Regular Paper

Parallelizable Block Rosenbrock Methods for Linear Variable-coefficient System of ODEs

Nobuyuki Esaki† and Taketomo Mitsui††

In the previous paper (Esaki and Mitsui, 2001), we proposed parallelizable ROW-type discretization methods to apply to a linear variable-coefficient system of ODEs. They showed good performance on the parallel computing system, but, since the maximum order cannot exceed three, they are considered not to be much practical. In the present paper, we develop a generalized implicit Runge-Kutta method and its block upper-triangular form, named as a block Rosenbrock method, and derive a parallelizable one. Order analysis, global convergence and stability analysis are carried out for the fourth order scheme of the new method. Numerical experiments show its practicality under a parallel computer environment by comparing other conventional methods.

1. Introduction

We are concerned with numerical solutions of the initial value problem (IVP) of d-dimensional linear variable-coefficient system of ordinary differential equations (ODEs) given by

$$\boldsymbol{y}' = \boldsymbol{f}(t, \boldsymbol{y}) = \boldsymbol{L}(t)\boldsymbol{y} + \boldsymbol{F}(t), \quad (1)$$
$$\boldsymbol{y}(0) = \boldsymbol{y}_0.$$

Here the coefficient matrix $\boldsymbol{L}(t)$ is assumed to be huge, sparse and stiff. The problem often occurs in any spatial discretization method for partial differential equations of evolution type possessing time-varying coefficients. A typical example can be seen in the semi-discretization method along the characteristics for advection-diffusion equations [6),13)]. Taking the property of sparseness and stiffness of the equations into account, we are required to develop a parallelizable numerical integration method coping with these difficulties. However, to our knowledge, studies of this direction are still rare. A Magnus-expansion method (e.g., Refs. 2), 3)) has been proposed for linear variable-coefficient case. Also a similar way to apply a matrix commutator to highly oscillatory problems was proposed [9)]. Nevertheless they are far from a practical application because of a complicated formula manipulation of the solution form. Henceforth, in the present paper, we will focus on the topic developing a parallelizable discrete variable method for the linear variable-coefficient system Eq. (1).

† Toyota National College of Technology
†† Graduate School of Information Science, Nagoya University

An IVP of ODEs with general right-hand side function $\boldsymbol{f}$ can be solved numerically by means of an implicit Runge-Kutta (IRK) method which is expressed with the following scheme [8),11)].

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_y + h\sum_{i=1}^{s} \beta_i \boldsymbol{k}_i, \quad (2)$$

$$\boldsymbol{k}_i = \boldsymbol{f}\left(t_n + \gamma_i h, \boldsymbol{y}_n + h\sum_{j=1}^{s} \alpha_{ij}\boldsymbol{k}_j\right), \quad (3)$$

for $i = 1, 2, \cdots, s$. Note that the schemes Eqs. (2) and (3) are often associated with its Butcher tableau

$$\begin{array}{c|c} \boldsymbol{c} & \boldsymbol{A} \\ \hline & \boldsymbol{b}^T \end{array},$$

where $\boldsymbol{A}, \boldsymbol{b}$ and $\boldsymbol{c}$ stand for s-square matrix (α_{ij}), s-dimensional vectors (β_i) and (γ_i), respectively. To cope with stiffness of the system, the IRK schemes must be A- or AN-stable [8)]. In this case, however, the method costs much in view of its computational time, even though the stepsize h is not constrained due to the stability. This mainly comes from the cost in solving the (nonlinear) Eq. (3) with respect to the stage-values $\{\boldsymbol{k}_i\}$ at every time-step t_n. For a linear system of ODEs, the problem turns out to solve a bigger system of simultaneous linear equations at t_n.

To overcome this difficulty we proposed [5)] parallelizable ROW-type methods, by referring to the parallelizable IRK methods [7),10)] for constant-coefficient linear systems. For readers' convenience, we give a summary of our methods [5)] here. An application of IRK to Eq. (1) reduces its stage Eq. (3) into a huge and

sparse linear system given by Eq. (55). To parallelize Eq. (55), we approximate all of the matrices $\boldsymbol{L}(t_n+\gamma_i h)$ for $i=1,2,\cdots,s$ in the left-hand side of the equation by the same $\boldsymbol{L}_C = \boldsymbol{L}(t_n+Ch)$, whereas its right-hand side remains as in Eq. (55). The approximated equation can be written as

$$(\boldsymbol{I}-h\boldsymbol{A}\otimes\boldsymbol{L}_C)\boldsymbol{K}=\boldsymbol{\Phi} \tag{4}$$

by introducing the Kronecker product $\otimes$ of matrices. Here we select the formula parameters so as to let the matrix $\boldsymbol{A}$ diagonalizable as $\boldsymbol{A}=\boldsymbol{S}^{-1}\boldsymbol{\Lambda}\boldsymbol{S}$, where $\boldsymbol{\Lambda}=\mathrm{diag}(\lambda_i)$. Then the Kronecker product of $\boldsymbol{A}$ and $\boldsymbol{L}_C$ can be expressed as

$$\begin{aligned}&\boldsymbol{A}\otimes\boldsymbol{L}_C\\&=(\boldsymbol{S}^{-1}\otimes\boldsymbol{I})(\boldsymbol{\Lambda}\otimes\boldsymbol{L}_C)(\boldsymbol{S}\otimes\boldsymbol{I}).\end{aligned} \tag{5}$$

Consequently a left-multiplication of $(\boldsymbol{S}\otimes\boldsymbol{I})$ to Eq. (4) yields

$$(\boldsymbol{I}-h(\boldsymbol{\Lambda}\otimes\boldsymbol{L}_C))\boldsymbol{U}=\boldsymbol{V}, \tag{6}$$

which means Eq. (56). Here $\boldsymbol{U}$ and $\boldsymbol{V}$ are the transformed $\boldsymbol{K}$ and $\boldsymbol{\Phi}$. Thus it suffices to solve only d-dimensional linear equations

$$(\boldsymbol{I}-h\lambda_i\boldsymbol{L}_C)\boldsymbol{u}_i=\boldsymbol{v}_i \tag{7}$$

for each i $(i=1,2,\cdots,s)$, and furthermore the computation can be carried out in parallel with an s-CPU parallel computer. An extra work, we have, however, to compute

$$\boldsymbol{K}=(\boldsymbol{S}^{-1}\otimes\boldsymbol{I})\boldsymbol{U}$$

in every temporal step and to assemble them by

$$\boldsymbol{k}_i=\sum_{j=1}^{s}(\boldsymbol{S}^{-1})_{ij}\boldsymbol{u}_j,$$

for $i=1,2,\cdots,s$. To sum up, this approximation reduces the computational time by the reasons

- that the size of the linear system to be solved can be reduced
- and that each solver can perform in parallel.

We derived single-stage second order and two-stage third order schemes of this type, which are parallelizable and possess a good stability. However we found that its highest order cannot exceed three. To compromize the demand of computational cost with the required accuracy, the order of accuracy of the numerical method is expected to be as higher as possible. Therefore, in the present paper we are to propose another approximation of higher oder to Eq. (55) without much loss of parallelism.

