)} \\
y_{n}^{(4)}
\end{bmatrix}
\]

(10)

The eigenvalue of the matrix is \(1 + h\lambda + 0.78h^2\lambda^2 + 0.1667h^3\lambda^3\) and \(1 + 0.2\lambda h\). Therefore, the absolute stability region of equation (8) is \(\left|1 + h\lambda + 0.78h^2\lambda^2 + 0.1667h^3\lambda^3\right| \leq 1 \cap \left|1 + 0.2\lambda h\right| \leq 1\).

4. The Application of Parallel 4th order Runge-Kutta method

Taking the orbit prediction of a low-orbit satellite as an example, the integral step \(h\) was taken as 0.1 and 0.01 respectively, and the orbit was calculated on the Sugon CB65-G HPC cluster with the parallel Runge-Kutta equation (8) and the conventional serial 4th order Runge-Kutta method (6), the prediction time is 24h (86400s). Under the constraint of non-perturbed motion, according to the law of universal gravitation, for the \(x\) direction orbit calculation, the motion equation is as follows:

\[
\begin{aligned}
x' &= v_x \\
v_x &= \frac{-GMx}{(\sqrt{x^2 + y^2 + z^2})^3}
\end{aligned}
\]

(11)

The equations for \(y\) and \(z\) directions are similar.

The prediction results are shown in the table 1 (Runge-Kutta method abbreviated to RK, \(h\) is the integral step):

| Results of orbit prediction(24hours) | X/km     | Y/km     | Z/km     | VX/km·s⁻¹ | VY/km·s⁻¹ | VZ/km·s⁻¹ |
|-------------------------------------|----------|----------|----------|-----------|-----------|-----------|
| Initial Orbit                      | 9902.3403| 161.3662 | 0.0000   | 2.9712    | 2.7891    | 5.6743    |
| STK prediction                      | 9940.7866| 5071.2895| 10465.175| 1.8964    | 1.8590    | 3.6558    |
| Serial 4th order RK, h=0.1         | 9947.6119| 5065.4309| 10453.6831| 1.8926   | 1.8609    | 3.6596    |
| Serial 4th order RK, h=0.01        | 9951.7136| 5061.9008| 10446.7573| 1.8904   | 1.8620    | 3.6619    |
| Parallel 4th order RK, h=0.01      | 9942.1357| 5070.1344| 10462.9104| 1.8957   | 1.8594    | 3.6566    |
| Parallel 4th order RK, h=0.01      | 9943.5055| 5068.9589| 10460.6043| 1.8949   | 1.8598    | 3.6573    |

It can be seen from Table 1 that the accuracy of the parallel 4th order Runge-Kutta method is slightly lower than that of the serial 4th order Runge-Kutta method under the long integral step. But the result of parallel fourth-order Runge-Kutta method is close to the serial Runge-Kutta method with the decrease of integral step. Therefore, a reasonable selection of integral steps is the key factor that the parallel 4th order Runge-Kutta method used in the actual computer numerical simulation.

| Simulation Type            | Calculation Time (ms, h=0.1) | Calculation Time (ms, h=0.01) |
|---------------------------|------------------------------|------------------------------|
| Serial 4th order RK       | 0.00661                      | 0.06607                      |
| Parallel 4th order RK     | 0.00342                      | 0.03452                      |

It can be seen from Table 2 that the calculation time of the parallel 4th order Runge-Kutta method is less than that of serial 4th order Runge-Kutta method. When the integral step is 0.1, the calculation time of parallel method is about half of the latter. With the decrease of the integral step, the parallel method and the serial method are getting close slightly, which is caused by the delay of data exchange between clusters nodes in the parallel method. Therefore, the parallel depth of the parallel method
should not be too high in practical applications. According to experiments on high-performance servers, it is better to control the parallel depth within 5 times.

5. Conclusion
In this manuscript, a parallel Runge-Kutta method is proposed. In this method, $n$ order Runge-Kutta method can be parallelized into an anterior $n-1$ order Runge-Kutta method and a followed $n$ order Runge-Kutta method, and it can be further parallel to $n-m$ order Runge-Kutta method. The parallel equation can be used to realize the rapid solution of ODE in a cluster with $m+1$ processor. The experiments results show that the accuracy of the parallel 4th order Runge-Kutta method is higher than the accuracy of the serial 3rd order Runge-Kutta method. In the case of small integral step, the accuracy of the parallel 4th order Runge-Kutta method can meet the accuracy requirements of serial 4th order Runge-Kutta method approximately. The calculation time of parallel 4th order Runge-Kutta method is almost half of that of serial 4th order Runge-Kutta method in proper parallel depth.

References
[1] Hairer, E. and Wanner, G. (2006) Solving Ordinary Differential Equations II, Stiff and Differential-Algebraic Problems, Springer-Verlag New York
[2] Oh, H.(2005) SRAK Methods for Mixed DAEs of Index 2 and 3 and Their Application in Mechanics. PhD thesis, University of Iowa
[3] Potzold, L. (1982) Differential/algebraic equations are not ODEs[J]. SIAM Journal of scientific statistical computing,3:145-151
[4] Hairer, E., Lubich, C.and Wanner, G.(2006) Geometric Numerical Integration, Springer-Verlag Berlin Heidelberg
[5] Hairer, E., Norsett, S.P. and Wanner, G.(2000) Solving Ordinary Differential Equations I Nonstiff Problems, Springer New York
[6] Jay, L.O. (2006) Specialized Runge-Kutta methods for index 2 differential-algebraic equations[J]. Mathematics of Computation, 75:641-654
[7] Jay, L.O. (2007) Specialized partitioned additive Runge-Kutta methods for systems of overdetermined DAEs with holonomic constraints[J],SIAM Journal on Numerical Analysis,45:1814-1842
[8] Miranker, W.L and Liniger, W.M. (1967) Parallel Methods for Numerical Integration of Ordinary Differential Equations[J], Math Compute., 21:303-320
[9] Fei, J. (1993)A new multi-step parallel Runge-Kutta equation[J], Applied mathematics, 6:411-416
[10] Bleszynski, E., Bleszynski, M., and Jaroszewicz, T. (1996) AIM: adaptive integral method for solving large-scale electromagnetic scattering and radiation problems[J], Radio Science, 31: 1225-1251
[11] Guo, H., Hu,J. and Nie,Z.(2014)An MPI-OpenMP Hybrid Parallel H-LU Direct Solver for Electromagnetic Integral Equations[J],International Journal of Antennas and Propagation 2015:1-10
[12] Enenkel, R. F. and Jackson, K. R.(1997)DIMSEMs-diagonally implicit single-eigenvalue methods for the numerical solution of stiff ODEs on parallel computers[J],Advances in Computational Mathematics,7:97-133
[13] Michielssen, E. and Boag, A. (1996)A Multilevel matrix decomposition algorithm for analyzing scattering from large structures[J],IEEETransactions on Antennas and Propagation,44:1086-1093
[14] Rius J.M., Parron J., Ubeda E. and Mosig J.R. (1999) Multilevel ‘matrix decomposition algorithm for analysis of electrically large electromagnetic problems in 3-D[J], Microwave and Optical Technology Letters,22: 177-182
[15] Guo, H., Hu, J.and Michielssen, E.(2013) On MLMDA/butterfly compressibility of inverse integral operators[J], IEEE Antennas and Wireless Propagation Letters,12:31-34