Parallel PC iteration of pseudo two-step RK methods for nonstiff IVPs *

Nguyen Huu Cong † and Taketomo Mitsui ††

† *Faculty of Mathematics, Mechanics and Informatics, Hanoi University of Science*
334 Nguyen Trai, Thanh Xuan, Hanoi, Vietnam
†† *Graduate School of Human Informatics, Nagoya University*
Furo-cho, Chikusa-ku, Nagoya 464-8601, Japan

Abstract

A parallel PC iteration scheme for a general class of pseudo two-step Runge-Kutta methods (PTRK methods) of arbitrarily high order is analyzed for solving first-order nonstiff initial-value problems (IVPs) on parallel computers. Starting with an s-stage pseudo two-step RK method of order p^* with w implicit stages, we apply the highly parallel PC iteration process in P(EC)^{m}E mode. The resulting parallel-iterated pseudo two-step RK method (PIPTRK method) uses an optimal number of processors equal to w. By a number of numerical experiments, we show the superiority of the PIPTRK methods proposed in this paper over both sequential and parallel methods available in the literature.

Key words: RK methods, PC methods, parallelism.
AMS(MOS) subject classifications (1991): 65M12, 65M20
CR subject classifications: G.1.7

1 Introduction

The arrival of parallel computers influences the development of methods for the numerical solution of a nonstiff initial value problem (IVP) for systems of first-order ordinary differential equations (ODEs)

$$\boldsymbol{y}'(t) = \boldsymbol{f}(t, \boldsymbol{y}(t)), \qquad \boldsymbol{y}(t_0) = \boldsymbol{y}_0, \qquad t_0 \le t \le T, \tag{1.1}$$

where $\boldsymbol{y}$, $\boldsymbol{f} \in \mathbb{R}^d$. The most efficient numerical methods for solving this problem are the explicit Runge-Kutta methods (RK methods). In the literature, sequential explicit RK methods up to order 10 can be found in *e.g.,* [15, 17, 18]. In order to exploit the facilities of parallel computers, a number of parallel explicit methods have been investigated in *e.g.,* [1, 2, 3, 4, 6, 7, 8, 9, 12, 13, 14, 19, 20, 21]. A common challenge in the latter mentioned works is to reduce, for a given order of accuracy, the required number of effective sequential $\boldsymbol{f}$-evaluations per step, using parallel processors. In the present paper, we propose a class of

*This work was supported by Scientist Exchange Program FY2001 of JSPS

parallel PC iteration methods based on a new class of pseudo two-step RK corrector methods (PTRK corrector methods) recently proposed in [10]. A new s-stage PTRK corrector method using w implicit stages, $v = s - w$ explicit stages is of step point and stage order both at least equal to s with any integer pair w, v, with $w + v = s$. It is always zero-stable and can attain the step point order $p^* = s + 1$, stage order $q^* = s$ (see Section 2). Applying a highly parallel PC iteration scheme to PTRK methods gives us the methods which are analogous to the parallel-iterated RK methods (PIRK methods) proposed in [20]. Therefore, the resulting PC iteration methods will be termed *parallel-iterated PTRK methods* (PIPTRK methods) (see Section 3).

Although, for a given number of processors, the order of the PIPTRK methods used for the numerical experiments in this paper is equal to that of the PIRK methods, their rate of convergence is better, their predictor formula is more accurate, so that their efficiency is expected to be increased (see Section 4). The increased efficiency is demonstrated in Sections 4.1 and 4.2, where numerical results are presented by comparing the PIPTRK methods with PIRK and sequential explicit RK methods available in the literature.

In the following sections, for the sake of simplicity of notation, we assume that the IVP (1.1) is a scalar problem. However, all considerations below can be straightforwardly extended to a system of ODEs.

2 PTRK corrector methods

The PTRK corrector methods were firstly proposed in [10], following the similar approach used in [11]. In this section, we give an overview on these methods. Let collocation vector $\boldsymbol{c}$ be partitioned into two subvectors $\boldsymbol{c}_v$ and $\boldsymbol{c}_w$ that is $\boldsymbol{c} = (\boldsymbol{c}_v^T, \boldsymbol{c}_w^T)^T$, then a general s-stage PTRK method based on $\boldsymbol{c}$ for a scalar equation is defined by

$$\boldsymbol{V}_n = u_n \boldsymbol{e}_v + hA_{vv} f(t_{n-1}\boldsymbol{e}_v + h\boldsymbol{c}_v, \boldsymbol{V}_{n-1}) + hA_{vw} f(t_{n-1}\boldsymbol{e}_w + h\boldsymbol{c}_w, \boldsymbol{W}_{n-1}), \tag{2.1a}$$

$$\boldsymbol{W}_n = u_n \boldsymbol{e}_w + hA_{wv} f(t_n \boldsymbol{e}_v + h\boldsymbol{c}_v, \boldsymbol{V}_n) + hA_{ww} f(t_n \boldsymbol{e}_w + h\boldsymbol{c}_w, \boldsymbol{W}_n), \tag{2.1b}$$

$$u_{n+1} = u_n + h\boldsymbol{b}_v^T f(t_n \boldsymbol{e}_v + h\boldsymbol{c}_v, \boldsymbol{V}_n) + h\boldsymbol{b}_w^T f(t_n \boldsymbol{e}_w + h\boldsymbol{c}_w, \boldsymbol{W}_n), \tag{2.1c}$$

where, $u_{n+1} \approx y(t_{n+1})$, A_{ij} are $i \times j$ method-parameter matrices, $\boldsymbol{b}_j$, $\boldsymbol{d}_j$ and $\boldsymbol{c}_j$ are j-dimensional method-parameter vectors, $\boldsymbol{e}_j$ is the j-dimensional vector with unit entries (for $i, j = v, w$, $v + w = s$). The v-dimensional vector $\boldsymbol{V}_n$ is called the explicit stage subvector representing the numerical approximation to the exact solution vector $y(t_n \boldsymbol{e}_v + \boldsymbol{c}_v h)$ $= [y(t_n + c_1 h), \ldots, y(t_n + c_v h)]^T$, while the w-dimensional vector $\boldsymbol{W}_n$ stands for the implicit stage subvector representing the numerical approximation to the exact solution vector $y(t_n \boldsymbol{e}_w + \boldsymbol{c}_w h) = [y(t_n + c_{v+1} h), \ldots, y(t_n + c_s h)]^T$. Furthermore, in (2.1) and elsewhere in this paper, we employ for any two vectors $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_s)^T$, $\boldsymbol{\eta} = (\eta_1, \ldots, \eta_s)^T$ and any scalar function f the notation $f(\boldsymbol{\xi}, \boldsymbol{\eta}) := [f(\xi_1, \eta_1), \ldots, f(\xi_s, \eta_s)]^T$. Also we employ the notational convenience that a power of any vector means the vector consisiting of the componentwise powered one.