2. Block Rosenbrock Methods

Taking the linear variable-coefficient case into account, we generalize IRK methods into the matrix form of the stage equation as Eq. (57). In fact, the case for $C_i=\gamma_i$ $(i=1,2,\cdots,s)$ implies an s-stage IRK, while the case for $C_i=C$ yields our ROW-type methods [5]. To achieve the aim described in the previous section, we decompose the coefficient matrix into b blocks in each of which the parameter C_i is equalized, and take its block-upper triangular form. That is, we consider the stage equation given by Eq. (58). Here $\boldsymbol{A}_{kl}$ is an $(s_k\times s_l)$ matrix for $k,l=1,2,\cdots,b$, $\boldsymbol{L}_{C_k}$ denotes $\boldsymbol{L}(t_n+C_kh)$ $(k=1,2,\cdots,b)$, while $\boldsymbol{K}_k$ and $\boldsymbol{\Phi}_k$ are appropriate block of the stage-value and the right-hand side vectors, respectively. The proposed scheme can be called block Rosenbrock method due to its feature. We will assign a specified block Rosenbrock method by the row $(b,s_1,\cdots,s_b)$ where b stands for the number of blocks and s_i shows the number of dimension of i-th block. In the present paper we devote ourselves to the method of the type $(2,2,2)$. The actual computation proceeds as follows. First, the equation in the second block

$$(\boldsymbol{I}-h\boldsymbol{A}_{22}\otimes\boldsymbol{L}_{C_2})\boldsymbol{K}_2=\boldsymbol{\Phi}_2 \tag{8}$$

is solved with the parallel algorithm similar to that for Eq. (4) by applying a two-CPU parallelism. Second, the equation in the first block

$$\begin{aligned}&(\boldsymbol{I}-h\boldsymbol{A}_{11}\otimes\boldsymbol{L}_{C_1})\boldsymbol{K}_1\\&=\boldsymbol{\Phi}_1+h(\boldsymbol{A}_{12}\otimes\boldsymbol{L}_{C_1})\boldsymbol{K}_2\end{aligned} \tag{9}$$

is solved with the same algorithm as above after the substitution for $\boldsymbol{K}_2$. Finally, the computed stage-values are employed for Eq. (2).

From the computational point of view, when applying a two-CPU parallel computer, the block Rosenbrock method of the type (2,2,2) costs almost twice more than the corresponding parallelizable ROW-type method $(s=4)$ in each temporal step. Considering similarly, the computational cost of a block Rosenbrock method with b blocks is of b times of that of the ROW-type of the same total number of stages, provided that we have a parallel processor which can cover every block at the same cost. This linear growth of costs in block Rosenbrock methods along the number of blocks would not be much serious, for, when the same level of accuracy is demanded for numerical solutions, a higher order method can reduce the number of steps to achieve the pre-assigned accuracy level.

3. Order Conditions and Their Solution

A derivation of the order conditions is carried out by analyzing the generalized IRK method Eq. (57) of s stages. The emerging parameters $\{C_i\}$ in the scheme can be appended to the Butcher tableau of the IRK as

$$\begin{array}{cc|c} \boldsymbol{C} & \boldsymbol{c} & \boldsymbol{A} \\ \hline & & \boldsymbol{b}^T \end{array},$$

where $\boldsymbol{C} = (C_1, \cdots, C_s)^T$.

As is common in the discrete variable methods for ODEs, we expand the exact and numerical solutions, $\boldsymbol{y}(t_{n+1})$ and $\boldsymbol{y}_{n+1}$, respectively, with respect to the stepsize h under the localization assumption $\boldsymbol{y}_n = \boldsymbol{y}(t_n)$. The exact solution at the next step $(t_n + h)$ is expanded as Eq. (59). In this section, for simplicity, we will temporarily denote the functional evaluations at t_n for $\boldsymbol{y}, \boldsymbol{L}, \boldsymbol{F}$ and their derivatives by their function name. That is, $\boldsymbol{y}(t_n)$ by y, $\boldsymbol{L}(t_n)$ by L and so on. Then, by taking into account the linearity of the ODE system, the higher derivatives of $\boldsymbol{y}(t)$ are expressed as

$$\begin{aligned} \boldsymbol{y}'(t_n) &= Ly + F, \\ \boldsymbol{y}''(t_n) &= L(Ly + F) + L'y + F', \\ \boldsymbol{y}'''(t_n) &= L^2(Ly + F) + 2L'(Ly + F) \\ &\quad + L(L'y + F') + L''y + F'', \\ \boldsymbol{y}^{(4)}(t_n) &= L^3(Ly + F) + 5LL'(Ly + F) \\ &\quad + 3L''(Ly + F) + L^2(L'y + F') \\ &\quad + 3L'(L'y + F') + L(L''y + F'') \\ &\quad + L'''y + F'''. \end{aligned}$$

Taking advantage that we are considering a linear variable-coefficient case Eq. (1), we can reduce formula manipulation a bit, but still have to carry out the power series expansion and its substitution repeatedly for the numerical solution $\boldsymbol{y}_{n+1}$ given by Eqs. (2) and (57). As a result, we obtain Eq. (60). By comparing the coefficients of the same powers of h in Eqs. (59) and (60) we can obtain the order conditions of the method. Note that these conditions are valid for the linear variable-coefficient system. At present we are not intending to the general system of ODEs. A detailed description of the analysis of the order conditions is given in Appendix A.1. Here we only mention the derived parameters of the (2,2,2)-type block Rosenbrock method of fourth order in the double precision accuracy.

$$\begin{aligned} \alpha_{11} &= 1.00625 \\ \alpha_{12} &= -0.37638641839513261 \\ \alpha_{13} &= -0.29985410339729551 \\ \alpha_{14} &= 0.0 \\ \alpha_{21} &= 0.49030606531690384 \\ \alpha_{22} &= -0.12016964692177122 \\ \alpha_{23} &= 0.0 \\ \alpha_{24} &= 0.29985410339729551 \\ \alpha_{31} &= 0.0 \\ \alpha_{32} &= 0.0 \\ \alpha_{33} &= 1.01087594700249180 \\ \alpha_{34} &= -0.94144410279951808 \\ \alpha_{41} &= 0.0 \\ \alpha_{42} &= 0.0 \\ \alpha_{43} &= -0.12994816623471965 \\ \alpha_{44} &= 1.06051632203174594 \\ \beta_1 &= 0.32607257743127307 \\ \beta_2 &= 0.32607257743127307 \\ \beta_3 &= 0.17392742256872692 \\ \beta_4 &= 0.17392742256872692 \\ \gamma_1 &= 0.3300094782075718 \\ \gamma_2 &= 0.6699905217924281 \\ \gamma_3 &= 0.0694318442029737 \\ \gamma_4 &= 0.9305681557970262 \\ C_1 &= 0.83881017107725915 \\ C_2 &= 0.83881017107725915 \\ C_3 &= 0.34393851177186564 \\ C_4 &= 0.34393851177186564 \end{aligned}$$

Furthermore, the diagonal blocks $\boldsymbol{A}_{11}, \boldsymbol{A}_{22}$ are made to be diagonalizable through a similarity transformation as $\boldsymbol{A}_{11} = \boldsymbol{S}_1^{-1}\boldsymbol{\Lambda}_1\boldsymbol{S}_1$ and $\boldsymbol{A}_{22} = \boldsymbol{S}_2^{-1}\boldsymbol{\Lambda}_2\boldsymbol{S}_2$. When we denote $\boldsymbol{\Lambda}_i = \mathrm{diag}(\lambda_j^{(i)}), \boldsymbol{S}_i = (s_{j\ell}^{(i)})$ and $\boldsymbol{S}_i^{-1} = (t_{j\ell}^{(i)})$, the components are given as follows:

$$\begin{aligned} \lambda_1^{(1)} &= 0.80726642682978542 \\ \lambda_2^{(1)} &= 0.07881392624844334 \\ s_{11}^{(1)} &= 1.44012843462329139 \\ s_{12}^{(1)} &= -0.58445514346259248 \\ s_{21}^{(1)} &= -0.72639611344244829 \\ s_{22}^{(1)} &= 1.37401106593291927 \\ t_{11}^{(1)} &= 0.88405955099841603 \\ t_{12}^{(1)} &= 0.37604730014123471 \\ t_{21}^{(1)} &= 0.46737427217218432 \\ t_{22}^{(1)} &= 0.92660046840938308 \\ \lambda_1^{(2)} &= 1.38634549852559605 \\ \lambda_2^{(2)} &= 0.68504677050864169 \\ s_{11}^{(2)} &= 0.50019556522965889 \\ s_{12}^{(2)} &= -1.44525475035481424 \end{aligned}$$

$$s_{21}^{(2)} = -0.56655017298169639$$
$$s_{22}^{(2)} = -1.42055545417733843$$
$$t_{11}^{(2)} = 0.92885320219021638$$
$$t_{12}^{(2)} = -0.94500323721970348$$
$$t_{21}^{(2)} = -0.37044801090163920$$
$$t_{22}^{(2)} = -0.32706097542244446$$

