SIAM J. SCI. COMPUT.
Vol. 23, No. 6, pp. 2041–2054

FOURTH ORDER CHEBYSHEV METHODS WITH RECURRENCE RELATION[*]

ASSYR ABDULLE[†]

Dedicated to Professor Gerhard Wanner on the occasion of his 60th birthday

Abstract. In this paper, a new family of fourth order Chebyshev methods (also called stabilized methods) is constructed. These methods possess nearly optimal stability regions along the negative real axis and a three-term recurrence relation. The stability properties and the high order make them suitable for large stiff problems, often space discretization of parabolic PDEs. A new code ROCK4 is proposed, illustrated at several examples, and compared to existing programs.

Key words. stiff ordinary differential equations, explicit Runge–Kutta methods, orthogonal polynomials, parabolic partial differential equations

AMS subject classifications. 65L20, 65M20

PII. S1064827500379549

1. Introduction.

Chebyshev methods are a class of explicit Runge–Kutta methods with extended stability domains along the negative real axis. The stability properties of these methods make them suitable for stiff problems which possess a Jacobian matrix with (possibly large) eigenvalues close to the real negative axis. Since they are explicit, Chebyshev methods avoid linear algebra difficulties and can be applied to very large problems. The main applications are parabolic PDEs when discretized by finite difference. It usually gives a large system of ODEs with a symmetric and negative definite Jacobian matrix. Thus, the eigenvalues of the discretized parabolic PDEs are real negative and, furthermore, become larger while refining the space discretization.

Recently, a new strategy to construct second order Chebyshev methods has been proposed by Abdulle and Medovikov [2]. It combines the advantages of the methods introduced by Lebedev [11], [12] and van der Houwen and Sommeijer [9] (see also [14] for the latest implementation of these methods). An algorithm to construct nearly optimal stability functions along the real negative axis based on orthogonal polynomials was proposed in [2]. The advantage of using orthogonal polynomials is the three-term recurrence relation which can be used to construct the numerical methods. At the same time, choosing an appropriate weight function for these polynomials leads to a nearly optimal stability domain (see [2]).

For order more than 2, the only known Chebyshev methods are those of Medovikov [10]. They are constructed upon the strategy of Lebedev-type methods: the zeros of the optimal stability polynomials are computed, and the numerical methods are based on a suitable ordering of these zeros. The drawback is that the ordering, crucial for the internal stability of the methods, depends on the degree of the polynomials and needs some art. There are also no recurrence relations. The methods of order 4 proposed in this paper are based on a three-term recurrence relation and avoid the preceding problems.

[*] Received by the editors October 16, 2000; accepted for publication (in revised form) October 18, 2001; published electronically February 27, 2002.
http://www.siam.org/journals/sisc/23-6/37954.html

[†] Section de Mathématiques, Université de Genève, CH-1211 Genève 24, Switzerland (Assyr.Abdulle@math.unige.ch).

The paper is organized as follows. In section 2 we explain how to compute nearly optimal fourth order stability polynomials. Section 3 is devoted to the construction of a family of numerical methods. These methods have been implemented in a new code called ROCK4 which is briefly described in section 4. Finally, in section 5 we present some numerical experiments and comparisons with other codes.

2. Fourth order stability polynomials. The aim is to construct a family of polynomials of order 4 depending on the degree s,[1]

$$\widetilde{R}_s(z) = 1 + z + \frac{z^2}{2!} + \frac{z^3}{3!} + \frac{z^4}{4!} + \mathcal{O}(z^5), \tag{2.1}$$

which remain bounded by 1 as long as possible along the real negative axis, i.e.,

$$|\widetilde{R}_s(z)| \leq 1 \quad \text{for } z \in [-\widetilde{l}_s, 0], \tag{2.2}$$

with $\widetilde{l}_s$ as large as possible. It is proved in [1] that fourth order optimal stability polynomials (which exist and are unique see [13]) possess exactly four complex zeros. Thus, they can be written as

$$\widetilde{R}_s(z) = \widetilde{w}_4(z)\widetilde{P}_{s-4}(z), \tag{2.3}$$

where $\widetilde{w}_4(z) = (1-/t_1)(1-/\bar{t}_1)(1-/t_2)(1-/\bar{t}_2)$, t_i are complex (conjugate) numbers, and $\widetilde{P}_{s-4}(z)$ possesses only real zeros. The idea developed in [2] for second order is to approximate these polynomials by

$$R_s(z) = w_4(z)P_{s-4}(z), \tag{2.4}$$

where $w_4(z) = (1-/z_1)(1-/\bar{z}_1)(1-/z_2)(1-/\bar{z}_2)$ and $P_{s-4}(z)$ is an orthogonal polynomial associated with the weight function $w_4(z)^2/\sqrt{1-z^2}$. We want to find such a decomposition which satisfies (2.2) for $z \in [-l_s, 0]$ with l_s close to $\widetilde{l}_s$. At the same time, we want that the product satisfies the fourth order conditions (2.1).

The motivation for considering such polynomials can be found in [2]. Notice that for first order optimal polynomials we have a formula similar to (2.3), with $\widetilde{w}(z) = 1$ and $\widetilde{R}_s(z) = T_s(1 + \frac{z}{s^2})$, where $T_s(z)$ are the Chebyshev polynomials. These optimal polynomials are at the same time orthogonal with respect to the weight function $1/\sqrt{1-z^2}$. The arguments developed in [2] show that for order 2 the polynomials (2.3) and (2.4) are very close. These arguments can be generalized for even orders higher than 2. Figure 2.1 shows for $s = 9$ the difference between a fourth order optimal stability polynomial and its approximation. We see that they can hardly be distinguished.

For second order, the algorithm for computing the orthogonal polynomials and the zeros of the function $w(z)$ of (2.4) was given in [2]. We adapt here this algorithm for order 4.

In the following we will work in the normalized interval $[-1, 1]$ instead of $[-l_s, 0]$ by setting $x = 1 + \frac{2z}{l_s}$ (l_s is the length of the stability domain along $\mathbb{R}^-$ we want to optimize), and we take the same notation for the shifted polynomials except for the fact that we use the variable x instead of z. Thus, we are searching for order 4 at $x = 1$ (see below). If we shift the polynomials defined by (2.4) and normalize them

[1] We use the notation $\widetilde{R}_s$ because we reserve R_s for the stability polynomials we will construct.