The method-parameter matrices A_{ij} and vectors $\boldsymbol{b}_j$ (for $i, j = v, w$, $v + w = s$) will be determined by the order conditions (see Theorem 2.1). This PTRK method is conveniently

specified by the tableau

$$\begin{array}{cc|c|cc} A_{vv} & A_{vw} & \boldsymbol{c}_v & O_{vv} & O_{vw} \\ O_{wv} & O_{ww} & \boldsymbol{c}_w & A_{wv} & A_{ww} \\ \hline & & u_{n+1} & \boldsymbol{b}_v^T & \boldsymbol{b}_w^T \end{array} \tag{2.2}$$

In order to start the method (2.1), an appropriate starting procedure is needed for generating sufficiently accurate starting stage vectors $\boldsymbol{V}_0$, $\boldsymbol{W}_0$ and step point value u_1 from $u_0 = y_0$. This can be done, for example, by using an appripriate PIRK method considered in [20] or a sequential RK code with dense output.

The s-stage PTRK method (2.1) consists of v explicit stages and w implicit stages. Its order can be studied in the same way as the order of RK methods. Thus suppose that $u_n = y(t_n)$, $\boldsymbol{V}_{n-1} = y(t_{n-1}\boldsymbol{e}_v + h\boldsymbol{c}_v)$ and $\boldsymbol{W}_{n-1} = y(t_{n-1}\boldsymbol{e}_w + h\boldsymbol{c}_w)$, then we have the following order definition (see [10]):

Definition 2.1 *The PTRK method* (2.1) *is said to be of order* p^* *if the equation*

$$y(t_{n+1}) - u_{n+1} = O(h^{p^*+1})$$

holds. Furthermore, it is said to be of stage order $q^* = min\{p^*, q_1, q_2\}$ *if, in addition to the above, the equations*

$$\begin{aligned} y(t_n\boldsymbol{e}_v + h\boldsymbol{c}_v) - \boldsymbol{V}_n &= O(h^{q_1+1}), \\ y(t_n\boldsymbol{e}_w + h\boldsymbol{c}_w) - \boldsymbol{W}_n &= O(h^{q_2+1}) \end{aligned}$$

hold.

The following theorem gives the order conditions for PTRK methods (see[10, Theorem 2.1]).

Theorem 2.1 *If the function* f *is Lipschitz continuous, and if the equations*

$$(A_{vv}, A_{vw})(\boldsymbol{c} - \boldsymbol{e})^{j-1} = \frac{\boldsymbol{c}_v^j}{j}, \qquad j = 1, \ldots, q_1, \tag{2.3a}$$

$$(A_{wv}, A_{ww})\boldsymbol{c}^{j-1} = \frac{\boldsymbol{c}_w^j}{j}, \qquad j = 1, \ldots, q_2, \tag{2.3b}$$

$$(\boldsymbol{b}_v^T, \boldsymbol{b}_w^T)\boldsymbol{c}^{j-1} = \frac{1}{j}, \qquad j = 1, \ldots, p, \tag{2.3c}$$

holds, then the PTRK method (2.1) *has order* $p^* = \min\{p, q_1 + 1, q_2 + 1\}$ *and stage order* $q^* = \min\{p^*, q_1, q_2\}$ *for any collocation vector* $\boldsymbol{c}$ *with distinct abscissae and for any integer pair* v, w *with* $w + v = s$.

In order to express the parameter matrices A_{vv}, A_{vw}, A_{wv}, A_{ww} and vectors $\boldsymbol{b}_v$, $\boldsymbol{b}_w$ explicitly in terms of the collocation vector $\boldsymbol{c}$, we introduce the matrices and vectors

$$\begin{aligned} P_v &:= \Big(\frac{\boldsymbol{c}_v}{1}, \frac{\boldsymbol{c}_v^2}{2}, \frac{\boldsymbol{c}_v^3}{3}, \frac{\mathbf{c}_v^4}{4} \ldots, \frac{\mathbf{c}_v^s}{s}\Big), & P_w &:= \Big(\frac{\boldsymbol{c}_w}{1}, \frac{\boldsymbol{c}_w^2}{2}, \frac{\mathbf{c}_w^3}{3}, \frac{\boldsymbol{c}_w^4}{4} \ldots, \frac{\mathbf{c}_w^s}{s}\Big), \\ R &:= \Big(\mathbf{e}, \boldsymbol{c}, \boldsymbol{c}^2, \boldsymbol{c}^3, \ldots, \boldsymbol{c}^{s-1}\Big), & \boldsymbol{g} &:= \Big(\frac{1}{1}, \frac{1}{2}, \ldots, \frac{1}{s}\Big)^T \\ Q &:= \Big(\boldsymbol{e}, (\boldsymbol{c} - \boldsymbol{e}), (\mathbf{c} - \boldsymbol{e})^2, \ldots, (\boldsymbol{c} - \boldsymbol{e})^{s-1}\Big). \end{aligned}$$

Then the order conditions (2.3) in Theorem 2.1 for $q_1 = q_2 = p = s$ can be presented in the form (cf. [10, (2.10)])

$$(A_{vv}, A_{vw})Q = P_v, \quad (A_{wv}, A_{ww})R = P_w, \quad (\boldsymbol{b}_v^T, \boldsymbol{b}_w^T)R = \boldsymbol{g}^T. \tag{2.4}$$

From (2.4) the parameter matrices and vectors of the PTRK method (2.1) can be expressed as follows

$$(A_{vv}, A_{vw}) = P_v Q^{-1}, \quad (A_{wv}, A_{ww}) = P_w R^{-1}, \quad (\boldsymbol{b}_v^T, \boldsymbol{b}_w^T) = \boldsymbol{g}^T R^{-1}. \tag{2.5}$$