4. Convergence Analysis

We will show the global convergence of the proposed block Rosenbrock method. First, we denote a single-step discrete variable method for Eq. (1) with a general right-hand side function $\boldsymbol{f}$ by

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h\Phi(t_n, \boldsymbol{y}_n; h), \tag{10}$$

where the mapping Φ means all the stage computations including $\boldsymbol{k}_i$. Therefore Φ depends on $\boldsymbol{f}$. The local discretization error defined by

$$\boldsymbol{T}_{n+1} = \frac{1}{h}\{\boldsymbol{y}(t_{n+1}) - \boldsymbol{y}(t_n) - h\Phi(t_n, \boldsymbol{y}(t_n), h)\}$$

or

$$\boldsymbol{y}(t_{n+1}) = \boldsymbol{y}(t_n) + h\Phi(t_n, \boldsymbol{y}(t_n); h) + h\boldsymbol{T}_{n+1} \tag{11}$$

turns out to be

$$\boldsymbol{T}_{n+1} = \boldsymbol{A}(t_n, \boldsymbol{y}(t_n))h^p + O(h^{p+1}), \tag{12}$$

for a p-th order method. The coefficient $\boldsymbol{A}(t_n, \boldsymbol{y}(t_n))$ generally becomes a complicated combination of many higher order derivatives of $\boldsymbol{f}$. When $\boldsymbol{f}$ is sufficiently continuously differentiable, we can assume that for a certain positive h_0 there is a positive $\bar{A}$ satisfying the inequality

$$\| \boldsymbol{T}_{n+1} \| \leq \bar{A}h^p \quad (h \leq h_0). \tag{13}$$

Under the condition, the global error $\boldsymbol{e}_{n+1}$ defined by the difference $\boldsymbol{y}(t_{n+1})$ and $\boldsymbol{y}_{n+1}$ has the recurrence formula

$$\boldsymbol{e}_{n+1} = \boldsymbol{e}_n + h\{\Phi(t_n, \boldsymbol{y}(t_n); h) - \Phi(t_n, \boldsymbol{y}_n; h)\} + h\boldsymbol{T}_{n+1} \tag{14}$$

due to Eqs. (11) and (10).

Here we concentrate ourselves to the case of (2,2,2)-type block Rosenbrock method ($p = 4$) for the linear variable-coefficient system Eq. (1). Then, when we assume the four times continuous differentiability for $\boldsymbol{L}(t)$ and $\boldsymbol{F}(t)$, the conditions above mentioned hold, and furthermore we have a Lipschitz constant $\overline{L}$ satisfying $\| \boldsymbol{L}(t)\boldsymbol{y} - \boldsymbol{L}(t)\boldsymbol{y}^* \| \leq \overline{L} \| \boldsymbol{y} - \boldsymbol{y}^* \|$. We assume that the stage equation of the generalized IRK Eq. (57) at t_n is exactly solved to give

$$\boldsymbol{k}_i = h\sum_{j=1}^{s} \alpha_{ij}\boldsymbol{L}(t_n + C_i h)\boldsymbol{k}_j + \boldsymbol{L}(t_n + \gamma_i h)\boldsymbol{y}_n + \boldsymbol{F}(t_n + \gamma_i h).$$

Denoting the solution of the stage equation with $\boldsymbol{y}(t_n)$ in place of $\boldsymbol{y}_n$ by $\boldsymbol{k}_i^*$, that is,

$$\boldsymbol{k}_i^* = h\sum_{j=1}^{s} \alpha_{ij}\boldsymbol{L}(t_n + C_i h)\boldsymbol{k}_j^* + \boldsymbol{L}(t_n + \gamma_i h)\boldsymbol{y}(t_n) + \boldsymbol{F}(t_n + \gamma_i h),$$

we have the inequality

$$\| \boldsymbol{k}_i - \boldsymbol{k}_i^* \| \leq \overline{L} \| \boldsymbol{y}(t_n) - \boldsymbol{y}_n \| + h\overline{L}\sum_{j=1}^{s} |\alpha_{ij}| \| \boldsymbol{k}_j - \boldsymbol{k}_j^* \| .$$

As expressed in Eq. (2), the increment function Φ depends on $\boldsymbol{k}_i$'s linearly, when the simultaneous stage equation (57) is solved exactly. Thus solution of the above simultaneous inequalities with respect to $\| \boldsymbol{k}_i - \boldsymbol{k}_i^* \|$ $(i = 1, 2, \cdots, s)$ yields

$$\| \Phi(t_n, \boldsymbol{y}(t_n); h) - \Phi(t_n, \boldsymbol{y}_n; h) \| \leq M \| \boldsymbol{e}_n \| \tag{15}$$

with a certain positive constant M. Consequently, together with the inequality (13), we can obtain the recurrence relationship

$$\| \boldsymbol{e}_{n+1} \| \leq (1 + hM) \| \boldsymbol{e}_n \| + \bar{A}h^{p+1},$$

which derives

$$\| \boldsymbol{e}_n \| \leq \frac{\bar{A}h^p}{M}((1 + hM)^n - 1),$$

for $n = 0, 1, 2, \cdots$, as $\boldsymbol{e}_0 = 0$. Eventually, for the (2,2,2)-type block Rosenbrock method, we obtain

$$\| \boldsymbol{e}_n \| \leq Ch^4 \tag{16}$$

with a constant C independent of h or n. Therefore when h tends to 0, the global error converges to 0.

5. Stability Analysis

The numerical stability is crucial in our block Rosenbrock methods, for we target a linear huge and stiff system. As shown below, their A-stability is rather easily established, whereas a stability suitable to the linear variable-coefficient case can hold under a certain restriction of the size of the stepsize h. In the literature a parallel concept of A-stability is called AN-stability (A-stability for Nonautonomous system, see Refs. 4), 8)). This means that we cannot expect AN-stability of our methods. As we are interested in their application with a sufficiently small stepsize h, a confirmation of such a stepsize for the new sta-

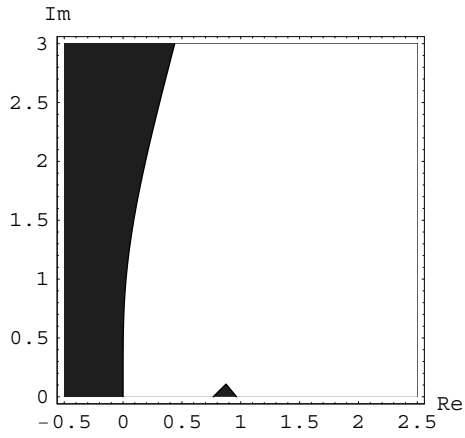

Fig. 1 Region of absolute stability.

bility should be, however, useful.

5.1 *A*-stability

Since A-stability is a necessary condition of AN-stability, we first analyze the former of the block Rosenbrock methods.

When the scalar test equation

$$y' = \lambda y \ (t \geq 0), \quad \lambda \in \mathbb{C}, \quad y(0) = y_0 \quad (17)$$

is solved by the block Rosenbrock method, its absolute stability factor can be given by

$$R(z) = \frac{\det(\boldsymbol{I} - z\boldsymbol{A} + z\boldsymbol{e}\boldsymbol{b}^T)}{\det(\boldsymbol{I} - z\boldsymbol{A})}, \quad (18)$$

where $z = h\lambda$ ($z \in \mathbb{C}$) and $\boldsymbol{e}$ denotes the s-vector all of whose components are unity. It is right as same as that of IRK, for we are now taking the linear test Eq. (17). Therefore, defining A-stability of the method by the criterion

$$|R(z)| \leq 1 \quad \text{for any } z \in \mathbb{C} \text{ satisfying } \Re z \leq 0, \quad (19)$$

we choose the formula parameters of (2,2,2)-type method so that the criterion attains. Then, as described in Appendix A.1, the remained freedom of α_{11} can serve this aim. Our analysis shows that the condition $1.00613647058398071 \leq \alpha_{11}$ is sufficient for A-stability, and we selected $\alpha_{11} = 1.00625$. Actually, under this selection, the region of absolute stability $\{z \in \mathbb{C}; |R(z)| \leq 1\}$ is approximately displayed in **Fig. 1**.

