Iterated Runge-Kutta Methods with Parallelization Capability for Stiff Problems

I. Bondar, B. Faleichik

In this work we describe a class of numerical algorithms for the solution of stiff ODE systems which arise from specific implementation of implicit Runge-Kutta (RK) methods using the principle of steadying. These algorithms do not require Jacobian evaluations and factorizations which is beneficial when the dimension of the ODE is large, and can be naturally parallelized. It is shown that the resulting computational scheme is equivalent to some explicit RK method of first order, but it converges to the solution which corresponds to the base implicit method of high order. Numerical comparison with explicit DOP853 solver is given.

Introduction

The most of this article is based on the material from our recent paper [1]. Here we expanded the numerical experiment section which includes the results of parallelization on 2-core processor.

Consider a system of ordinary differential equations(ODE) of dimension n

$$y' = f(t, y(t)), \quad y(t_0) = y_0, \tag{1}$$

where $f : \mathbb{R}^{n+1} \to \mathbb{R}^n$, f is continuously differentiable function; in the linear case

$$f(t, y(t)) = Jy(t) + g(t), \quad J \in \mathbb{R}^{n\times n}. \tag{2}$$

Take some time step τ and apply the implicit s-stage Runge-Kutta method (base method) to the problem (1):

$$y(t_0 + \tau) \approx y_1 = y_0 + \tau(b^T \otimes I)F(t_0, Y), \tag{3}$$

where $F(t, Y) = (f(t + c_1\tau, Y_1), \ldots, f(t + c_s\tau, Y_s))^T$, I is identity matrix, $Y = (Y_1, \ldots, Y_s)^T$, $Y \in R^{ns}$ is the solution of the following system of equations

$$Y = Y_0 + \tau(A \otimes I)F(t_0, Y). \tag{4}$$

Here $A = (a_{ij})_{i=1}^s$, $b = (b_i)_{i=1}^s$, $c = (c_i)_{i=1}^s$ are the coefficients of the base implicit method, and

$$Y_0 = \underline{1}_s \otimes y_0,$$

$\underline{1}_s = (1, \ldots, 1)^T \in R^s$. As a rule, (4) is solved by Newton's method or its modifications. However, this approach requires significant computational cost of the Jacobi matrix factorization in the case of large dimensions. This paper presents an alternative method based on the idea of steadying.

Consider a system of (non-)linear equations

$$r(X) = 0; \quad X \in R^N, \quad r : R^N \to R^N, \tag{5}$$

where r is continuously differentiable function. Let X^* be its exact solution. Consider the process

$$X' = r(X). \tag{6}$$

The derivative here is taken with respect to the additional variable – the fictitious time. It is known that steady-state solution of (6) is an exact solution to (5). If we apply some explicit σ-stage Runge-Kutta method (*an auxiliary method*) to (6) then we get a family of iterative processes of the following form:

$$X^{[k+1]} = \Phi(X^{[k]}) = X^{[k]} + \omega\beta^T r(W^{[k]}), \tag{7}$$

$$W^{[k]} = X^{[k]} + \omega\mathcal{A}r(W^{[k-1]}), \tag{8}$$

where β, $\mathcal{A}$ are the coefficients of an auxiliary method:

$$\left[\begin{array}{c|c} \mathcal{A} \\ \hline \beta \end{array}\right] = \left[\begin{array}{ccccc} 0 & & & & \\ \alpha_{21} & 0 & & & \\ \alpha_{31} & \alpha_{32} & 0 & & \\ \vdots & \vdots & \vdots & & \\ \alpha_{\sigma 1} & \alpha_{\sigma 2} & \dots & \alpha_{\sigma\sigma-1} & 0 \\ \hline \beta_1 & \beta_2 & \dots & \beta_{\sigma-1} & \beta_\sigma \end{array}\right]. \tag{9}$$