According to Theorem 2.1, (2.5) implies

$$\begin{aligned} y(t_n \boldsymbol{e}_v + h\boldsymbol{c}_v) - \boldsymbol{V}_n &= O(h^{s+1}), \\ y(t_n \boldsymbol{e}_w + h\boldsymbol{c}_w) - \boldsymbol{W}_n &= O(h^{s+1}), \\ y(t_n + h) - u_{n+1} &= O(h^{p+1}). \end{aligned} \tag{2.6}$$

where $p = s$ for any collocation vector $\boldsymbol{c}$ with distinct abscissae. Moreover, the vectors $\boldsymbol{b}_v$, $\boldsymbol{b}_w$, defined in (2.5) which satisfy order conditions (2.3c), are the weights of the collocation implicit RK method (IRK method) based on collocation vector $\boldsymbol{c}$ (see g.e., [18]). Therefore, the order results for collocation IRK methods show that a special selection of the vector $\boldsymbol{c}$ may enable us to increase the order p beyond s (superconvergence) through the orthogonality relation (see also [18, p. 212]). The following theorem is a direct consequence of Theorem 2.1, the explicit expressions (2.5) of parameters of the PTRK methods (2.1) and the application of the orthogonality relation (see also [10]).

Theorem 2.2 *An s-stage PTRK method defined by (2.1) is of step point order $p^* = s$ and of stage order $q^* = s$ if the parameter matrices $A_{vv}, A_{vw}, A_{wv}, A_{ww}$ and vectors $\boldsymbol{b}_v$, $\boldsymbol{b}_w$, of the method satisfy the relations (2.5) for any collocation vector $\boldsymbol{c}$ with distinct abscissae and for any integer pair v, w with $w + v = s$. It has step point order $p^* = s + 1$ if, in addition, the equation*

$$P_j(1) = 0, \qquad P_j(x) := \int_0^x \xi^{j-1} \prod_{i=1}^{s} (\xi - c_i) d\xi, \qquad j = 1, \ldots, k,$$

holds for $k \geq 1$.

Theorem 2.2 indicates that an s-stage PTRK method can attain the step point order $p^* = s + 1$. According to the analysis of the local errors in this section, the starting vectors $\boldsymbol{V}_0$, $\boldsymbol{W}_0$ and the approximated value u_1 should be of order $s + 1$ and $p^* + 1$, respectively, that is

$$\begin{aligned} y(t_0 \boldsymbol{e}_v + h\boldsymbol{c}_v) - \boldsymbol{V}_0 &= O(h^{s+1}), \\ y(t_0 \boldsymbol{e}_w + h\boldsymbol{c}_w) - \boldsymbol{W}_0 &= O(h^{s+1}), \\ y(t_1) - u_1 &= O(h^{p^*+1}). \end{aligned}$$

Since the PTRK method (2.1) is of the two-step nature, its property of zero-stability is an important requirement. The following theorem gives the answer to this issue.

Theorem 2.3 *The PTRKN methods based on any colocation vectors $\boldsymbol{c}$ with distinct abscissae are always zero-stable for any integer pair v, w with $v + w = s$.*

This theorem was proved in [10] by expressing the PTRK methods into a sinlge-step form of the general linear method (GLM) (see e.g., [5]).

3 PIPTRK methods

In this section, we apply a parallel PC iteration scheme to the PTRK methods. Using (2.1) as the corrector method with the predictor formula

$$\begin{aligned}\boldsymbol{W}_n^{(0)} &= y_n\boldsymbol{e}_w + hB_{wv}f(t_{n-1}\boldsymbol{e}_v + h\boldsymbol{c}_v, \boldsymbol{V}_{n-1})\\ &\quad + hB_{ww}f(t_{n-1}\boldsymbol{e}_w + h\boldsymbol{c}_w, \boldsymbol{W}_{n-1}^{(m)}),\end{aligned} \tag{3.1a}$$

where the matrices B_{vw} and B_{ww} are determined by the order conditions given below (Section 3.1), we arrive at the following PC iteration scheme (in P(EC)^{k}E mode with $k = m$ or $k = m + 1$):

$$\begin{aligned}\boldsymbol{V}_n &= y_n\boldsymbol{e}_v + hA_{vv}f(t_{n-1}\boldsymbol{e}_v + h\boldsymbol{c}_v, \boldsymbol{V}_{n-1})\\ &\quad + hA_{vw}f(t_{n-1}\boldsymbol{e}_w + h\boldsymbol{c}_w, \boldsymbol{W}_{n-1}^{(m)}),\end{aligned} \tag{3.1b}$$

$$\begin{aligned}\boldsymbol{W}_n^{(j)} &= y_n\boldsymbol{e}_w + hA_{wv}f(t_n\boldsymbol{e}_v + h\boldsymbol{c}_v, \boldsymbol{V}_n)\\ &\quad + hA_{ww}f(t_n\boldsymbol{e}_w + h\boldsymbol{c}_w, \boldsymbol{W}_n^{(j-1)}), \qquad j = 1, \ldots, m,\end{aligned} \tag{3.1c}$$

$$y_{n+1} = y_n + h\boldsymbol{b}_v^T f(t_n\boldsymbol{e}_v + h\boldsymbol{c}_v, \boldsymbol{V}_n) + h\boldsymbol{b}_w^T f(t_n\boldsymbol{e}_w + h\boldsymbol{c}_w, \boldsymbol{W}_n^{(m)}). \tag{3.1d}$$

We note that u_n, u_{n+1} are the numerical approximations obtained by PTRK method (2.1) and y_n, y_{n+1} are those obtained by PIPTRK method (3.1). As for every explicit method, the computational cost of the method (3.1) is measured by the number of sequential $\boldsymbol{f}$-evaluations per step. Notice that v components of $f(t_n\boldsymbol{e}_v + h\boldsymbol{c}_v, \boldsymbol{V}_n)$ and w components of $f(t_n\boldsymbol{e}_w + h\boldsymbol{c}_w, \boldsymbol{W}_n^{(j-1)})$, $j = 1, \ldots, m + 1$, can be computed in parallel, provided that $\max(v, w)$ processors are available. Since $f(t_{n-1}\boldsymbol{e}_v+h\boldsymbol{c}_v, \boldsymbol{V}_{n-1})$ and $f(t_n\boldsymbol{e}_w+h\boldsymbol{c}_w, \boldsymbol{W}_{n-1})$ can be reused, in general, we need $m+2$ sequential $\boldsymbol{f}$-evaluations per step. However, by a special choice of collocation vector $\boldsymbol{c}$, one sequential $\boldsymbol{f}$-evaluation of components of $f(t_n\boldsymbol{e}_v+h\boldsymbol{c}_v, \boldsymbol{V}_n)$ can be saved and only w processors are needed (see Section 4). We shall call the iteration scheme defined by (3.1), a *parallel-iterated PTRK method* (PIPTRK method).