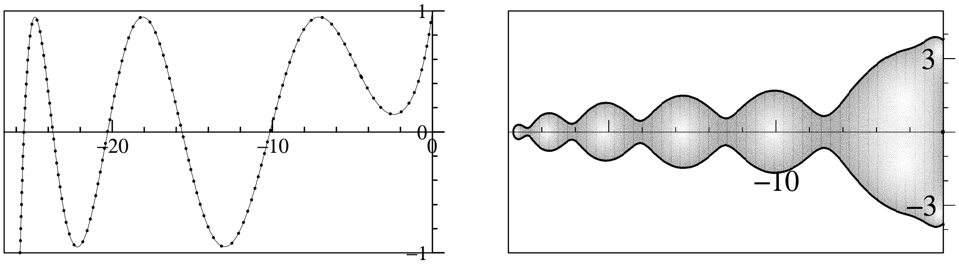


FIG. 2.1. *$R_9(x)$ and its stability domain, with $\widetilde{R}_9(x)$ in the dotted line, $\eta = 0.95$ (damping).*

such that $|w_4(x)P_{s-4}(x)| \leq 1$ for $x \in [-1,1]$, then $w_4(1)P_{s-4}(1)$ is usually not equal to 1. As in [2], we therefore introduce a parameter a close to 1, set

$$R_s(x) = \frac{w_4(x)P_{s-4}(x)}{w_4(a)P_{s-4}(a)}, \tag{2.5}$$

and we search fourth order conditions at the point a. We denote the complex zeros of $w_4(x)$ by $x_1 = \alpha + i\beta, \bar{x}_1 = \alpha - i\beta, x_2 = \gamma + i\delta, \bar{x}_2 = \gamma - i\delta$, and to emphasize the dependence of $R_s(x)$ on $\alpha, \beta, \gamma, \delta$ we will write $R_s(x, \alpha, \beta, \gamma, \delta)$. We have now an optimization problem.

Problem. Find $a, d, \alpha, \beta, \gamma, \delta$ (depending on s) such that

$$R_s'(a, \alpha, \beta, \gamma, \delta) = d, \qquad R_s''(a, \alpha, \beta, \gamma, \delta) = d^2, \tag{2.6}$$

$$R_s'''(a, \alpha, \beta, \gamma, \delta) = d^3, \qquad R_s^{(4)}(a, \alpha, \beta, \gamma, \delta) = d^4, \tag{2.7}$$

$$|R_s(x)| \leq 1 \quad \text{for} \quad x \in [-1, a] \tag{2.8}$$

$$\text{with} \quad l_s = (1+a)d \quad \text{as large as possible.} \tag{2.9}$$

If we then set $z = (x-a)/d$, we will have

$$R_s(0) = 1, \quad R_s'(0) = 1, \quad R_s''(0) = 1, \quad R_s'''(0) = 1, \quad R_s^{(4)}(0) = 1, \tag{2.10}$$

$$|R_s(z)| \leq 1 \quad \text{for} \quad z \in [-l_s, 0]. \tag{2.11}$$

Here again we used the same notation for the shifted polynomials. For more details about this algorithm we refer the reader to [2].

We have computed the parameters $l_s, \alpha, \beta, \gamma, \delta, a$ (depending on s) for degree 5 up to degree more than 1000. In practice the bound in (2.8) should be replaced by a value $\eta < 1$ so that the stability domain is at a safe distance from the real axis (see Figure 2.1). We chose $\eta = 0.95$. In Table 2.1 we give the values of l_s and $c(s) = l_s/s^2$ for some degrees s. Several authors observed that the optimal values $\widetilde{l}_s$ satisfy $\widetilde{l}_s = c(s)s^2$, with $c(s)$ rapidly approaching a limit value. The value $c(s) \simeq 0.34$ is given in [15] (see also [8]) and $c(s) \simeq 0.35$ in [10]. We observe in Table 2.1 that $l_s \simeq 0.35 \cdot s^2$, which means that our stability regions are nearly optimal.

TABLE 2.1
The stability parameters of $R_s(x)$.

Degree s	Stability region l_s	Value $c(s) = l_s/s^2$	Degree s	Stability region l_s	Value $c(s) = l_s/s^2$
5	5.9983	0.239931	100	3538.1276	0.353813
10	32.4470	0.324470	250	22184.4995	0.354952
20	138.3586	0.345897	500	88746.9995	0.354988
50	879.8864	0.351955	750	199684.4999	0.354995

3. Fourth order Chebyshev methods.

In the preceding section we have explained how to compute our stability polynomials. Assume now that we have such a family of fourth order polynomials (depending on the degree s)

$$R_s(z) = w_4(z)P_{s-4}(z), \tag{3.1}$$

which satisfy

$$|R_s(z)| \le 1 \quad \text{for} \quad z \in [-l_s, 0], \tag{3.2}$$

where $w_4(z) = (1 - /z_1)(1 - /\bar{z}_1)(1 - /z_2)(1 - /\bar{z}_2)$ and $P_{s-4}(z)$ is an orthogonal polynomial associated with the weight function

$$w_4(z)^2/\sqrt{1-z^2}, \tag{3.3}$$

and normalized such that $P_{s-4}(0) = 1$. For a given degree s we will further use the family of orthogonal polynomials $(P_j)_{j=0}^{s-4}$ associated with the weight function (3.3), normalized such that $P_j(0) = 1$. These polynomials possess a three-term recurrence relation

$$P_j(z) = (\mu_j z - \nu_j)P_{j-1}(z) - \kappa_j P_{j-2}(z).$$

In [2] it was explained how to compute explicitly these polynomials, given the zeros of the function (3.3). The same procedure can be applied here, and the recurrence coefficients can be computed simply by solving a linear system.

For second order methods, it is sufficient to construct a numerical method which is of order 2 for linear problems, since the order conditions are the same for both linear and nonlinear problems. This is not true for orders larger than 2, and it is therefore not sufficient to consider only the linear case. Here we have to give a realization of our weight function $w_4(z)$ so that the order conditions up to order 4 (8 conditions) are satisfied. As in [10] we will use the theory of composition of methods (the "Butcher group") to realize fourth order Runge–Kutta–Chebyshev methods.