In our case (4) the natural choice of the residual function is

$$r(Y) = Y_0 - Y + \tau(A \otimes I)F(t_0, Y). \tag{10}$$

As a result, we obtain a class of methods which we call Generalized Picard iterations (GPI):

$$Y^{[k+1]} = Y^{[k]} + \omega \sum_{p=1}^{\sigma} \beta_p r(Y^{[k,p]}), \quad k = 1, \dots, m-1 \tag{11}$$

$$Y^{[k,p]} = Y^{[k]} + \omega \sum_{q=1}^{p-1} \alpha_{pq} r(Y^{[k,q]}), \quad p = 1, \dots, \sigma \tag{12}$$

$$y_1 = y_0 + \tau(b^T \otimes I)F(t_0, Y^{[m]}). \tag{13}$$

One way to select the coefficients of auxiliary method is described in [2]. In particular, using the fact that the process (12) - (12) in the linear case can be written as

$$Y^{[k+1]} = R_\sigma(\omega G)Y^{[k]} + P(\omega, G), \tag{14}$$

where R_σ is a polynomial of degree σ, which is called the stability polynomial of the auxiliary method, G is the matrix of the resulting system, $G = \tau A \otimes J - I$ (see (2)), P is some matrix polynomial function.

The polynomial R_σ determines the stability properties of the ODE integration method. In our case, it determines the convergence properties of the iterative process. In particular, the stability domain

$$S = \{z \in C \,:\, |R_\sigma(z)| < 1\}$$

must contain the spectrum of ωG [2].

Let us write the stability polynomial in the form

$$R_\sigma(z) = 1 + \sum_{j=1}^{\sigma} a_j z^j,$$

where $z \in \mathbb{C}$. The coefficients $\{a_i\}_{i=1}^{\sigma}$, $a_i \in \mathbb{R}$ are chosen to minimize the functional

$$F(a_1, \ldots, a_\sigma) = \int_0^1 \int_{\pi-\alpha}^{\pi+\alpha} |R_\sigma(\rho e^{\mathrm{i}\varphi})|^2 d\varphi d\rho. \tag{15}$$

Here α is some predetermined angle that defines the area of stability of the polynomial R_σ [3], [4]. To achieve faster convergence we need R_σ to take as small absolute values as possible over its stability region. Therefore, selection of α should be based on the concepts of the spectrum of the matrix G, because the better the approximate range, the more effective is the method. The parameter ω is chosen so that the spectrum of ωG is completely contained in the stability domain. It's enough to put ω equal to the inverse of the spectral radius of G.

Having constructed the stability polynomial we can restore the tableau of auxiliary method using the approach described in [5], [6].

If the initial condition $Y^{[0]}$ is Y_0, we can prove the following statement:

Proposition 1. *If in* (10) $Y^{[0]} = Y_0$, *then for all of auxiliary explicit methods type* (9) *and for all* $m \geq 0$ *generalized Picard iteration equivalent to* $s(m\sigma + 1)$ *– stage Runge -Kutta method with the table*

$$\begin{array}{c|c} c^{[m]} & A^{[m]} \\ \hline & \left(b^{[m]}\right)^T \end{array} = \begin{array}{c|c} 1_{m\sigma+1} \otimes & \omega \left(I + \omega \mathcal{A}\right)^{-1} \mathcal{A}^{[m]} \left(I_{m\sigma+1} \otimes A\right) \\ \hline & \left[0, 0, \ldots, b^T\right] \end{array}, \tag{16}$$

where

$$\mathcal{A}^{[m]} = \begin{bmatrix} \mathcal{A} \otimes I_s & & & & & \\ \mathcal{B} & \mathcal{A} \otimes I_s & & & & \\ \mathcal{B} & \mathcal{B} & \mathcal{A} \otimes I_s & & & \\ \vdots & \vdots & \ddots & \ddots & & \\ \mathcal{B} & \mathcal{B} & \ldots & \mathcal{B} & \mathcal{A} \otimes I_s & \\ \beta^T \otimes I_s & \beta^T \otimes I_s & \ldots & \beta^T \otimes I_s & \beta^T \otimes I_s & O_s \end{bmatrix}, \tag{17}$$

$\mathcal{B} = (\mathbf{1}_\sigma \beta^T) \otimes I_s$, *$I_s$ and O_s - identity and zero matrix $s \times s$ respectively.*

Proposition 2. *Let R be the stability polynomial of auxiliary method,* $\deg R = \sigma$. *Then the polynomial stability of the resulting method* (16) *is*

$$R^{[m]}(z) = 1 + zb^T \left(I + \sum_{k=1}^{m\sigma} z^k C_k^{[m]} \right) \mathbf{1}_s, \tag{18}$$

$$C_k^{[m]} = A^k - \sum_{i=1}^{k-1} B_i^{[m]} A^{ki},$$

where $B_i^{[m]}$ coefficients of the power series expansion

$$\left(R\left(\omega(zA - I)\right)\right)^m = \sum_{i=0}^{m\sigma} z^i B_i^{[m]}. \tag{19}$$

Implementation

There are several important points when implementing implicit Runge-Kutta methods with the use of GPI.