Figure 1 only shows the region of absolute stability in the upper half plane of $\mathbb{C}$ as the shadowed picture, for the region should be symmetric with respect to the real axis. Since the region includes the whole left half plane, the method is A-stable.

5.2 *NA*-stability

While A-stability analysis is based on the autonomous linear equation, it is not sufficient for the variable-coefficient linear system Eq. (1). The concept of NA-stability tries to bridge the gap by taking a scalar, linear and nonautonomous (variable) equation

$$y'(t) = \lambda(t)y(t) \quad (t \geq 0), \quad \lambda(t) \in \mathbb{C}, \quad y(0) = y_0 \quad (20)$$

as the test equation. Here $\lambda(t)$ is an arbitrary varying complex-valued function. Applying an s-stage generalized IRK method, which includes our block Rosenbrock method, to Eq. (20), we treat different s values of the function $\lambda(t)$ both in the left- and right-hand sides of the stage equation. Consequently, introducing the notations $z_i^{(\ell)} = h\lambda(t_n + C_i h)$ and $z_i^{(r)} = h\lambda(t_n + \gamma_i h)$ ($z_i^{(\ell)}, z_i^{(r)} \in \mathbb{C}$ for $i = 1, 2, \cdots, s$) and the two s-square diagonal matrices $\boldsymbol{Z}_\ell = \mathrm{diag}(z_i^{(\ell)})$ and $\boldsymbol{Z}_r = \mathrm{diag}(z_i^{(r)})$, the NA-stability factor of the generalized IRK can be given by

$$R(\boldsymbol{Z}_\ell, \boldsymbol{Z}_r) = \frac{\det(\boldsymbol{I} - \boldsymbol{Z}_\ell \boldsymbol{A} + \boldsymbol{Z}_r \boldsymbol{e}\boldsymbol{b}^T)}{\det(\boldsymbol{I} - \boldsymbol{Z}_\ell \boldsymbol{A})}. \quad (21)$$

Since the matrices $\boldsymbol{Z}_\ell$ and $\boldsymbol{Z}_r$ are governed with $z_i^{(\ell)}$ and $z_i^{(r)}$, we define the $(2s)$-vector $\boldsymbol{z} = (\boldsymbol{z}_\ell^T, \boldsymbol{z}_r^T)^T$, which relates to the NA-stability factor as $R(\boldsymbol{z}) = R(\boldsymbol{Z}_\ell, \boldsymbol{Z}_r)$.

The best stability criterion should be

$$\mathcal{R} \equiv \{\boldsymbol{z}; |R(\boldsymbol{z})| \leq 1\} \supset \mathbb{C}^{2s,-} \equiv \{\boldsymbol{z}; \Re z_i < 0 \ \forall i\}$$

and, when it holds, we call the method unconditionally NA-stable. This concept coincides with the conventional AN-stability. The set $\mathcal{R}$ will be referred to as the domain of NA-stability of the method. When the domain $\mathcal{R}$ exists but does not include the whole $\mathbb{C}^{2s,-}$, the method is said to be conditionally NA-stable.

Consider our block Rosenbrock methods, then we can say that it is impossible to be unconditionally NA-stable, for the numerator determinant of the factor $R(\boldsymbol{Z}_\ell, \boldsymbol{Z}_r)$ carries $z_i^{(r)}$ which contributes to the factor monotonically increasing its magnitude when $|z_i^{(r)}|$ tends to infinity. Thus we must turn our study to a criterion of conditional NA-stability. It is still hard to carry out completely due to the number of freedom in $\boldsymbol{z}$ which is generally of $(2s)$-dimension.

To establish a conditional NA-stability, we

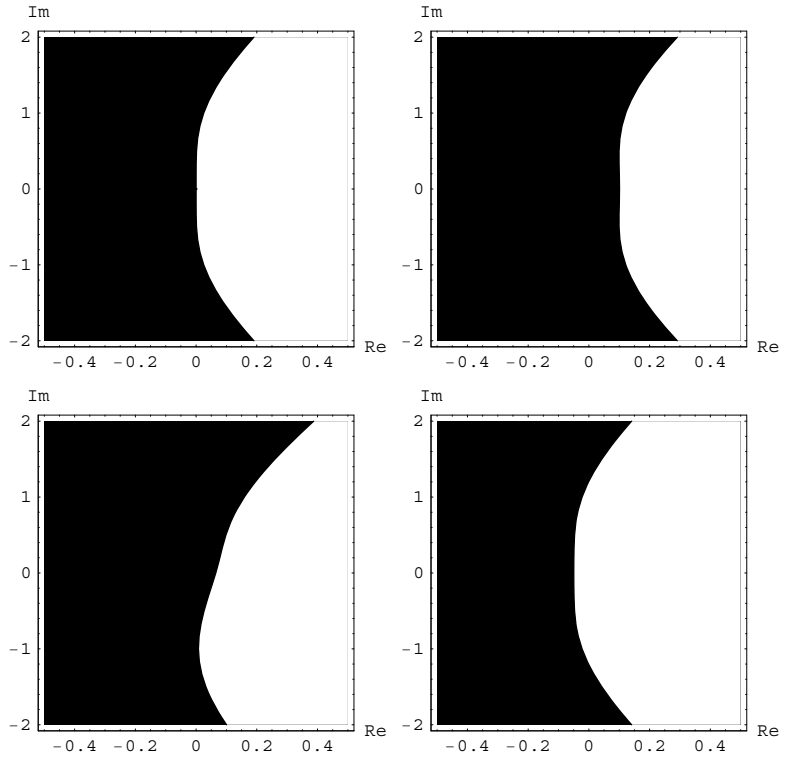

Fig. 2 Regions of conditional NA-stability of the (2,2,2)-type block Rosenbrock method.
top-left: $z_0' = 0$, top-right: $z_0' = -0.2$
bottom-left: $z_0' = -0.1 + 0.5\mathrm{i}$, bottom-right: $z_0' = 0.1$

assume a sufficiently small stepsize h and employ the Taylor series expansion of $\lambda(t_n + \gamma_i h)$ or $\lambda(t_n + C_i h)$ appearing in the scheme of block Rosenbrock methods with respect to h. As the symbol c_i stands for either C_i or γ_i, we have

$$z_i = h\lambda(t_n + c_i h) \approx h\lambda(t_n) + c_i h^2 \lambda'(t_n)$$

for every i. Introducing the new complex parameters $z_0 = h\lambda(t_n)$ and $z_0' = h^2\lambda'(t_n)$, and, with a specified formula parameters, substituting the above second order approximation into all $z_i^{(\ell)}$ and $z_i^{(r)}$, we obtain the approximate NA-stability factor $\hat{R}(z_0, z_0')$ satisfying

$$\hat{R}(z_0, z_0') = R(\boldsymbol{z}) + O(h^3). \tag{22}$$

Note that $\hat{R}(z, 0)$ coincides with the corresponding absolute stability factor $R(z)$ in Eq. (18). We may say therefore $\hat{R}(z_0, z_0')$ is a second order approximation of the NA-stability factor with two parameters for a small h.

After a formula manipulation by computer for the NA-stability factor of the (2,2,2)-type block Rosenbrock method, we derive its approximant $\hat{R}(z_0, z_0')$ and plot figures of the region

$$\{z_0 \in \mathbb{C};\ |\hat{R}(z_0, z_0')| \le 1\}$$

for several fixed value of z_0'. The results are shown in **Fig. 2**. Actually the figures for $z_0' = 0, -0.2, -0.1 + 0.5\mathrm{i}$ and 0.1 are displayed.