3.1 Order conditions for the predictor

The conditions for the predictor formula (3.1a) being of order s can be derived by replacing $\boldsymbol{V}_{n-1}$, y_n, $\boldsymbol{W}_{n-1}$ and $\boldsymbol{W}_n^{(0)}$ by the exact solution values $y(t_{n-1}\boldsymbol{e}_v + h\boldsymbol{c}_v) = y(t_n\boldsymbol{e}_v + h(\boldsymbol{c}_v - \boldsymbol{e}_v))$, $y(t_{n-1}\boldsymbol{e}_w + h\boldsymbol{c}_w) = y(t_n\boldsymbol{e}_w + h(\boldsymbol{c}_w - \boldsymbol{e}_w))$, $y(t_n)$ and $y(t_n\boldsymbol{e}_w + h\boldsymbol{c}_w)$, respectively. On substitution of these exact solution values into (3.1a), we are led to

$$\begin{aligned}y(t_n\boldsymbol{e}_w + h\boldsymbol{c}_w) - y(t_n)\boldsymbol{e}_w - hB_{wv}y'(t_n\boldsymbol{e}_v + h(\boldsymbol{c}_v - \boldsymbol{e}_v))\\ - hB_{ww}y'(t_n\boldsymbol{e}_w + h(\boldsymbol{c}_w - \boldsymbol{e}_w)) = O(h^{s+1}).\end{aligned} \tag{3.2}$$

Using the Taylor expansion for a sufficiently smooth function $y(t)$ in the neighbourhood of t_n, we can expand the left-hand side of (3.2) in powers of h and obtain the order conditions (cf., e.g., [8, 10])

$$\boldsymbol{c}_w^j - j[B_{wv}(\boldsymbol{c}_v - \boldsymbol{e}_v)^{j-1} + B_{ww}(\boldsymbol{c}_w - \boldsymbol{e}_w)^{j-1}] = \boldsymbol{0}, \quad j = 1, \ldots, s, \tag{3.3a}$$

or equivalently

$$(B_{wv}, B_{ww})(\boldsymbol{c}-\boldsymbol{e})^{j-1} = \frac{\boldsymbol{c}_w^j}{j}, \qquad j = 1, \ldots, s. \tag{3.3b}$$

The conditions (3.3) determine the matrix (B_{wv}, B_{ww}). With the matrices P_w and Q defined in Section 2, these conditions (3.3) can be written in the form

$$(B_{wv}, B_{ww})Q - P_w = O, \tag{3.4}$$

which leads to the explicit expression of the matrix (B_{wv}, B_{ww}),

$$(B_{wv}, B_{ww}) = P_w Q^{-1}. \tag{3.5}$$

If (3.4) is satisfied, then we have

$$\boldsymbol{W}_n - \boldsymbol{W}_n^{(0)} = [\boldsymbol{W}_n - y(t_n\boldsymbol{e}_w + h\boldsymbol{c}_w)] + [y(t_n\boldsymbol{e}_w + h\boldsymbol{c}_w) - \boldsymbol{W}_n^{(0)}] = O(h^{s+1}). \tag{3.6}$$

Since the function f is Lipschitz continuous and each iteration in (3.1) raises the order of the iteration error by 1, the following order relations are obtained:

$$\begin{aligned}
\boldsymbol{W}_n - \boldsymbol{W}_n^{(m)} &= O(h^{m+s+1}),\\
u_{n+1} - y_{n+1} &= h\boldsymbol{b}_w^T[f(\boldsymbol{W}_n) - f(\boldsymbol{W}_n^{(m)})] = O(h^{m+s+2}),\\
y(t_{n+1}) - y_{n+1} &= [y(t_{n+1}) - u_{n+1}] + [u_{n+1} - y_{n+1}] = O(h^{p^*+1}) + O(h^{m+s+2}),
\end{aligned}$$

where, p^* is the order of the PTRK corrector method (2.1). Thus, we have the following theorem:

Theorem 3.1 *If the PTRK method* (2.1) *has step point order* p^**, and if the conditions* (3.3) *(equivalently* (3.4) *or* (3.5)*) are satisfied, then the PIPTRK method* (3.1) *has step point order* $p^{**} = \min\{p^*, m+s+1\}$*, for any collocation vector* $\boldsymbol{c}$ *with distinct abscissae.*

3.2 Rate of convergence

The rate of convergence of the PIPTRK method (3.1) is defined by using the model test equation $y'(t) = \lambda y(t)$, where λ runs through the eigenvalues of the Jacobian matrix $\partial \boldsymbol{f}/\partial \boldsymbol{y}$ (cf., e.g., [7, 12, 14, 19]). Applying the method (3.1) to this model test equation, we obtain the iteration error equation

$$\boldsymbol{W}_n^{(j)} - \boldsymbol{W}_n = zA_{ww}\big[\mathbf{W}_n^{(j-1)} - \boldsymbol{W}_n\big], \quad z := h\lambda, \quad j = 1, \ldots, m. \tag{3.7}$$

Hence, in view of the model test equation, the rate of convergence is determined by the spectral radius $\rho(zA_{ww})$ of the iteration matrix zA_{ww}. Requiring that $\rho(zA_{ww}) < 1$, leads us to the convergence condition

$$|z| < \frac{1}{\rho(A_{ww})} \quad \text{or} \quad h < \frac{1}{\rho(\partial \boldsymbol{f}/\partial \boldsymbol{y})\rho(A_{ww})}. \tag{3.8}$$

We shall call $\rho(A_{ww})$ the *convergence factor* and $1/\rho(A_{ww})$ the *convergence boundary* of the PIPTRK method. The freedom in the choice of the collocation vector $\boldsymbol{c}$ of PTRK correctors can be used for minimizing the convergence factor $\rho(A_{ww})$, or equivalently, for maximizing the *convergence region* S_{conv} defined by

$$S_{conv} := \big\{z : |z| < 1/\rho(A_{ww})\big\}. \tag{3.9}$$

Specification of convergence factors for a specified class of PIPTRK methods used in our numerical experiments is reported in Section 4.