Suppose that we have two Runge–Kutta methods. The idea of composition of methods is to apply one method after the other to an initial value y_0 with the same step size. The result of this process can be interpreted as a large Runge–Kutta method: a composition of the two latter methods. For the theory of composition of Runge–Kutta methods, we refer to [3],[6, pp. 264–273],[5].

To construct a fourth order Runge–Kutta method with the polynomials (3.1) we proceed as follows:

- We construct a first method, denoted by P, which possesses $P_{s-4}(z)$ as stability polynomial.
- We then determine a second method, denoted by W, which possesses $w_4(z)$ as stability polynomial, to achieve fourth order for the "composite" method denoted by WP.

$$\begin{array}{c|cccc} 0 & & & & \\ \widetilde{c}_2 & \widetilde{a}_{21} & & & \\ \vdots & \vdots & \ddots & & \\ \widetilde{c}_{s-4} & \widetilde{a}_{s-4,1} & \dots & \widetilde{a}_{s-4,s-5} & \\ \hline & \widetilde{b}_1 & \dots & \widetilde{b}_{s-5} & \widetilde{b}_{s-4} \end{array} \qquad \begin{array}{c|cccc} 0 & & & & \\ \widehat{c}_2 & \widehat{a}_{21} & & & \\ \widehat{c}_3 & \widehat{a}_{31} & \widehat{a}_{32} & & \\ \widehat{c}_4 & \widehat{a}_{41} & \widehat{a}_{42} & \widehat{a}_{43} & \\ \hline & \widehat{b}_1 & \widehat{b}_2 & \widehat{b}_3 & \widehat{b}_4 \end{array}$$

FIG. 3.1. *Tableau of the method P (left) and W (right).*

The resulting method will be of order 4 and will possess $R_s(z) = w_4(z)P_{s-4}(z)$ as stability polynomial.

The first method. To construct the method P we apply a procedure similar to that used in [2]. That is, the three-term recurrence relation of the orthogonal polynomials $(P_j)_{j=1}^{s-4}$ is used to define the internal stages of the method as follows:

$$\begin{aligned} &g_0 := y_0, \\ &g_1 := y_0 + h\mu_1 f(g_0), \\ &g_j := h\mu_j f(g_{j-1}) - \nu_j g_{j-1} - \kappa_j g_{j-2}, \qquad j = 2, \dots, s-4, \\ &y_1 := g_{s-4}. \end{aligned} \tag{3.4}$$

Applied to $y' = \lambda y$ with $z = h\lambda$ yields

$$g_{s-4} = P_{s-4}(z) g_0. \tag{3.5}$$

This method is given in the left tableau of Figure 3.1. The coefficients of the method P can be expressed recursively in term of the coefficients ν_j, μ_j, κ_j. Indeed, using the notation $k_i = f(y_0 + h\sum_{j=1}^{i-1} \widetilde{a}_{ij} k_j)$ (autonomous form), we obtain for the first stage

$$g_1 = y_0 + h\mu_1 f(y_0) = y_0 + h\widetilde{a}_{21} k_1; \tag{3.6}$$

this yields $a_{21} = \mu_1$. For the second stage we have

$$\begin{aligned} g_2 &= h\mu_2 f(y_0 + h\widetilde{a}_{21}k_1) - \nu_2(y_0 + h\widetilde{a}_{21}k_1) - \kappa_2 y_0 \\ &= y_0(-\nu_2 - \kappa_2) + h(-\nu_2\widetilde{a}_{21})k_1 + h\mu_2 k_2; \end{aligned} \tag{3.7}$$

hence $\widetilde{a}_{31} = -\nu_2 \widetilde{a}_{21}$ and $\widetilde{a}_{32} = \mu_2$. (Notice that $-\nu_2 - \kappa_2 = 1$ because of the normalization $P_j(0) = 1$.) The third stage is then given by

$$\begin{aligned} g_3 &= h\mu_3 f(y_0 + h(\tilde{a}_{31}k_1 + \tilde{a}_{32}k_2)) - \nu_3(y_0 + h(\tilde{a}_{31}k_1 + \tilde{a}_{32}k_2)) - \kappa_3(y_0 + h\tilde{a}_{21}k_1) \\ &= y_0(-\nu_3 - \kappa_3) + h(-\nu_3\tilde{a}_{31} - \kappa_3\tilde{a}_{21})k_1 + h(-\nu_3\tilde{a}_{32})k_2 + h\mu_3 k_3, \end{aligned}$$

and thus $\tilde{a}_{41} = -\nu_3\tilde{a}_{31} - \kappa_3\tilde{a}_{21}$, $\tilde{a}_{42} = -\nu_3\tilde{a}_{32}$ and $\tilde{a}_{43} = \mu_3$. By induction we obtain the following lemma.

LEMMA 3.1. *For the method P given by* (3.4) *the coefficients $\widetilde{a}_{ij}, \widetilde{b}_i, \widetilde{c}_i$ of the corresponding Runge–Kutta method are given by*

$$\begin{array}{lcl} \widetilde{a}_{i,j} &=& -\nu_{i-1}\widetilde{a}_{i-1,j} - \kappa_{i-1}\widetilde{a}_{i-2,j}, \qquad j \le i-2, \qquad i \le s-3 \ (\widetilde{a}_{jj} := 0), \\ \widetilde{a}_{i,i-1} &=& \mu_{i-1}, \qquad i \le s-3, \\ \widetilde{b}_j &=& \widetilde{a}_{s-3,j}, \qquad 1 \le j \le s-4, \end{array} \tag{3.8}$$

and the $\widetilde{c}_i$ satisfy the usual relation $\widetilde{c}_i = \sum_{j=1}^{i-1} \widetilde{a}_{ij}$. □

We emphasize that the coefficients $\tilde{a}_{ij}, \tilde{b}_i$ will be used only to compute the method W. For the implementation of the method, formulas (3.4) will be used.