From the figure, we can observe as follows. When $z_0' = 0$, the top-left of Fig. 2 just coincides with Fig. 1 as expected. When z_0' is a negative real number, as shown at the top-right Fig. 2, the region includes the whole left half plane. When z_0' has non-zero imaginary and negative real parts, as shown at the bottom-left of Fig. 2, the region which is no longer symmetric with respect to the real line tends to shrink as its imaginary part increases in magnitude, but still includes the whole left half plane. When z_0' has positive real part, as shown at the bottom-right of Fig. 2, the region no longer includes the whole left half plane. Furthermore, a tedious calculation in each case except the bottom-right can derive that the NA-stability factor is less than unity in magnitude on the imaginary axis and that it has no singularity in the left half-plane. Therefore we can say the

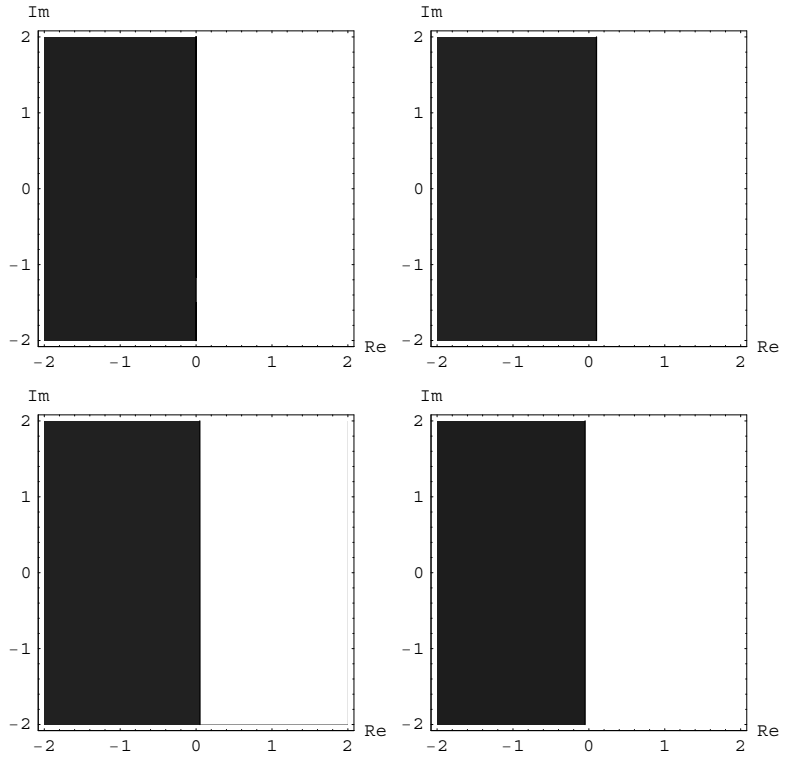

Fig. 3 Regions of conditional NA-stability of BK24.
top-left: $z_0' = 0$, top-right: $z_0' = -0.2$
bottom-left: $z_0' = -0.1 + 0.5\mathrm{i}$, bottom-right: $z_0' = 0.1$

(2,2,2)-type block Rosenbrock method is conditionally NA-stable.

To confirm our way of analysis, we carried out the conditional NA-stability analysis for the two-stage fourth order Butcher-Kuntzmann method [8]. The results are displayed in **Fig. 3**.

From the figure, we can say that for every combination of z_0 and z_0' satisfying $\Re z_0 < 0$ and $\Re z_0' < 0$ the region of conditional NA-stability includes the left half plane, whereas the inclusion breaks down for $\Re z_0' > 0$. These observations coincide with the fact that BK24 is unconditionally NA-stable, that is, AN-stable.

Consequently our way of NA-stability analysis can be reasonable. Finally our (2,2,2)-type block Rosenbrock method is NA-stable for sufficiently small stepsize h. Unfortunately a quantitative knowledge of the upper bound of NA-stable stepsize is quite hard.

6. Numerical Tests

In the present section we will show the results of numerical experiments by the (2,2,2)-type block Rosenbrock method for a model problem.

6.1 Convergence Behavior

To check the convergence of our method, we carry out numerical tests with several varying stepsizes. We employ the following d-dimensional ODEs possessing the exact solution $\boldsymbol{y}(t) = \boldsymbol{g}(t)$.

$$\begin{aligned} \frac{\mathrm{d}\boldsymbol{y}(t)}{\mathrm{d}t} &= \boldsymbol{L}(t)\boldsymbol{y}(t) + \boldsymbol{g}'(t) - \boldsymbol{L}(t)\boldsymbol{g}(t), \quad (23) \\ \boldsymbol{y}(0) &= \boldsymbol{g}(0). \end{aligned}$$

As an example, we take $\boldsymbol{L}(t)$ and $\boldsymbol{g}(t)$ as follows:

$$\boldsymbol{L}(t) = \mathrm{tridiag}\left(1 - \frac{1}{2}\sin t, 1, 1 - \frac{1}{2}\cos t\right),$$

$$\boldsymbol{g}(t) = \mathrm{e}^{-2t}\begin{bmatrix} 1 \\ 2 \\ \vdots \\ d \end{bmatrix}.$$

We take the system Eq. (23) suggested by the scalar model equation of Prothero-Robinson [12]. Although it is still an artificial one, the system is linear variable-coefficient and has its stiffness ratio varying between some 10^2 and 10^6 in our numerical calculations. The latter ratio appears when $d = 200$ and $t = 0.875$. Our target system described in 6) has its stiffness ratio of 10^6.

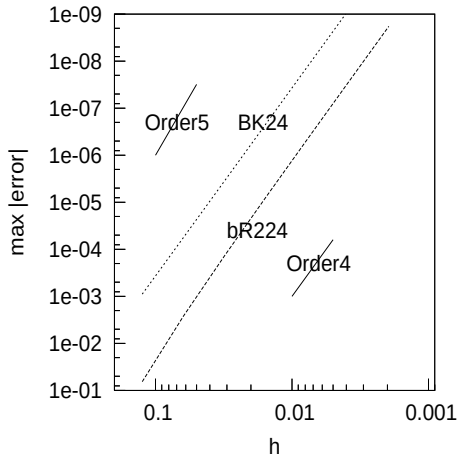

Fig. 4 Maximum absolute errors at $t = 1.0$ versus the stepsize h.

Thus we can consider Eq. (23) as a test bed towards a real-life application.

We solve the above system numerically by two methods, *i.e.,* our (2,2,2)-type block Rosenbrock method (abbreviated as bR224) and the two-stage fourth order Butcher-Kuntzmann method (abbreviated as BK24), which is known as one of the conventional AN-stable IRK methods. For bR224 we apply the parallelized algorithm, but only on the sequential architecture for the convergence test. The convergence behavior is given in **Fig. 4**.

In Fig. 4, the symbols Order4 and Order5 are inserted to refer to the slopes corresponding to the convergence order $O(h^4)$ and $O(h^5)$, respectively. The result of bR224 exhibits fourth order of convergence, expectedly, with the varying temporal stepsize h, while that of BK24, which exhibits the same order, is shown for comparison. Moreover, for other t's and d's we had essentially the same results.

6.2 Computational Performance on Parallel Processors

Next to check the computational performance of the parallelized algorithm, we employ a 4-CPU parallel cluster computing environment, which is composed of two 2-CPU computers, VT-Alpha6 500DP2, connected by LAN of 100 Mbps. The CPU clock is declared to be Alpha21264 500 MHz. The SDRAM of the CPUs is operated in the distribution mode. MIMD programs on Tru64 C-compiler are realized with the MPICH of version 1.1, which is one of the Message-Passing Interface (MPI) packages, on the computer that performs as a distributed parallel machine. For the communications between processors we employ the library function calls, MPI_Isend() and MPI_Irecv(), which perform as immediate (nonblocking) point-to-point communication. The default two-CPU mode is set to run with one cluster. However, when we employ the full clusters for the two-CPU operation by allocating two jobs to each cluster separately, computational experiments revealed little difference in the computing time, provided the network traffic was not so busy. Henceforth we employ the default mode in the actual computations.

We numerically solve the aforementioned test problem up to $t = 1.0$ by the parallelized block Rosenbrock method on the one and the two CPU architectures. For comparison, we also solve the test problem on the one CPU architecture by the two-stage fourth order Butcher-Kuntzmann scheme, and also by the two-stage third order ROW-type scheme [5], denoted by ROW23, on the one and the two CPU architectures. As K. Burrage[1] describes, any proposed parallel algorithm should be compared with the most efficient serial codes. Thus we take both BK24 and ROW23 as the conventional methods of good stability and convergence, and try to measure the total performance of our methods by comparing with them.