3.3 Stability regions

The linear stability of the PIPTRK methods (3.1) is investigated by again using the model test equation $y'(t) = \lambda y(t)$, where $\lambda \in \mathbb{C}^- := \{z : z \in \mathbb{C}, \quad \Re(z) \le 0\}$. Denoting $z := \lambda h$ and applying (3.1) to the model test equation yields

$$\boldsymbol{V}_n = zA_{vv}\boldsymbol{V}_{n-1} + zA_{vw}\boldsymbol{W}_{n-1}^{(m)} + \boldsymbol{e}_v y_n, \tag{3.10a}$$

$$\begin{aligned}
\boldsymbol{W}_n^{(m)} &= [I + zA_{ww} + \cdots + (zA_{ww})^{m-1}](\boldsymbol{e}_w y_n + zA_{wv}\boldsymbol{V}_n) + (zA_{ww})^m \boldsymbol{W}_n^{(0)} \\
&= [I + zA_{ww} + \cdots + (zA_{ww})^{m-1}](\boldsymbol{e}_w y_n + zA_{wv}\boldsymbol{V}_n) \\
&\quad + (zA_{ww})^m(\boldsymbol{e}_w y_n + zB_{wv}\boldsymbol{V}_{n-1} + zB_{ww}\boldsymbol{W}_{n-1}^{(m)}) \\
&= [I + zA_{ww} + \cdots + (zA_{ww})^{m-1}][\boldsymbol{e}_w y_n + zA_{wv}(\boldsymbol{e}_v y_n + zA_{vv}\boldsymbol{V}_{n-1} + zA_{vw}\boldsymbol{W}_{n-1}^{(m)})] \\
&\quad + (zA_{ww})^m(\boldsymbol{e}_w y_n + zB_{wv}\boldsymbol{V}_{n-1} + zB_{ww}\boldsymbol{W}_{n-1}^{(m)}) \\
&= \big\{z^2[I + zA_{ww} + \cdots + (zA_{ww})^{m-1}]A_{wv}A_{vv} + z^{m+1}(A_{ww})^m B_{wv}\big\}\boldsymbol{V}_{n-1} \\
&\quad + \big\{z^2[I + zA_{ww} + \cdots + (zA_{ww})^{m-1}]A_{wv}A_{vw} + z^{m+1}(A_{ww})^m B_{ww}\big\}\boldsymbol{W}_{n-1}^{(m)} \\
&\quad + \big\{[I + zA_{ww} + \cdots + (zA_{ww})^{m-1}](\boldsymbol{e}_w + zA_{wv}\boldsymbol{e}_v) + z^m(A_{ww})^m \boldsymbol{e}_w\big\} y_n \\
&= M_{21}^{(m)}(z)\boldsymbol{V}_{n-1} + M_{22}^{(m)}(z)\boldsymbol{W}_{n-1}^{(m)} + M_{23}^{(m)}(z) y_n,
\end{aligned} \tag{3.10b}$$

$$\begin{aligned}
y_{n+1} &= y_n + z\boldsymbol{b}_v^T\boldsymbol{V}_n + z\boldsymbol{b}_w^T\boldsymbol{W}_n^{(m)} \\
&= y_n + z\boldsymbol{b}_v^T(zA_{vv}\boldsymbol{V}_{n-1} + zA_{vw}\boldsymbol{W}_{n-1}^{(m)} + \boldsymbol{e}_v y_n \\
&\quad + z\boldsymbol{b}_w^T(M_{21}^{(m)}(z)\boldsymbol{V}_{n-1} + M_{22}^{(m)}(z)\boldsymbol{W}_{n-1}^{(m)} + M_{23}^{(m)}(z) y_n) \\
&= \big\{z^2\boldsymbol{b}_v^T A_{vv} + z\boldsymbol{b}_w^T M_{21}^{(m)}(z)\big\}\boldsymbol{V}_{n-1} + \big\{z^2\boldsymbol{b}_v^T A_{vw} + z\boldsymbol{b}_w^T M_{22}^{(m)}(z)\big\}\boldsymbol{W}_{n-1}^{(m)} \\
&\quad + \big\{1 + z\boldsymbol{b}_v^T\boldsymbol{e}_v + z\boldsymbol{b}_w^T M_{23}^{(m)}(z)\big\} y_n \\
&= M_{31}^{(m)}(z)\boldsymbol{V}_{n-1} + M_{32}^{(m)}(z)\boldsymbol{W}_{n-1}^{(m)} + M_{33}^{(m)}(z) y_n.
\end{aligned} \tag{3.10c}$$

From (3.10) we are led to the recursion

$$\begin{pmatrix} \boldsymbol{V}_n \\ \boldsymbol{W}_n^{(m)} \\ y_{n+1} \end{pmatrix} = M_m(z) \begin{pmatrix} \boldsymbol{V}_{n-1} \\ \boldsymbol{W}_{n-1}^{(m)} \\ y_n \end{pmatrix}, \tag{3.11a}$$

where $M_m(z)$ is the $(s+1)\times(s+1)$ matrix defined by

$$M_m(z) = \begin{pmatrix} zA_{vv} & zA_{vw} & \boldsymbol{e}_v \\ M_{21}^{(m)}(z) & M_{22}^{(m)}(z) & M_{23}^{(m)}(z) \\ M_{31}^{(m)}(z) & M_{32}^{(m)}(z) & M_{33}^{(m)}(z) \end{pmatrix} \tag{3.11b}$$

The explicit formulas of $M_{ij}^{(m)}(z)$ with $i = 2, 3$, $j = 1, 2, 3$ in (3.11b) are clear from (3.10). The matrix $M_m(z)$ in (3.11) which determines the stability of the PIPTRK methods, will be called the *amplification matrix*, whereas its spectral radius $\rho\big(M_m(z)\big)$ the *stability function*. For a given number of iterations m, the stability region of the PIPTRK methods are defined as

$$S_{stab}(m) := \big\{z : \rho\big(M_m(z)\big) < 1, \quad \Re(z) \le 0\big\}.$$

The real and imaginary boundaries for a given m, $\beta_{re}(m)$ and $\beta_{im}(m)$, respectively, can be defined in the familiar way. The stability pairs $\big(\beta_{re}(m), \beta_{im}(m)\big)$ for the PIPTRK methods used in our numerical experiments can be found in Section 4.