$$
\begin{array}{c|cccccccc}
0 & & & & & & & & \\
\widetilde{c}_2 & \widetilde{a}_{21} & & & & & & & \\
\vdots & \vdots & \ddots & & & & & & \\
\widetilde{c}_{s-4} & \widetilde{a}_{s-4,1} & \dots & \widetilde{a}_{s-4,s-5} & & & & & \\
\sum \widetilde{b}_i & \widetilde{b}_1 & \dots & \widetilde{b}_{s-3} & \widetilde{b}_{s-4} & & & & \\
\sum \widetilde{b}_i + \widehat{c}_2 & \widetilde{b}_1 & \dots & \widetilde{b}_{s-3} & \widetilde{b}_{s-4} & \widehat{a}_{21} & & & \\
\vdots & \vdots & & & & & \ddots & & \\
\sum \widetilde{b}_i + \widehat{c}_4 & \widetilde{b}_1 & \dots & \widetilde{b}_{s-3} & \widetilde{b}_{s-4} & \widehat{a}_{41} & \dots & \widehat{a}_{43} & \\
\hline
 & \widetilde{b}_1 & \dots & \widetilde{b}_{s-3} & \widetilde{b}_{s-4} & \widehat{b}_1 & \dots & \widehat{b}_3 & \widehat{b}_4
\end{array}
$$

FIG. 3.2. *Tableau of method WP.*

The second and the "composite" methods. For the method W we take a fourth stage method (right tableau of Figure 3.1) so that the composite method WP is given by Figure 3.2. In the tableau of method WP, we will denote by c_i the elements of the first column, by a_{ij} the elements of the "triangle," and by b_i the elements of the last row. The order conditions of the method WP are the usual ones for order 4:

$$
\begin{aligned}
wp(\bullet) &= \sum b_i &&= 1,\\
wp([\bullet]) &= 2\sum b_i a_{ij} &&= 1,\\
wp([\bullet,\bullet]) &= 3\sum b_i a_{ij} a_{ik} &&= 1,\\
wp([[\bullet]]) &= 6\sum b_i a_{ij} a_{jk} &&= 1,\\
wp([\bullet,\bullet,\bullet]) &= 4\sum b_i a_{ij} a_{ik} a_{il} &&= 1,\\
wp([[\bullet],\bullet]) &= 8\sum b_i a_{ij} a_{jk} a_{il} &&= 1,\\
wp([[\bullet,\bullet]]) &= 12\sum b_i a_{ij} a_{jk} a_{jl} &&= 1,\\
wp([[[\bullet]]]) &= 24\sum b_i a_{ij} a_{jk} a_{kl} &&= 1.
\end{aligned}
\tag{3.9}
$$

Here we used the trees notation (connected graphs without cycles and a distinguished vertex) for the elementary weights ($wp(\dots)$) involved in the order conditions (see [6, pp. 145–154] or [3]).

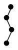
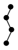
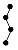
Theorem 12.6 of [6, p. 267] can be used to express the order conditions of the method WP in function of the two submethods, W and P. See also [5] for a new simple proof of this latter theorem (with another normalization for the elementary weights). We obtain

$$
\begin{aligned}
wp(\bullet) &= w(\bullet) + p(\bullet),\\
wp([\bullet]) &= w([\bullet]) + 2w(\bullet)p(\bullet) + p([\bullet]),\\
wp([\bullet,\bullet]) &= w([\bullet,\bullet]) + 3w([\bullet])p(\bullet) + 3w(\bullet)p(\bullet)^2 + p([\bullet,\bullet]),\\
wp([[\bullet]]) &= w([[\bullet]]) + 3w([\bullet])p(\bullet) + 3w(\bullet)p([\bullet]) + p([[\bullet]]),\\
wp([\bullet,\bullet,\bullet]) &= w([\bullet,\bullet,\bullet]) + 4w([\bullet,\bullet])p(\bullet) + 6w([\bullet])p(\bullet)^2 + 4w(\bullet)p(\bullet)^3 + p([\bullet,\bullet,\bullet]),
\end{aligned}
$$

$wp($ $) = w($ $) + 4(\tfrac{1}{3}w($ $)p(\bullet) + \tfrac{2}{3}w($ $)p(\bullet)) + 6(\tfrac{1}{3}w($ $)p($ $) + \tfrac{2}{3}w($ $)p(\bullet)^2)$

$+ 4w(\bullet)p(\bullet)p($ $) + p($ $),$

$wp($ $) = w($ $) + 4w($ $)p(\bullet) + 6w($ $)p(\bullet)^2 + 4w(\bullet)p($ $) + p($ $),$

$wp($$) = w($$) + 4w($ $)p(\bullet) + 6w($ $)p($ $) + 4w(\bullet)p($ $) + p($$),$

where $w(\dots)$ and $p(\dots)$ are the expressions (3.9) for the methods W and P, respectively. These formulas allow us to compute recursively the expressions $w(\bullet), w($ $), \dots$ for the method W, since the expressions $p(\bullet), p($ $), \dots$ can be computed with the coefficients of the method P given by (3.8). This leads to the following equations for the method W:

(3.10)

$\widehat{b}_1 + \widehat{b}_2 + \widehat{b}_3 + \widehat{b}_4 = w(\bullet),$

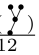
$\widehat{b}_2\widehat{c}_2 + \widehat{b}_3\widehat{c}_3 + \widehat{b}_4\widehat{c}_4 = w($ $)/2,$

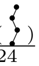
$\widehat{b}_2\widehat{c}_2^2 + \widehat{b}_3\widehat{c}_3^2 + \widehat{b}_4\widehat{c}_4^2 = w($ $)/3,$

$\widehat{b}_3\widehat{a}_{32}\widehat{c}_2 + \widehat{b}_4(\widehat{a}_{42}\widehat{c}_2 + \widehat{a}_{43}\widehat{c}_3) = w($ $)/6,$

$\widehat{b}_2\widehat{c}_2^3 + \widehat{b}_3\widehat{c}_3^3 + \widehat{b}_4\widehat{c}_4^3 = w($ $)/4,$

$\widehat{b}_3\widehat{c}_3\widehat{a}_{32}\widehat{c}_2 + \widehat{b}_4\widehat{c}_4(\widehat{a}_{42}\widehat{c}_2 + \widehat{a}_{43}\widehat{c}_3) = w($ $)/8,$