Firstly, it should be noted that during the computation it is better to use "shifted" variable $Z = Y - Y_0$, then

$$r = r(Z) = -Z + \tau(A \otimes I)F(t_0, Y_0 + Z).$$

Additionally, if the matrix Butcher of basic method is not singular, it is possible to carry out the so-called "preconditioning". In this case,

$$r(Z) = -(A^{-1} \otimes I)Z + \tau F(t_0, Y_0 + Z),$$

often accelerates the convergence of iterative processes.

The error estimate is the second important point. Since we do not build LU-decomposition of the Jacobi matrix, we can not use the method described in [5] for estimating the error. Therefore, in order to estimate the error, we will solve the problem with two implicit collocation method with the various orders. Then, if A, b, c, s and $\tilde{A}$, $\tilde{b}$, $\tilde{c}$, $\tilde{s}$ – parameters of the respective methods(their orders let p and $\tilde{p}$, respectively, $p > \tilde{p}$), we obtain two systems of nonlinear equations for the implementation:

$$Y = Y_0 + \tau(A \otimes I)F(t_0, Y) \tag{20}$$

$$\tilde{Y} = Y_0 + \tau(\tilde{A} \otimes I)F(t_0, \tilde{Y}) \tag{21}$$

For definiteness, we assume that $s > \tilde{s}$. After solving of these systems it is easy to get an estimate of the error.

To speed up the solution of (21) use the fact that the method in (20) is a collocation one. This means that there exists a polynomial P of degree s (collocation polynomial) which satisfies the initial condition $y(t_0) = y_0$ and the equation at some point ξ_i:

$$P'(\xi_i) = f(\xi_i, P(\xi_i)), \tag{22}$$

$\xi_i = t_0 + c_i\tau$. Furthermore, since the $Y^* = (Y_1^*, \ldots, Y_s^*)^T$ is the solution of (20) which is already known, and provided that $Y_i^* \approx y(\xi_i), i = \overline{1,s}$, these values may be interpolated using Lagrangian polynomial of the form

$$P(t) = \sum_{i=1}^{s} Y_i \prod_{j \neq i} \frac{t - (t_0 + c_j\tau)}{(c_i - c_j)\tau} \tag{23}$$

Similarly, for (21),

$$\tilde{Y}_i^* \approx y(\tilde{\xi}_i), i = \overline{1,s},$$

$\tilde{\xi}_i = t_0 + \tilde{c}_i\tau$. The polynomial $P(t)$ interpolates $y(t)$ on the interval $[t_0, t_0 + \tau]$ and in the points $\xi_i, \quad i = \overline{1s}$ is almost identical to that function. It is supposed that the polynomial (23) is good enough approximate $y(t)$ and in points $\tilde{\xi}_i, \quad i = \overline{1,\tilde{s}}$. It is reasonable to use the vector $\tilde{Y}^0 = (\tilde{Y}_1 = 0, \ldots, \tilde{Y}_{\tilde{s}}^0)$ for an initial approximation to the solution of the equation (21) ($Y_i^0, \quad i = \overline{1,\tilde{s}}$ are the values of the polynomial $P(t)$ at points $\tilde{\xi}_i = t_0 + \tilde{c}_i\tau$):

$$\tilde{Y}_k^0 = P(\tilde{\xi}_k), \quad k = \overline{1,s} \tag{24}$$

In addition, computing $r(Y)$ by the formula (10) can be naturally parallelized in the linear and nonlinear cases (in general, the number of threads equals to the number of stages of the implicit method).

Numerical experiment

For the experiment we took the test problem BRUSS-2D from [5] without any modification (the dimension of ODE is 32768, the diffusion coefficient is 1, so the problem is stiff). We compared our GPI code with the explicit method of Dormand-Prince (DOP853). Such a comparison is reasonable, since both methods belong to the same class and are useful for the tasks of moderate stiffness. In the GPI codes the evaluation of residual function r was parallelized using OpenMP. As a basic method we used the 4-stage Radau-IIA method of order 7 and 4-stage Gaussian method of order 8 was used to estimate the error. In the role of the auxiliary method 7-step method optimized with $\alpha = \pi/180$ was applied. The experiment was performed on two machines: the first is Intel Core 2 Duo T54501.66GHz and the second is 4-core Intel Core 2 Quad Q6600 2.4 GHz processor. Both machines run Linux operating system. The results of the experiment are presented on Figure **??**. Further details of the implementation are described in [1].