4 Numerical experiments

In this paper, we report numerical results for PIPTRK methods with $w = s/2$ and $v = s/2$, where $s = 4, 6, 8, 10$ (see Section 3). The PTRK corrector methods defined by (2.1) are based on collocation vectors $\boldsymbol{c}$ defined by

$$\boldsymbol{c}_v = (c_1, \ldots, c_k)^T, \quad \boldsymbol{c}_w = (1 + c_1, \ldots, 1 + c_k)^T, \quad k = 2, 3, 4, 5, \tag{4.1}$$

where $c_1, \ldots, c_k$ are k components of the k-dimensional Gauss-Legendre collocation vector. This choice of $\boldsymbol{c}$ gives the PTRK correctors step point order $p^* = s$ and stage order $q^* = s$ (see Theorem 2.2), so that the corresponding PIPTRK methods defined by (3.1) have step point order $p^{**} = s$ for any m (see Theorem 3.1). Also by this choice, v components of $f(t_n\boldsymbol{e}_v + h\boldsymbol{c}_v, \boldsymbol{V}_n)$ can be copied from the preceding step. Therefore the resulting PIPTRK methods require $m + 1$ sequential $\boldsymbol{f}$-evaluations per step and can be implemented on $w = p^{**}/2$ processors. We do not claim that the above chosen collocation vectors are the best possible. A further study of this topic will be subject of future research. These orders and number of processors are the same as used for the PIRK methods proposed in [20]. However, a direct numerical computation reveals that the convergence factors of PIPTRK methods as defined in Section 3.2 are smaller than those of PIRK methods of the same order (see Table 1). Table 2 lists the satbility boundaries of the PIPTRK methods. As shown in this table, the stability pairs of these PIPTRK methods are sufficiently large for nonstiff problems. Especially, all methods have a non-empty intersection of $S_{stab}(m)$ with the imaginary axis. We shall compare the PIPTRK methods with parallel and sequential explicit RK methods from the literature. In the numerical experiments, for the first step, the starting velues $\boldsymbol{V}_0$, $\boldsymbol{W}_0$ and y_1 of a PIPTRK method will be generated by the associated PIRK method based on the same collocation vector $\boldsymbol{c}$ as the underlying PIPTRK method. The absolute error obtained at the end point of the integration interval is presented in the form 10^{-NCD} (NCD

Table 1: Convergence factors for various parallel pth-order methods

Parallel pth-order methods	$p=4$	$p=6$	$p=8$	$p=10$
PIRK (cf. [7])	0.289	0.215	0.165	0.137
PIPTRK	0.194	0.136	0.106	0.086

Table 2: Stability pairs $(\beta_{re}(m), \beta_{im}(m))$ for various PIPTRK methods

Parallel pth-order methods	$p=4$	$p=6$	$p=8$	$p=10$
$m=1$	(0.937, 0.818)	(0.474, 0.115)	(0.240, 0.234)	(0.092, 0.092)
$m=2$	(1.206, 1.377)	(0.944, 0.123)	(0.551, 0.562)	(0.278, 0.266)
$m=3$	(1.796, 1.700)	(1.342, 0.124)	(0.911, 0.588)	(0.484, 0.509)
$m=4$	(2.035, 2.100)	(1.796, 0.124)	(1.190, 0.810)	(0.871, 0.890)
$m=5$	(2.476, 2.173)	(2.245, 0.124)	(1.701, 1.215)	(0.863, 0.856)
$m=6$	(2.782, 2.662)	(2.641, 0.124)	(1.711, 1.645)	(0.933, 0.917)

may be interpreted as the number of correct decimal digits). The computational efforts are measured by the values of N_{seq} denoting the total number of sequential $\boldsymbol{f}$-evaluations required over the total number of integration steps N_{stp}.

Ignoring load balancing factors and communication times between processors in parallel methods, the comparison of various methods in this section is based on N_{seq} and the obtained $NCDs$. The numerical experiments with small widely-used test problems taken from the literature below show a potential superiority of the new PIPTRK methods over the extant methods. This superiority will be significant in a parallel machine if the test problems are large enough and/or the $\boldsymbol{f}$-evaluations are expensive (cf. e.g., [3]). All the computations were carried out on a 29-digit precision computer. An actual implementation with stepsize strategy for large and expensive problems on a parallel machine is a subject of further studies.

4.1 Comparison with parallel PC methods

In order to see the efficiency of various parellel PC methods, we follow a dynamical strategy in all PC methods for determining the number of iterations in the successive steps. It seems natural to require that the iteration error is of the same order in h as the local error of the corrector. This leads us to the stopping criterion (cf. e.g., [7, 12])

$$\|\boldsymbol{W}_n^{(m)} - \boldsymbol{W}_n^{(m-1)}\|_\infty \leq TOL = Ch^{p^*}, \tag{4.2}$$

where C is a problem- and method-dependent parameter, and p^* is order of the PTRK corrector. We shall report numerical results obtained by ones of the best parallel explicit RK methods available in the literature, that is the PIRK methods proposed in [20] and the

methods constructed in this paper. We selected a test set of three problems taken from the literature.