$\widehat{b}_3\widehat{a}_{32}\widehat{c}_2^2 + \widehat{b}_4(\widehat{a}_{42}\widehat{c}_2^2 + \widehat{a}_{43}\widehat{c}_3^2) = w($$)/12,$

$\widehat{b}_4\widehat{a}_{43}\widehat{a}_{32}\widehat{c}_2 = w($$)/24.$

This is a system of 8 equations for 10 unknowns $\widehat{b}_i, \widehat{a}_{ij}$ ($\widehat{c}_i$ are determined by $\widehat{c}_i = \sum_{j=1}^{i-1}\widehat{a}_{ij}$.) These equations are similar to the equations of usual fourth order methods (i.e., when $w(\bullet) = 1, w($ $) = 1, \dots$; see [6, pp. 135–136]). We have two degrees of freedom, and we choose $\widehat{c}_3 = \frac{w(\bullet)}{3}$ and $\widehat{c}_4 = \frac{2w(\bullet)}{3}$ after some experimentation. This choice keeps the absolute value of the coefficients $\widehat{b}_i, \widehat{a}_{ij}$ less than 1, which is suitable for a numerical method.

The solution of equations (3.10) gives us the coefficients of the method W. Thus, we have obtained a family of methods (depending on the degree of the stability polynomial) of order 4 with recurrence formulas and with the polynomial (3.1) as a stability function.

The embedded method. For the estimation of the local error of the constructed numerical method

$$y_1 = y_0 + h\sum_{i=1}^{s} b_i f\left(y_0 + h\sum_{j=1}^{i-1} a_{ij}k_j\right) = y_0 + h\sum_{i=1}^{s} b_i k_i, \tag{3.11}$$

we use an embedded method of order 3.

Since for fourth order methods there is no embedded method of order 3 using the same function values $k_i = f(y_0 + h\sum a_{ij}k_j)$ (see [6, p. 167]), we search for a lower

$$
\begin{array}{c|ccccc}
0 & & & & & \\
\widehat{c}_2 & \widehat{a}_{21} & & & & \\
\widehat{c}_3 & \widehat{a}_{31} & \widehat{a}_{32} & & & \\
\widehat{c}_4 & \widehat{a}_{41} & \widehat{a}_{42} & \widehat{a}_{43} & & \\
\hline
\widehat{c}_5 & \widehat{b}_1 & \widehat{b}_2 & \widehat{b}_3 & \widehat{b}_4 & \\
\hline
 & \bar{b}_1 & \bar{b}_2 & \bar{b}_3 & \bar{b}_4 & \bar{b}_5
\end{array}
$$

FIG. 3.3. *Tableau of the method $\overline{W}$.*

order method of the form

(3.12)

$$\overline{y}_1 = y_0 + h\left(\sum_{i=1}^{s}\beta_i f\left(y_0 + h\sum_{j=1}^{i-1} a_{ij}k_j\right) + \beta_{s+1} f(y_1)\right) = y_0 + h\left(\sum_{i=1}^{s}\beta_i k_i + \beta_{s+1}k_{s+1}\right).$$

This is no extra work (if the step is accepted) because $f(y_1)$ has to be computed anyway for the next stage. As a measure of the error after one step we will take

$$err = \|y_1 - \overline{y}_1\|. \tag{3.13}$$

We want to keep the recurrence formulas for the embedded method. Therefore, the embedded method will be a composition of the method P, defined in (3.4), with a new method W denoted by $\overline{W}$. For that, we add a fifth stage to the method W as shown in Figure 3.3. Similarly to (3.10) we derive third order conditions for the method $\overline{W}$:

$$
\begin{aligned}
&\bar{b}_1 + \bar{b}_2 + \bar{b}_3 + \bar{b}_4 + \bar{b}_5 &&= w(\bullet),\\
&\bar{b}_2\widehat{c}_2 + \bar{b}_3\widehat{c}_3 + \bar{b}_4\widehat{c}_4 + \bar{b}_4\widehat{c}_5 &&= \frac{w(\substack{\bullet\\ /\\ \bullet})}{2},\\
&\bar{b}_2\widehat{c}_2^2 + \bar{b}_3\widehat{c}_3^2 + \bar{b}_4\widehat{c}_4^2 + \bar{b}_5\widehat{c}_5^2 &&= \frac{w(\substack{\bullet\,\bullet\\ \vee\\ \bullet})}{3},\\
&\bar{b}_3\widehat{a}_{32}\widehat{c}_2 + \bar{b}_4(\widehat{a}_{42}\widehat{c}_2 + \widehat{a}_{43}\widehat{c}_3) + \bar{b}_5(\widehat{b}_2\widehat{c}_2 + \widehat{b}_3\widehat{c}_3 + \widehat{b}_4\widehat{c}_4) &&= \frac{w(\substack{\bullet\\ \bullet\\ \bullet})}{6}.
\end{aligned}
\tag{3.14}
$$

Notice that $\widehat{c}_5 = \sum \widehat{b}_i = w(\bullet)$. The last equation of (3.14) can be simplified since $\widehat{b}_2\widehat{c}_2 + \widehat{b}_3\widehat{c}_3 + \widehat{b}_4\widehat{c}_4 = \frac{w(\substack{\bullet\\ /\\ \bullet})}{2}$ (see (3.10)). We obtain a system of 4 equations for 5 unknowns $\bar{b}_i$. We require the following additional condition:

$$\bar{w}_5(-l_s) = 0, \tag{3.15}$$

where $\bar{w}_5(z)$ is the stability polynomial of the method $\overline{W}$ and l_s the length of the stability domain along the negative real axis (see (3.2)).