Table 1 shows the statistics of the elapsed CPU time in seconds, which derives the speedup ratio as **Table 2**. Note that the CPU time statistics is measured only for the derivation of the speedup ratio. It may vary when the computational conditions are changed. All the algorithms are programmed so that the matrix $\boldsymbol{L}(t)$ can be solved as a full one. The way of measurement is as follows. Set the problem dimension as 200, and measure the CPU time of the algorithms when they gave the same level of the maximum absolute error. This of course means that the required numbers of steps for the integration interval might be different between the algorithms, as denoted in Appendix A.2. The level of the maximum absolute error varies as $10^{-3}, 10^{-4}, 10^{-5}$ and 10^{-6}. Their performance is compared on these measurement. Same process is repeated for the problem of dimension 400.

The left-hand side in the second column of Table 2 shows the speedup ratio of the parallelized bR224, while the right-hand side that of ROW23. That is, the ratio of the elapsed CPU time of its 2-CPU implementation ver-

Table 1 Elapsed CPU time (sec).

d	max |error|	bR224 1-CPU (1)	bR224 2-CPU (2)	BK24 1-CPU (3)	ROW23 1-CPU (4)	ROW23 2-CPU (5)
200	1.0E-3	2.00	1.14	1.20	1.49	0.76
200	1.0E-4	3.20	1.85	2.40	3.26	1.70
200	1.0E-5	5.59	3.22	4.20	6.75	3.53
200	1.0E-6	9.97	5.74	8.27	14.35	7.48
400	1.0E-3	13.26	7.33	10.08	10.23	5.20
400	1.0E-4	21.16	11.66	20.21	22.38	11.48
400	1.0E-5	37.06	20.37	35.31	46.35	23.75
400	1.0E-6	66.15	36.35	69.26	98.46	59.38

Table 2 Speedup for elapsed CPU time.

d	max |error|	(1)/(2)	(4)/(5)	(3)/(2)	(5)/(2)	(3)/(5)
200	1.0E-3	1.75	1.96	1.05	0.66	1.58
200	1.0E-4	1.72	1.91	1.29	0.91	1.41
200	1.0E-5	1.73	1.91	1.30	1.09	1.19
200	1.0E-6	1.73	1.92	1.44	1.30	1.11
400	1.0E-3	1.80	1.96	1.37	0.70	1.93
400	1.0E-4	1.81	1.94	1.73	0.98	1.76
400	1.0E-5	1.81	1.95	1.73	1.16	1.49
400	1.0E-6	1.81	1.95	1.90	1.63	1.37

sus that of 1-CPU case. Since it attains 1.72–1.75 for bR224 and 1.91–1.96 for ROW23, the parallelization is realized properly although the speedup ratio of bR224 is inferior to that of ROW23.

The speedup ratio in the last column of Table 2 exhibits the advantage of the bR224 comparing with the BK24 and the ROW23. The right-hand side in the column shows results in our previous paper [5], which implies an advantage of the parallelized ROW23 over BK24, when the matrix $\boldsymbol{L}(t)$ becomes large in the size but we do not need higher accuracy. While, the left-hand side in the column shows the advantage of parallelized bR224 over sequential BK24, especially the case of larger d, both of which achieve the same convergence behavior of fourth order. Next as mentioned in section 2, the parallel computational cost of bR224 is about twice more than ROW23 in the same number of steps, however, it is expected to compensate the loss due to its faster convergence. The middle in the column indicates the advantage of bR224 over ROW23 when accuracy level becomes higher. This means that the higher order accuracy may overcome its less parallelism. Summing up, we can composed a new descretization scheme, which achieves fourth order but does not loose the parallelism much.

Next we rearrange the algorithms to take advantage of the sparseness of the coefficient matrix $\boldsymbol{L}(t)$ of the equation. This enables the algorithm of both bR224 and ROW23 to solve d-dimensional *tridiagonal* linear equations on each step. (We employed the Thomas algorithm.) However, BK24 cannot enjoy the tridiagonality and has to solve a linear system with *banded matrix* of band-width $(d+1)$. The rearrangement of the program of the linear solver much reduces the computing time in all the cases. Due to the Thomas algorithm the computing time of both bR224 and ROW23 drastically decreases, and this effect becomes more when the number of the dimension increases. This stresses the advantage of our block Rosenbrock methods as well. The result is shown in **Table 3**. By comparing the statistics for bR224 and ROW23 in the 2-CPU mode in Table 3, in all cases bR224 is faster than ROW23 in spite of its less speedup ratio.

7. Concluding Remarks

We proposed a parallelizable block Rosenbrock methods to solve the time-dependent linear stiff system of ODEs. An actually derived totally four-stage scheme achieves fourth order, be computable in two blocks, parallelizable in each block with two CPUs, and can be considered practical. The method also possesses a good NA-stability and its parallel implementation on a cluster computing environment shows that it gives better performance than the conventional methods BK24 and ROW23. We

Table 3 CPU time (sec) when taking advantage of the sparseness.

d	max |error|	bR224 1-CPU	bR224 2-CPU	BK24 1-CPU	ROW23 1-CPU	ROW23 2-CPU
200	1.0E-6	0.19	0.20	3.57	0.31	0.26
400	1.0E-6	0.38	0.38	27.25	0.63	0.49
800	1.0E-6	0.75	0.63	293.80	1.26	0.80
1600	1.0E-6	1.52	1.26	3342.18	2.51	1.57

also proposed a new way for conditional NA-stability analysis, which is efficient for a sufficiently small stepsize h.

Acknowledgments The present authors express their gratitude for the useful comments as well as the constructive criticism by the anonymous reviewers.

References

1) Burrage, K.: *Parallel and Sequential Methods for Ordinary Differential Equations*, Oxford University Press, Oxford (1995).
2) Casas, F.: Solution of linear partial differential equations by Lie algebraic methods, *J. Comput. Appl. Math.*, Vol.76, pp.159–170 (1996).
3) Casas, F.: Numerical integration methods for the double-bracket flow, *J. Comput. Appl. Math.,* Vol.166, pp.477–495 (2004).
4) Dekker, K. and Verwer, J.G.: *Stability of Runge-Kutta Methods for Stiff Nonlinear Differential Equations*, North-Holland, Amsterdam (1984).
5) Esaki, N. and Mitsui, T.: Parallelizable ROW methods for linear variable-coefficient system of ODEs, *IPSJ Trans. HPS.* (2001).
6) Esaki, N., Mitsui, T., Oliveira, F.A. and Patrício, F.: Semi-discretization method along the characteristics for linear advection-diffusion equations, *SEA Bull. Math.*, Vol.27, pp.813–827 (2004).
7) Goto, A. and Mitsui, T.: A 3-stage fourth order RK formula for parallel computers, *Trans. JSIAM*, Vol.1, No.4, pp.291–304 (1991) (in Japanese).
8) Hairer, E. and Wanner, G., *Solving Ordinary Differential Equations II, Stiff and Differential-Algebraic Systems*, 2nd revised ed., Springer-V. (1996).
9) Iserles, A.: Solving linear ordinary differential equations by exponentials of iterated commutators, *Numer. Math.,* Vol.45, pp.183–199 (1984).
10) Karakashian, O.A. and Rust, W.: On the parallel implementation of implicit Runge-Kutta methods, *SIAM J. Sci. Stat. Comput.*, Vol.9, pp.1085–1090 (1988).
11) Mitsui, T.: *Introduction to Numerical Analysis — With Emphasis on Ordinary Differential Equations*, Asakura Publ., Tokyo (1985) (in Japanese).
12) Prothero, A. and Rosinson, A.: On the stabilty and accuracy of one-step methods for solving stiff systems of ordinary differential equations, *Math. Comp.*, Vol.28, pp.145–162 (1974).
13) Vreugdenhil, C.B. and Koren, B. (ed.): *Numerical Methods for Advection-Diffusion Problems*, Vieweg, Braunschweig, Germany (1993).