4.1.1 Two body problem

As a first numerical test, we apply the various pth-order PC methods to the *two body problem* on the integration interval $[0, 20]$, with eccentricity $\varepsilon = \frac{3}{10}$ (cf., e.g., [20, 22])

$$\begin{aligned} y_1'(t) &= y_3(t), & y_1(0) &= 1-\varepsilon, \\ y_2'(t) &= y_4(t), & y_2(0) &= 0, \\ y_3'(t) &= \frac{-y_4(t)}{[y_1^2(t)+y_2^2(t)]^{3/2}}, & y_3(0) &= 0, \\ y_4'(t) &= \frac{-y_2(t)}{[y_1^2(t)+y_2^2(t)]^{3/2}}, & y_4(0) &= \sqrt{\frac{1+\varepsilon}{1-\varepsilon}}. \end{aligned} \tag{4.3}$$

The numerical results listed in Table 3 clearly show that the PIPTRK methods are much more efficient than the PIRK methods of the same order. For this problem, all the PIPTRK methods need only about two iterations per step.

Table 3: Values of NCD/N_{seq} for problem (4.3) obtained by various pth-order parallel PC methods

PC methods	p	$N_{stp}=100$	$N_{stp}=200$	$N_{stp}=400$	$N_{stp}=800$	$N_{stp}=1600$	C
PIRK	4	3.1/441	3.7/905	4.9/1947	6.1/4000	7.3/8000	10^0
PIPTRK	4	3.7/230	4.2/431	5.2/812	6.3/1604	7.5/3204	10^0
PIRK	6	5.0/643	7.2/1302	8.9/2637	10.5/5499	12.3/11200	10^{-1}
PIPTRK	6	5.3/285	7.1/526	8.9/972	10.7/1903	12.5/3661	10^{-1}
PIRK	8	7.6/837	10.4/1686	12.8/3397	15.0/6845	17.3/13827	10^{-2}
PIPTRK	8	7.8/353	10.2/649	12.7/1156	15.1/2193	17.5/4035	10^{-2}
PIRK	10	9.3/926	12.8/1926	16.3/3927	19.2/8226	22.2/16532	10^{-2}
PIPTRK	10	10.6/382	13.3/656	16.4/1198	19.5/2245	22.5/4260	10^{-2}

4.1.2 Fehlberg problem

For the second numerical test, we apply the various pth-order PC methods to the often-used *Fehlberg problem* on the integration interval $[0, 5]$ (cf. e.g., [7, 20, 22])

$$\begin{aligned} y_1'(t) &= 2ty_1(t)\log\big(\max\{y_2(t), 10^{-3}\}\big) \qquad y_1(0) = 1, \\ y_2'(t) &= -2ty_2(t)\log\big(\max\{y_1(t), 10^{-3}\}\big) \qquad y_2(0) = \mathrm{e}, \end{aligned} \tag{4.4}$$

with the exact solution $y_1(t) = \exp\big(\sin(t^2)\big)$, $y_2(t) = \exp\big(\cos(t^2)\big)$. The numerical results are reported in Table 4. These numerical results show that the PIPTRK methods are again far superior to the PIRK methods of the same order.

Table 4: Values of NCD/N_{seq} for problem (4.4) obtained by various pth-order parallel PC methods

PC methods	p	$N_{stp}=100$	$N_{stp}=200$	$N_{stp}=400$	$N_{stp}=800$	$N_{stp}=1600$	C
PIRK	4	2.7/392	4.0/842	5.2/1756	6.5/3650	7.7/7409	10^3
PIPTRK	4	2.9/227	4.3/432	5.8/829	7.2/1612	8.4/3201	10^3
PIRK	6	5.2/601	7.0/1245	8.9/2542	10.7/5199	12.5/10488	10^3
PIPTRK	6	6.0/302	8.4/563	10.3/1039	12.2/1946	14.0/3764	10^3
PIRK	8	7.8/774	10.2/1603	12.6/3297	15.1/6674	17.5/13468	10^3
PIPTRK	8	8.6/376	10.8/673	13.3/1217	15.8/2296	18.3/4385	10^3
PIRK	10	9.9/942	12.9/1947	15.9/3973	18.9/8134	22.0/16407	10^3
PIPTRK	10	11.0/454	14.2/791	17.2/1443	20.2/2663	23.2/4877	10^3

4.1.3 Jacobian elliptic functions problem

The final numerical example is the Jacobian elliptic functions sn, cn, dn problem for the equation of motion of a rigid body without external forces on theintegration interval $[0, 20]$ (cf., e.g., [18, problem JACB, p. 240], also [22])

$$\begin{aligned} y_1'(t) &= y_2(t)y_3(t), & y_1(0) &= 0, \\ y_2'(t) &= -y_1(t)y_3(t), & y_2(0) &= 1, \\ y_3'(t) &= -0.51y_1(t)y_2(t), & y_3(0) &= 1. \end{aligned} \tag{4.5}$$

The exact solution is given by the Jacobi elliptic functions $y_1(t) = sn(t : k)$, $y_2(t) = cn(t; k)$, $y_3(t) = dn(t; k)$ (see [16]). The numerical results for this problem are given in Table 5 and give rise to nearly the same conclusions as formulated in the two previous examples.

Table 5: Values of NCD/N_{seq} for problem (4.5) obtained by various pth-order parallel PC methods

PC methods	p	$N_{stp}=100$	$N_{stp}=200$	$N_{stp}=400$	$N_{stp}=800$	$N_{stp}=1600$	C
PIRK	4	2.3/300	5.1/800	6.3/1600	7.5/3200	8.9/6571	10^1
PIPTRK	4	4.5/202	6.7/403	7.7/803	8.8/1603	10.0/3203	10^1
PIRK	6	5.1/486	7.8/1126	11.2/2345	12.5/4775	14.3/9600	10^0
PIPTRK	6	7.9/205	10.0/405	11.8/805	13.6/1605	15.4/3205	10^0
PIRK	8	8.2/678	11.1/1470	14.0/3028	16.7/6195	19.1/12540	10^{-1}
PIPTRK	8	9.8/243	12.7/433	16.1/807	18.5/1607	20.9/3207	10^{-1}
PIRK	10	10.1/765	13.4/1655	16.8/3479	19.6/7095	23.2/14968	10^{-1}
PIPTRK	10	12.0/265	15.5/474	20.1/809	23.8/1609	26.4/3209	10^{-1}

4.2 Comparison with sequential methods

In Section 4.1, the PIPTRK methods were compared with PIRK methods. In this section, we shall compare these PIPTRK methods with some of the best sequential explicit RK methods currently available.