Solving (3.14) and (3.15), numerical computations show that the stability polynomials of the embedded methods are bounded (by $\eta = 0.95$) on the same interval as the stability polynomials of the numerical methods (see Figure 3.4). There is a simple criterion to check if $\bar{R}_{s+1}(z)$, the stability polynomials of the embedded method, is bounded by η on the same interval as $R_s(z)$. Recall that $R_s(z) = w_4(z)P_{s-4}(z)$

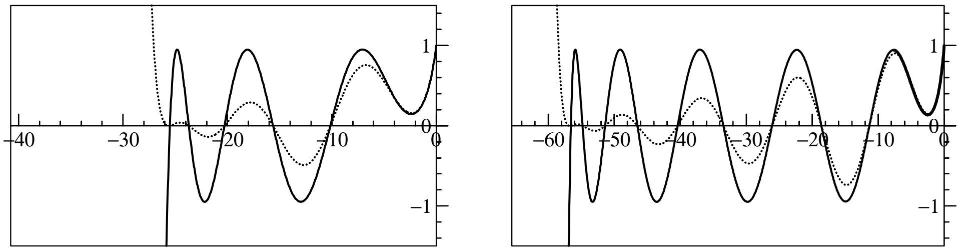


FIG. 3.4. *Stability polynomials of the method WP and the embedded method $\overline{W}P$ (dotted line) for $s = 9$ (left) and $s = 13$ (right).*

and $\bar{R}_{s+1}(z) = \bar{w}_5(z)P_{s-4}(z)$. Denote by z_i the zeros of $P_{s-4}(z)$. Because $P_{s-4}(z)$ is an orthogonal polynomial its zeros are simple and all in the stability interval, say $-l_s < z_1 < \cdots < z_{s-4} < 0$.

LEMMA 3.2. *With the above notations, suppose that $\bar{R}_{s+1}(-l_s) = 0$. If there exist $\delta > 0$ such that $\bar{R}_{s+1}(z) < R_s(z)$ for $z \in (z_{s-4}, z_{s-4} + \delta)$, then*

$$|\bar{R}_{s+1}(z)| \leq |R_s(z)| \quad \textit{for } z \in (z_1, z_{s-4} + \delta). \tag{3.16}$$

Proof. Define $d(z) = \bar{R}_{s+1}(z) - R_s(z)$; we have

$$d(z) = P_{s-4}(z)z^4(\beta_0 + \beta_1 z),$$

since $\bar{R}_{s+1}(z) - e^z = \mathcal{O}(z^4)$ and $R_s(z) - e^z = \mathcal{O}(z^5)$.

Suppose that there exist $z \in (z_1, z_{s-4})$ such that $|\bar{R}_{s+1}(z)| > |R_s(z)|$. Then, using the second hypothesis, either there exist $\widehat{z} \neq z_i$ such that $d(\widehat{z}) = 0$ or $d(z)$ has a double zero at some z_i. In both cases, it must then exist $\epsilon > 0$ such that $|\bar{R}_{s+1}(z)| > |R_s(z)|$ for $z \in (z_1, z_1 + \epsilon)$. Since $\bar{R}_{s+1}(-l_s) = 0$, $d(z)$ must have a double zero at z_1 or vanish at least once in $(-l_s, z_1)$. In both cases, counting the zeros of $d(z)$ outside of the origin leads to a contradiction. □

4. Description of ROCK4.

We implemented the numerical method described in section 3 in a code called ROCK4 for Orthogonal-Runge–Kutta–Chebyshev (appropriately permuted). This is the fourth order version of the code ROCK2 introduced in [2]. In this section we briefly describe the code.

Step size estimation. As in ROCK2, we implemented the "step size strategy with memory" of Watts [16] and Gustafsson [4],

$$h_{\text{new}} = \text{fac} \cdot h_n \left(\frac{1}{\text{err}_{n+1}}\right)^{\frac{1}{4}} \frac{h_n}{h_{n-1}} \left(\frac{\text{err}_n}{\text{err}_{n+1}}\right)^{\frac{1}{4}}, \tag{4.1}$$

in order to allow the step size to decrease reasonably without rejection (see also [7, p. 124]).

Stage number selection. While most stiff codes have a fixed number of stages, we used, as usual in Chebyshev codes, a family of fourth order methods. At each step we first select a step size in order to control the local error; then we select a stage order so that the stability property (see section 2 and Table 2.1)

$$h\rho\left(\frac{\partial f}{\partial y}(y)\right) \leq 0.35s^2 \tag{4.2}$$

is satisfied, where $\rho(\dots)$ denotes the spectral radius of the Jacobian matrix of the ODEs. This is possible because for practical purposes, the error constants of the family of fourth order methods are found to be almost the same. They are close to the error constants of optimal stability polynomials which have been described in [1].

Spectral radius estimate. The user can supply a function which estimates a bound for the spectral radius (for example, by using Gershgorin theorem; see [6, p. 89]). By specifying that the Jacobian is constant, this function will be called only once. If it is not possible to get an estimate of the spectral radius easily, the code ROCK4 can also compute it internally. For that, we have implemented with a slight modification a nonlinear power method proposed by Sommeijer, Shampine, and Verwer (see [14]).

Storage. Due to the three-term recurrence formula, the method requires only a few storage vectors. The number of storage vectors does not depend on the number of stages used. For the computation of an integration step and the error estimation, ROCK4 uses three vectors for the recurrence formula and five additional vectors for the finishing four-stage method and the embedded method. Notice that the low memory demand of these methods is suitable since we want to apply them to large problems.

5. Numerical experiments. We conclude this paper with several stiff problems taken from the test set of stiff problems proposed in [7] (first and second editions). All the parameters chosen for the examples are taken from [7]. We compare the following codes:

ROCK4: the fourth order code described in this paper.

ROCK2: the second order code based on orthogonal polynomials and described in [2].

RKC: the second order Chebyshev code of Sommeijer, Shampine, and Verwer (see [14]).

RADAU5: the well-known implicit code by Hairer and Wanner of order 5 based on a Radau IIA collocation method (see [7]).

For all examples which follow, we compared the obtained numerical results for the different codes with a reference solution for the given ODEs. The computing time is then displayed as a function of the error (in an Euclidian norm). For each problem the codes have been applied with different tolerances, say

$$tol = 10^{-2}, 10^{-2-\frac{1}{4}}, 10^{-2-\frac{1}{2}}, \dots . \tag{5.1}$$

The integer-exponent tolerances are displayed as enlarged symbols. The results were computed with scalar tolerances $atol = rtol = tol$ for all problems. The symbol for $tol = 10^{-5}$ is distinguished by its gray color.