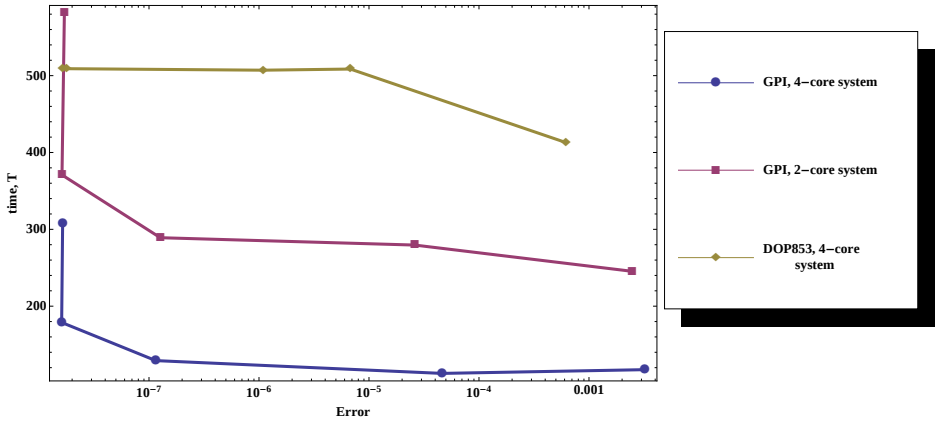

Figure 1. Convergence diagram for Generalized Picard Iterations (GPI) and DOP853 on BRUSS-2D problem. The required tolerance Atol is 10^{-k}, $k = 2, 4, 6, 8, 10$. Acceleration for GPI is more than two times (≈ 2.5 for $Atol = 10^{-4}$) due to differences in processor test machines.

Conclusion

The paper presented approach to the implementation of implicit Runge-Kutta method based on the principle of the steading. A distinctive feature of the method is its simplicity and uniformness. In addition, Generalized Picard Iteration not require storage and factorization of matrix therefore it is applicable in case of large-scale systems.

Using as a base calculation method, we can get rid of a large amount of unnecessary calculations when estimating the error.

References

[1] B. Faleichik, I. Bondar, and V. Byl., "Generalized picard iterations: a class of iterated runge–kutta methods for stiff problems," *To appear in Journal of Computational and Applied Mathematics*, 2013.

[2] B. Faleichik and I. Bondar, "Implementation of implicit methods for stiff problems by the method of steadying," in *Proceedings of the International Scientific Conference of Students and Young Scientists "Theoretical and Applied Aspects of Cybernetics"*, pp. 297–299, Bukrek Kyiv, 2001.

[3] B. Faleichik, "Nonparametric estimation of intensities of nonhomogeneous Poisson processes," *Journal of Numerical Analysis, Industrial and Applied Mathematics*, vol. 5, no. 1-2, pp. 49–59, 2010.

[4] B. Faleichik and I. Bondar, "Implementation of implicit runge-kutta method using the principle of steadying," in *Proceedings Reports of the International Conference "Analytical methods of analysis and differential equations"*, pp. 146–147, Minsk, 2011.

[5] E. Hairer and G. Wanner, *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*. Springer-Verlag, 1996.

[6] B. Faleichik, "Implementation of implicit methods for stiff problems using generalized picard iterations," in *Proceedings of the '6th International Conference "Analytical Methods of Analysis and Differential Equations" : in two volumes - Volume 1 Mathematical analysis*, pp. 131–135, Institute of Mathematics, National Academy of Sciences, 2012.

Authors

Ivan Bondar — Master of Science, Department of Computational mathematics, Belarusian State University, Minsk, Belarus ; E-mail: *bondarIV@bsu.by*

Boris Faleichik — Candidate of physical and mathematical sciences, Department of Computational mathematics, Belarusian State University, Minsk, Belarus ; E-mail: *faleichik@bsu.by*