In order to compare the methods of comparable order, we restricted the numerical experiments to the comparison of our 8^{th} order PIPTRK method denoted by PIPTRK8 with two sequential codes `DOPRI5` and `DOP853` for the Fehlberg problem (4.4). These `DOPRI5` and `DOP853`codes are embedded explicit RK methods due to Dormand and Prince andcoded by Hairer and Waner (see [18]). They are based on pair 5(4) and "triple" 8(5)(3), respectively. `DOP853` is the new version of `DOPRI8` with a "stretched" error estimator (see [18, p. 254]). These two codes belong to the most efficient currently existing sequential codes for nonstiff first-order ODE problems. We took the best results obtained by `DOPRI5` and `DOP853` given in [9] and added the results in the low accuracy range obtained by PIPTRK8 method. In spite of the fact that the results of the sequential codes are obtained using a stepsize strategy, whereas PIPTRK8 method is applied with fixed stepsizes, it is the PIPTRK8 method that is the most efficient (see Table 6).

Table 6: Comparison with sequential methods for problem (4.4)

Methods	N_{stp}	NCD	N_{seq}
`DOPRI5` (from [18])	75	3.2	452
	162	5.3	974
	393	7.4	2360
	979	9.4	5876
	2458	11.4	14750
`DOP853` (from [18])	47	4.5	552
	70	6.2	825
	107	8.0	1265
	164	10.2	1950
	261	12.2	3123
PIPTRK8 (in this paper)	25	3.3	147
	50	5.8	220
	100	8.6	376
	200	10.8	673
	400	13.3	1217

5 Concluding remarks

In this paper, we proposed a new class of paralle PC iteration methods called parallel-iterated pseudo two-step RK methods (PIPTRK methods) based on pseudo two-step RK correctors

(PTRK correctors). Three numerical experiments showed that the PIPTRK methods are much superior to the known parallel and sequential methods available in the literature. In the forthcoming papers, we will pursue the study of PIPTRK methods which involves with the optimal choice of the method parameters and variable stepsize control.

References

[1] K. Burrage, *Efficient block predictor-corrector methods with a small number of corrections*, J. Comput. Appl. Math. **45** (1993), 139-150.

[2] K. Burrage, *Parallel methods for initial value problems*, Appl. Numer. Math. **11** (1993), 5-25.

[3] K. Burrage, *Parallel and Sequential Methods for Ordinary DifferentialEquations*, Clarendon Press, Oxford, 1995.

[4] K. Burrage and H. Suhartanto, *Parallel iterated methods based on multistep Runge-Kutta mehods of Radau type*, Advances in Computational Mathematics **7** (1997), 37-57.

[5] J.C. Butcher, *The Numerial Analysys of Ordinary Differential Equations, Runge-Kutta and General Linear Methods*, Wiley, New York, 1987.

[6] M.T. Chu and H. Hamilton, *Parallel solution of ODEs by multi-block methods*, SIAM J. Sci. Statist. Comput. **3** (1987), 342-353.

[7] N.H. Cong, *Parallel iteration of symmetric Runge-Kutta for nonstiff initial-value problems*, J. Comput. Appl. Math. **51** (1994), 117-125.

[8] N.H. Cong, *Explicit pseudo two-step Runge-Kutta methods for parallel computers*, Intern. J. Comput. Math. **73** (1999), 77-91.

[9] N.H. Cong, *Continuous variable stepsize explicit pseudo two-step RK methods*, J. Comput. Appl. Math. **101** (1999), 105-116.

[10] N.H. Cong, *A general family of pseudo two-step Runge-Kutta methods*, SEA Bull. Math. **25** (2001), 61-73.

[11] N.H. Cong and T. Mitsui, *Collocation-based two-step Runge-Kutta methods*, Japan J. Indust. Appl. Math. **13** (1996), 171-183.

[12] N.H. Cong and T. Mitsui, *A class of explicit parallel two-step Runge-Kutta methods*, Japan J. Indust. Appl. Math. **14** (1997), 303-313.

[13] N.H. Cong, H. Podhaisky and R. Weiner, *Numerical experiments with some explicit pseudo two-step RK methods on a shared memory computer*, Comput. Math. Appl. **36** (1998), 107-116.

[14] N.H. Cong and H.T. Vi, *An improvement for explicit parallel Runge-Kutta methods*, Vietnam J. Math. **23** (1995), 241-252.

[15] A.R. Curtis, *High-order explicit Runge-Kutta formulae, their uses and limitations*, J. Inst. Math. Appl. **16** (1975), 35-55.

[16] A.R. Curtis, *Tables of Jacobian Elliptic Functions Whose Arguments are Rational Fractions of the Quarter Period*, H.M.S.O., London, 1964.

[17] E. Hairer, *A Runge-Kutta method of order 10*, J. Inst. Math. Appl. **21** (1978), 47-59.

[18] E. Hairer, S.P. Nørsett and G. Wanner, *Solving Ordinary Differential Equations, I. Nonstiff Problems*, 2^{nd} edition, Springer-Verlag, Berlin, 1993.

[19] P.J. van der Houwen and N.H. Cong, *Parallel block predictor-corrector methods of Runge-Kutta type*, Appl. Numer. Math. **13** (1993), 109-123.

[20] P.J. van der Houwen and B.P. Sommeijer, *Parallel iteration of high-order Runge-Kutta methods with stepsize control*, J. Comput. Appl. Math. **29** (1990), 111-127.

[21] P.J. van der Houwen and B.P. Sommeijer, *Block Runge-Kutta methods on parallel computers*, Z. Angew. Math. Mech. **68** (1992), 3-10.

[22] T.E. Hull, W.H. Enright, B.M. Fellen and A.E. Sedgwick, *Comparing numerical methods for ordinary differential equations*, SIAM J. Numer. Anal. **9** (1972), 603-637.

[23] S.P. Nørsett and H.H. Simonsen, *Aspects of parallel Runge-Kutta methods*, in Numerical Methods for Ordinary Differential Equations, Proceedings L'Aquilla 1987, Lecture Notes in Mathematics, **1386**, (Edited by A. Bellen, C.W. Gear and E. Russo), Springer-Verlag, Berlin, 1989.