The following examples are parabolic PDEs, discretized by the method of line into a system of ODEs. We replace the second order spatial derivatives by the finite difference scheme

$$\frac{\partial^2 u(x_i, y_j, t)}{\partial x^2} = \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{(\Delta x)^2} + \mathcal{O}\big((\Delta x)^2\big),$$

where u_{ij} are functions depending on time.

Example 1. The first example is the Burgers' equation

$$u_t + \left(\frac{u^2}{2}\right)_x = \mu u_{xx}, \tag{5.2}$$

with initial condition

$$u(x,0) = 1.5x(1-x)^2, \tag{5.3}$$

and boundary conditions

$$u(0,t) = u(1,t) = 0 \tag{5.4}$$

for $0 \leq x \leq 1$ and $0 \leq t \leq 2.5$. We discretize the space variable of (5.2) by the method of lines with $\Delta x = \frac{1}{501}$, and we choose $\mu = 0.0003$.

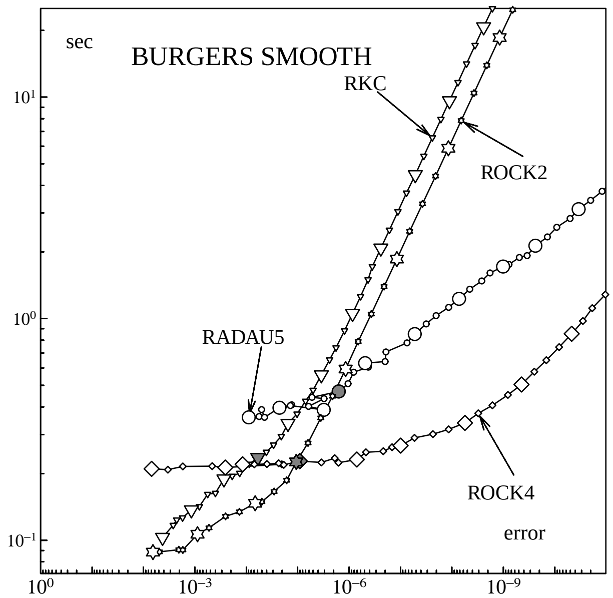


FIG. 5.1. *Work-precision diagram for Burgers' equations.*

Thus, we obtain an ODE (in time) of dimension 500. It is then solved by the different codes for $0 \leq t \leq 2.5$. For RADAU5 we used the banded algebra option, and for the Chebyshev codes we provide an estimation of the spectral radius by applying the Gershgorin theorem.

We see in Figure 5.1 that the two lower order methods, RKC and ROCK2, are better for lower tolerances. ROCK2 is slightly more efficient and better at delivering an accuracy close to the tolerance. For higher tolerances the high order codes RADAU5 and ROCK4 are better, with an advantage for ROCK4. This latter code also nicely preserves the tolerance proportionality.

Example 2. The second example is the two-dimensional Brusselator reaction-diffusion problem

$$\begin{aligned} \frac{\partial u}{\partial t} &= 1 + u^2 v - 4.4u + \alpha\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) + f(x,y,t), \\ \frac{\partial v}{\partial t} &= 3.4u - u^2 v + \alpha\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right), \end{aligned} \tag{5.5}$$

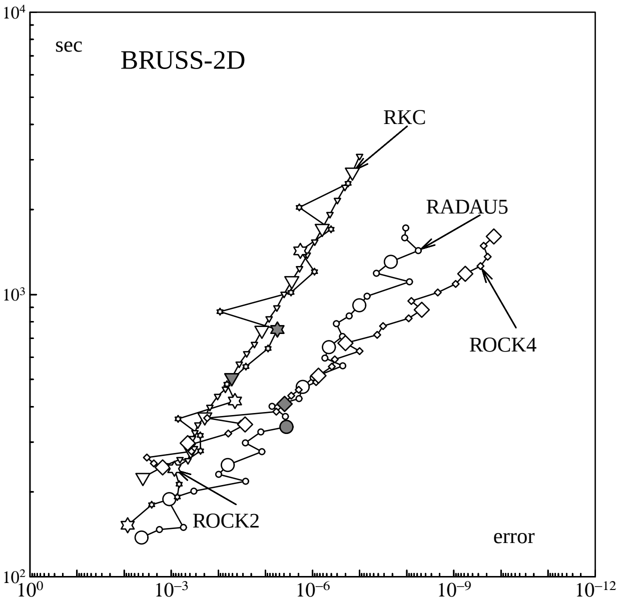


Fig. 5.2. *Work-precision diagram for the two-dimensional Brusselator problem.*

with initial conditions

$$u(x,y,0) = 22 \cdot y(1-y)^{3/2}, \qquad v(x,y,0) = 27 \cdot x(1-x)^{3/2},$$

and periodic boundary conditions

$$u(x+1,y,t) = u(x,y,t), \qquad u(x,y+1,t) = u(x,y,t)$$

for $0 \leq x \leq 1,\ 0 \leq y \leq 1,\ t \geq 0$. The function f is defined by

$$f(x,y,t) = \begin{cases} 5 & \text{if } (x-0.3)^2 + (y-0.6)^2 \leq 0.1^2 \text{ and } t \geq 1.1, \\ 0 & \text{else.} \end{cases}$$

We discretize the space variables of equations (5.5) with $x_i = \frac{i}{N+1}$, $y_i = \frac{i}{N+1}$, $i = 1, 2, \dots, N$ and choose $N = 128$ and $\alpha = 0.1$. Thus, we obtain a system of $2N^2 = 32768$ equations. We chose the output points $t_{\mathrm{out}} = 1.5$ and 11.5. The spectral radius of the Jacobian $\rho \simeq 13200$ can be estimated with the Gershgorin theorem; thus as in the previous example, we provide a bound for it when using Chebyshev methods. As advised in [7, p. 157] the linear equations in the code RADAU5 are solved by FFT methods so that the code is optimized for this problem. (Otherwise it will certainly not be competitive with Chebyshev methods.)

We see in Figure 5.2 that ROCK2 and RKC behaves similarly. For higher order methods, RADAU5 behaves better for low tolerances, while ROCK4 is better for higher tolerances. Between Chebyshev codes, except for very low tolerances ROCK4 gives the best results and nicely preserves the tolerance proportionality (as do ROCK2 and RADAU5).