Appendix

A.1 Analysis of the Order Conditions

First we take the conventional assumption

$$\gamma_i = \sum_{j=1}^{s} \alpha_{ij} \qquad \text{for} \quad i = 1, 2, \cdots, s \tag{24}$$

which, common in Runge-Kutta-type of methods, reduces the complexity of the equations.

The conditions for fourth order accuracy are as follows.

$$\sum_{i=1}^{s} \beta_i = 1 \tag{25}$$

$$\sum_{i=1}^{s} \beta_i \gamma_i = \frac{1}{2} \tag{26}$$

$$\frac{1}{2} \sum_{i=1}^{s} \beta_i \gamma_i^2 = \frac{1}{6} \tag{27}$$

$$\sum_{i=1}^{s} \beta_i \sum_{j=1}^{s} \alpha_{ij} \gamma_j = \frac{1}{6} \tag{28}$$

$$\sum_{i=1}^{s} \beta_i C_i \gamma_i = \frac{2}{6} \tag{29}$$

$$\frac{1}{6} \sum_{i=1}^{s} \beta_i \gamma_i^3 = \frac{1}{24} \tag{30}$$

$$\frac{1}{2} \sum_{i=1}^{s} \beta_i \sum_{j=1}^{s} \alpha_{ij} \gamma_j^2 = \frac{1}{24} \tag{31}$$

$$\sum_{i=1}^{s} \beta_i C_i \sum_{j=1}^{s} \alpha_{ij} \gamma_j = \frac{3}{24} \tag{32}$$

$$\sum_{i=1}^{s} \beta_i \sum_{j=1}^{s} \alpha_{ij} \sum_{l=1}^{s} \alpha_{jl} \gamma_l = \frac{1}{24} \tag{33}$$

$$\frac{1}{2}\sum_{i=1}^{s}\beta_i C_i^2\gamma_i = \frac{3}{24} \quad (34)$$

$$\sum_{i=1}^{s}\beta_i\sum_{j=1}^{s}\alpha_{ij}C_j\gamma_j = \frac{2}{24} \quad (35)$$

Hereafter we concentrate ourselves to the (2,2,2)-type block Rosenbrock scheme which attains fourth order. That is, we set $a_{31} = a_{32} = a_{41} = a_{42} = 0$, $C_1 = C_2$ and $C_3 = C_4$. Next we allocate for $\{\gamma_i\}$ the roots of the shifted Legendre polynomial of degree four. Thus we have

$$\gamma_1 = \frac{1}{70}\left(35 - \sqrt{35(15-2\sqrt{30})}\right) \quad (36)$$

$$\gamma_2 = \frac{1}{70}\left(35 + \sqrt{35(15-2\sqrt{30})}\right) \quad (37)$$

$$\gamma_3 = \frac{1}{70}\left(35 - \sqrt{35(15+2\sqrt{30})}\right) \quad (38)$$

$$\gamma_4 = \frac{1}{70}\left(35 + \sqrt{35(15+2\sqrt{30})}\right). \quad (39)$$

Substitution of these into Eqs. (25), (26), (27) and (30) determines $\{\beta_i\}$'s uniquely, however their exact algebraic expression is too much complicated with many rooting symbols. Instead we show these parameter solutions in the double precision accuracy.

$$\beta_1 = \beta_2 \simeq 0.32607257743127307 \quad (40)$$

$$\beta_3 = \beta_4 \simeq 0.17392742256872692. \quad (41)$$

Specification of these parameters in Eqs. (29) and (34) implies quadratic equations for C_i's, whose solution is given either pair of

$$\begin{cases} C_1 = 0.83881017107725915, \\ C_3 = 0.34393851177186564 \end{cases} \quad (42)$$

or

$$\begin{cases} C_1 = 0.49452316225607418, \\ C_3 = 0.98939482156146769. \end{cases} \quad (43)$$

We select the former Eq. (42).

The remained parameters $\{\alpha_{ij}\}$ are the solution of Eqs. (28), (31), (32), (33), (35) and (24), however the process is a little complicated. Hence we give its outline. First, we eliminate 4 unknowns through Eq. (24) as

$$\alpha_{12} = \gamma_1 - \alpha_{11} - \alpha_{13} - \alpha_{14} \quad (44)$$

$$\alpha_{21} = \gamma_2 - \alpha_{22} - \alpha_{23} - \alpha_{24} \quad (45)$$

$$\alpha_{34} = \gamma_3 - \alpha_{33} \quad (46)$$

$$\alpha_{43} = \gamma_4 - \alpha_{44}. \quad (47)$$

Further elimination is possible through Eqs. (27), (30), (31) and (35), and we obtain

$$\alpha_{13} = -0.29985410339729551 - \alpha_{23} \quad (48)$$

$$\alpha_{24} = 0.29985410339729551 - \alpha_{14}. \quad (49)$$

Finally Eqs. (28), (31) and (33) derive

$$\alpha_{22} = -1.12641964692177122 + \alpha_{11} + \alpha_{14} - \alpha_{23} \quad (50)$$

$$\alpha_{33} = -3.08467944220248923 + 4.07011715697389420\alpha_{11} - 3.11953127651679558\alpha_{14} - 7.18964843349068978\alpha_{23} \quad (51)$$

$$\alpha_{44} = -3.03503906717323509 + 4.07011715697389420\alpha_{11} - 3.11953127651679558\alpha_{14} - 7.18964843349068978\alpha_{23}. \quad (52)$$

Remaining α_{11}, α_{14} and α_{23} as free parameters, we arrive at the solution of the desired order conditions. The freedom of three parameters are employed to achieve both the diagonalizability and the stability of the scheme. The diagonalizability criteria of the matrices $\boldsymbol{A}_{11}$ and $\boldsymbol{A}_{22}$ are

$$(\alpha_{11} - \alpha_{22})^2 + 4\alpha_{12}\alpha_{21} > 0 \quad (53)$$

$$(\alpha_{33} - \alpha_{44})^2 + 4\alpha_{34}\alpha_{43} > 0, \quad (54)$$

respectively. Here we select $\alpha_{14} = 0$ and $\alpha_{23} = 0$ for the simplicity. Only remained freedom for α_{11} serves to achieve the above criteria. Then we have the condition $\alpha_{11} \leq 0.77513030203711903$ or $0.97413593153469024 \leq \alpha_{11}$. As described in Section 5, the value $\alpha_{11} = 1.00625$ attains A-stability as well as the conditional NA-stability. All the fixed value of the scheme parameters is given at the last paragraph of Section 3.

A.2 Actual Total Steps to Achieve the Accuracy

For readers' convenience we show the total steps of each scheme versus the required accuracy in **Table 4**.

Table 4 Total steps versus the accuracy.

d	max |error|	bR224	BK24	ROW23
200	1.0E-3	16	4	32
200	1.0E-4	32	8	69
200	1.0E-5	54	24	145
200	1.0E-6	62	44	307
400	1.0E-3	16	4	32
400	1.0E-4	32	8	69
400	1.0E-5	54	24	145
400	1.0E-6	107	44	307
200	1.0E-6	107	44	307
400	1.0E-6	107	44	307
800	1.0E-6	107	44	307
1600	1.0E-6	107	44	307