Example 3. The third example is the FitzHugh and Nagumo model for explaining

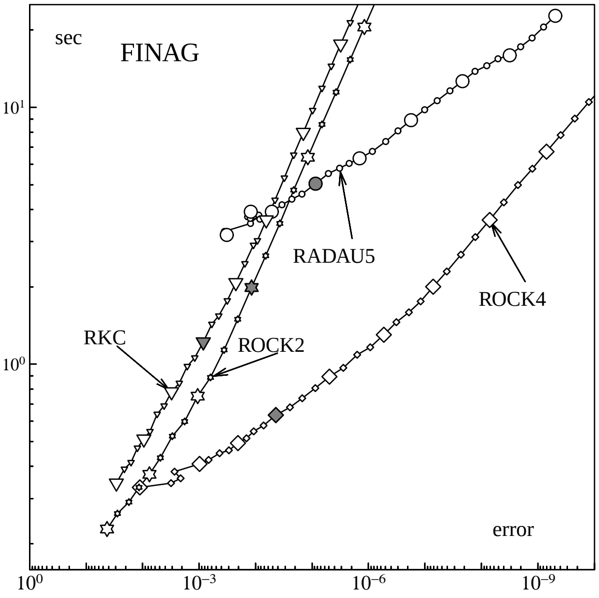


FIG. 5.3. *Work-precision diagram for FitzHugh and Nagumo equations.*

the nerve conduction as a traveling wave:

$$\begin{aligned} \frac{\partial u}{\partial t} &= \frac{\partial^2 u}{\partial x^2} - f(u) - v, \\ \frac{\partial v}{\partial t} &= \eta(u - \beta v), \end{aligned} \tag{5.6}$$

where

$$f(u) = u(u-\alpha)(u-1),$$

with initial conditions

$$u(x,0) = v(x,0) = 0,$$

and boundary conditions

$$\frac{\partial u}{\partial x}(0,t) = -0.3, \qquad \frac{\partial u}{\partial x}(100,t) = 0$$

for $0 \leq x \leq 100$ and $0 \leq t \leq 400$.

We chose $\alpha = 0.139$, $\eta = 0.008$, and $\beta = 2.54$ and discretize the space variable in 200 equidistant steps $x_i = \frac{2i+1}{4},\ i = 0, \ldots, 199$. We compute numerically a bound for the spectral radius of the Jacobian which was used for the Chebyshev methods. This is a mildly stiff problem, and we see in Figure 5.3 the advantage of the Chebyshev methods compared to an implicit one. Again, ROCK2 works slightly better than RKC. ROCK4 behaves well and is better compared to other Chebyshev methods even for low tolerances.

Conclusion. We have presented a new fourth order Chebyshev method and implemented it in a code called ROCK4. The numerical examples show a good behavior of this code for problems it is intended for. We emphasize that this code, as does other Chebyshev codes, is very simple to use. In fact, it is as simple to use as the forward Euler method. The first version of this code (as well as ROCK2) and some examples are available on the Internet at the address http://www.unige.ch/math/folks/hairer/software.html. Experiences with this code are welcome.

Acknowledgments. The author is grateful to Ernst Hairer and Gerhard Wanner for helpful discussions. He also thanks Alexei Medovikov for his constant interest in this subject.

REFERENCES

[1] A. Abdulle, *On roots and error constant of optimal stability polynomials*, BIT, 40 (2000), pp. 177–182.

[2] A. Abdulle and A.A. Medovikov, *Second order Chebyshev methods based on orthogonal polynomials*, Numer. Math., 90 (2001), pp. 1–18.

[3] J.C. Butcher, *The Numerial Analysis of Ordinary Differential Equations. Runge-Kutta and General Linear Methods*, John Wiley and Sons, Chichester, UK, 1987.

[4] K. Gustafsson, *Control-theoretic techniques for stepsize selection in implicit Runge-Kutta methods*, ACM Trans. Math. Software, 20 (1994), pp. 496–517.

[5] E. Hairer, C. Lubich, and G. Wanner, *Geometric Numerical Integration*, Springer-Verlag, Berlin, Heidelberg, New York, 2002.

[6] E. Hairer, S.P. Nørsett, and G. Wanner, *Solving Ordinary Differential Equations* I. *Nonstiff Problems*, 2nd ed., Springer Ser. Comput. Math. 8, Springer-Verlag, Berlin, 1993.

[7] E. Hairer and G. Wanner, *Solving Ordinary Differential Equations* II. *Stiff and Differential-Algebraic Problems*, 2nd ed., Springer Ser. Comput. Math. 14, Springer-Verlag, Berlin, 1996.

[8] P.J. van der Houwen, *Construction of Integration Formulas for Initial Value Problems*, North-Holland, Amsterdam, 1977.

[9] P.J van der Houwen and B.P. Sommeijer, *On the internal stability of explicit m-stage Runge-Kutta methods for large m-values*, Z. Angew. Math. Mech., 60 (1980), pp. 479–485.

[10] A.A. Medovikov, *High order explicit methods for parabolic equations*, BIT, 38 (1998), pp. 372–390.

[11] V.I. Lebedev, *A new method for determining the zeros of polynomials of least deviation on a segment with weight and subject to additional conditions. Part* I, Russian J. Numer. Anal. Math. Modelling., 8 (1993), pp. 195–222.

[12] V.I. Lebedev, *A new method for determining the zeros of polynomials of least deviation on a segment with weight and subject to additional conditions. Part* II, Russian J. Numer. Anal. Math. Modelling., 8 (1993), pp.397–426.

[13] W. Riha, *Optimal stability polynomials,* Computing, 9 (1972), pp. 37–43.

[14] B.P. Sommeijer, L.F. Shampine, and J.G. Verwer, *RKC: An explicit solver for parabolic PDEs*, J. Comput. Appl. Math., 88 (1998) pp. 316–326.

[15] J.W. Verwer, *Explicit Runge-Kutta methods for parabolic partial differential equations*, Appl. Numer. Math., 22 (1996), pp. 359–379.

[16] H.A. Watts, *Step size control in ordinary differential equation solvers*, Trans. Soc. Computer Simulation, 1 (1984) pp. 15–25.