A.3 Equations

$$\begin{bmatrix} \boldsymbol{I}-h\alpha_{11}\boldsymbol{L}(t_n+\gamma_1 h) & -h\alpha_{12}\boldsymbol{L}(t_n+\gamma_1 h) & \cdots & -h\alpha_{1s}\boldsymbol{L}(t_n+\gamma_1 h) \\ -h\alpha_{21}\boldsymbol{L}(t_n+\gamma_2 h) & \boldsymbol{I}-h\alpha_{22}\boldsymbol{L}(t_n+\gamma_2 h) & \cdots & -h\alpha_{2s}\boldsymbol{L}(t_n+\gamma_2 h) \\ \vdots & \vdots & \ddots & \vdots \\ -h\alpha_{s1}\boldsymbol{L}(t_n+\gamma_s h) & -h\alpha_{s2}\boldsymbol{L}(t_n+\gamma_s h) & \cdots & \boldsymbol{I}-h\alpha_{ss}\boldsymbol{L}(t_n+\gamma_s h) \end{bmatrix} \begin{bmatrix} \boldsymbol{k}_1 \\ \boldsymbol{k}_2 \\ \vdots \\ \boldsymbol{k}_s \end{bmatrix}$$
$$= \begin{bmatrix} \boldsymbol{L}(t_n+\gamma_1 h)\boldsymbol{y}_n + \boldsymbol{F}(t_n+\gamma_1 h) \\ \boldsymbol{L}(t_n+\gamma_2 h)\boldsymbol{y}_n + \boldsymbol{F}(t_n+\gamma_2 h) \\ \vdots \\ \boldsymbol{L}(t_n+\gamma_s h)\boldsymbol{y}_n + \boldsymbol{F}(t_n+\gamma_s h) \end{bmatrix}. \tag{55}$$

$$\begin{bmatrix} \boldsymbol{I}-h\lambda_1\boldsymbol{L}_C & & & 0 \\ & \boldsymbol{I}-h\lambda_2\boldsymbol{L}_C & & \\ & & \ddots & \\ 0 & & & \boldsymbol{I}-h\lambda_s\boldsymbol{L}_C \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \\ \vdots \\ \boldsymbol{u}_s \end{bmatrix} = \begin{bmatrix} \boldsymbol{v}_1 \\ \boldsymbol{v}_2 \\ \vdots \\ \boldsymbol{v}_s \end{bmatrix}. \tag{56}$$

$$\begin{bmatrix} \boldsymbol{I}-h\alpha_{11}\boldsymbol{L}(t_n+C_1 h) & -h\alpha_{12}\boldsymbol{L}(t_n+C_1 h) & \cdots & -h\alpha_{1s}\boldsymbol{L}(t_n+C_1 h) \\ -h\alpha_{21}\boldsymbol{L}(t_n+C_2 h) & \boldsymbol{I}-h\alpha_{22}\boldsymbol{L}(t_n+C_2 h) & \cdots & -h\alpha_{2s}\boldsymbol{L}(t_n+C_2 h) \\ \vdots & \vdots & \ddots & \vdots \\ -h\alpha_{s1}\boldsymbol{L}(t_n+C_s h) & -h\alpha_{s2}\boldsymbol{L}(t_n+C_s h) & \cdots & \boldsymbol{I}-h\alpha_{ss}\boldsymbol{L}(t_n+C_s h) \end{bmatrix} \begin{bmatrix} \boldsymbol{k}_1 \\ \boldsymbol{k}_2 \\ \vdots \\ \boldsymbol{k}_s \end{bmatrix}$$
$$= \begin{bmatrix} \boldsymbol{L}(t_n+\gamma_1 h)\boldsymbol{y}_n + \boldsymbol{F}(t_n+\gamma_1 h) \\ \boldsymbol{L}(t_n+\gamma_2 h)\boldsymbol{y}_n + \boldsymbol{F}(t_n+\gamma_2 h) \\ \vdots \\ \boldsymbol{L}(t_n+\gamma_s h)\boldsymbol{y}_n + \boldsymbol{F}(t_n+\gamma_s h) \end{bmatrix}. \tag{57}$$

$$\begin{bmatrix} \boldsymbol{I}-h\boldsymbol{A}_{11}\otimes\boldsymbol{L}_{C_1} & -h\boldsymbol{A}_{12}\otimes\boldsymbol{L}_{C_1} & \cdots & -h\boldsymbol{A}_{1b}\otimes\boldsymbol{L}_{C_1} \\ & \boldsymbol{I}-h\boldsymbol{A}_{22}\otimes\boldsymbol{L}_{C_2} & \cdots & -h\boldsymbol{A}_{2b}\otimes\boldsymbol{L}_{C_2} \\ & & \ddots & \vdots \\ 0 & & & \boldsymbol{I}-h\boldsymbol{A}_{bb}\otimes\boldsymbol{L}_{C_b} \end{bmatrix} \begin{bmatrix} \boldsymbol{K}_1 \\ \boldsymbol{K}_2 \\ \vdots \\ \boldsymbol{K}_b \end{bmatrix} = \begin{bmatrix} \boldsymbol{\Phi}_1 \\ \boldsymbol{\Phi}_2 \\ \vdots \\ \boldsymbol{\Phi}_b \end{bmatrix}. \tag{58}$$

$$\boldsymbol{y}(t_n+h) = \boldsymbol{y}(t_n) + h\boldsymbol{y}'(t_n) + \frac{h^2}{2}\boldsymbol{y}''(t_n) + \frac{h^3}{6}\boldsymbol{y}'''(t_n) + \frac{h^4}{24}\boldsymbol{y}^{(4)}(t_n) + O(h^5). \tag{59}$$

$$\begin{aligned} \boldsymbol{y}_{n+1} = \boldsymbol{y}_n &+ h\sum_{i=1}^{s}\beta_i(Ly+F) \\ &+ h^2\left\{\sum_{i=1}^{s}\beta_i\gamma_i(L'y+F') + \sum_{i=1}^{s}\beta_i\sum_{j=1}^{s}\alpha_{ij}L(Ly+F)\right\} \\ &+ h^3\left\{\frac{1}{2}\sum_{i=1}^{s}\beta_i\gamma_i^2(L''y+F'') + \sum_{i=1}^{s}\beta_i\sum_{j=1}^{s}\alpha_{ij}\gamma_j L(L'y+F')\right. \\ &\quad \left. + \sum_{i=1}^{s}\beta_i C_i\sum_{j=1}^{s}\alpha_{ij}L'(Ly+F) + \sum_{i=1}^{s}\beta_i\sum_{j=1}^{s}\alpha_{ij}\sum_{l=1}^{s}\alpha_{jl}L^2(Ly+F)\right\} \\ &+ h^4\left\{\frac{1}{6}\sum_{i=1}^{s}\beta_i\gamma_i^3(L'''y+F''') + \frac{1}{2}\sum_{i=1}^{s}\beta_i\sum_{j=1}^{s}\alpha_{ij}\gamma_j^2 L(L''y+F'')\right. \\ &\quad \left. + \sum_{i=1}^{s}\beta_i C_i\sum_{j=1}^{s}\alpha_{ij}\gamma_j L'(L'y+F') + \sum_{i=1}^{s}\beta_i\sum_{j=1}^{s}\alpha_{ij}\sum_{l=1}^{s}\alpha_{jl}\gamma_l L^2(L'y+F')\right. \end{aligned}$$

$$+\frac{1}{2}\sum_{i=1}^{s}\beta_i C_i^2\sum_{j=1}^{s}\alpha_{ij}L''(Ly+F)$$
$$+\left(\sum_{i=1}^{s}\beta_i C_i\sum_{j=1}^{s}\alpha_{ij}\sum_{l=1}^{s}\alpha_{jl}+\sum_{i=1}^{s}\beta_i\sum_{j=1}^{s}\alpha_{ij}C_j\sum_{l=1}^{s}\alpha_{jl}\right)LL'(Ly+F)$$
$$+\sum_{i=1}^{s}\beta_i\sum_{j=1}^{s}\alpha_{ij}\sum_{l=1}^{s}\alpha_{jl}\sum_{m=1}^{s}\alpha_{lm}L^3(Ly+F)\Bigg\}+O(h^5). \tag{60}$$

(Received February 1, 2004)
(Accepted July 4, 2004)

Nobuyuki Esaki was born in 1972. He received his M.A. degree from Nagoya Univ. in 1997. He has been working in Toyota College of Technology since 1998 and now is a lecturer. His current research interests are numerical analysis of differential equations and its parallelization. He is a member of the IPS of Japan and Japan Society for Industrial and Applied Mathematics (JSIAM).

Taketomo Mitsui is currently a professor of Department of Computer Science and Mathematical Informatics, Graduate School of Information Science, Nagoya University. His research interests include numerical solutions of differential equations for mathematical modelling and their parallel implementations on computer. He received the Dr. Sci. degree from Kyoto University in 1981. He is a member of the IPS of Japan, Japan Society for Industrial and Applied Mathematics (JSIAM), SIAM (USA) and Mathematical Society of Japan (MSJ).