Journal of Siberian Federal University. Mathematics & Physics 2009, 2(3), 319–3276

УДК 519.622

First-Order Methods With Extended Stability Regions for Solving Electric Circuit Problems

Mikhail V. Rybkov*
Institute of Mathematics and Computer Science,
Siberian Federal University,
Svobodny, 79, Krasnoyarsk, 660041,
Russia

Lyudmila V. Knaub†
Institute of Mathematics and Computer Science,
Siberian Federal University,
Svobodny, 79, Krasnoyarsk, 660041,
Russia

Danil V. Khorov‡
Institute of Space and Information Technologies,
Siberian Federal University,
Academician Kirensky 26к1, Krasnoyarsk, 660074,
Russia

Received 18.05.2009, received in revised form 25.06.2009, accepted 10.07.2009

Stability control of Runge-Kutta numerical schemes is researched for increasing efficiency while integrating stiff problems. The implementation of the algorithm for determining stability polynomials coefficients using the GMP library is given. Shape and size of the stability region of method can be preassigned using presented algorithm. Sets of first-order methods with extended stability domains are built. The results of electrical circuits problems simulation show increase of the efficiency of the constructed first-order methods in comparison with methods of higher order.

Keywords: stiff problem, explicit methods, stability region, accuracy and stability control.

Introduction

Systems of ordinary differential equations (ODEs) describe lots of dynamic processes and arise when solving different problems in the field of chemistry, physics, etc. One of the areas where ODEs may be effectively applied is theory of electric circuit theory. Any changes in electric circuit lead to transient processes where some voltage swells, electromagnetic oscillations, extra currents which may do damage to devices are very likely to appear. At the same time transient processes occur in electrical generators, other electric circuits where they are supposed to be. A large class of electric circuits problems is described with stiff systems of ODEs.

*mixailrybkov@yandex.ru
†lvknaub@yandex.ru
‡danilkhorov@gmail.com

When solving initial value problems of stiff ODEs in some cases explicit methods are required to be applied, because using of L-stable methods needs inversion of Jacobi matrix of the system which defines overall computational costs [1–2]. At the same time explicit methods do not require Jacobi matrix computation and they will be more preferable to use for problems which stiffness ratio is not so high.

At present time algorithms of variable structure which include explicit and implicit methods are built [3]. The former are used on the transition region where the integration step is restricted by accuracy criterion and there is no requirements for the method on having large stability interval. The latter are for the settling regions where great stability interval gives an opportunity to pass the integration interval for "several steps". Nevertheless these algorithms do not show high efficiency on solving high dimension systems of ODEs because of mentioned reasons.

Variable order algorithms based on explicit schemes exclusively where there is no need to use high-order methods on the settling regions are built in [4]. High speed of computations can be achieved by using there low-order methods with extended stability intervals which in fact play part of implicit method from the point of view of the stability interval length.

Low-order methods with large stability interval are needed for constructing such algorithms. In addition the more number of stages of method (and therefore the higher stability polynomial degree m) the bigger the stability interval is. The stability polynomials of degree up to $m = 13$ are constructed in [2]. The algorithm to determine the stability polynomial coefficients is developed such that the corresponding explicit Runge-Kutta methods have a predetermined shape and size of the stability region as shown in [7].

Here implementation of the algorithm of obtaining the stability polynomial coefficients using library for arbitrary precision arithmetic GMP is given. Set of the first-order methods with extended stability intervals is built. Numerical simulation of Van der Pol oscillator showed higher efficiency of proposed algorithms in comparison to Merson method of fourth order of accuracy.

1. Explicit Runge-Kutta Methods

We consider the Cauchy problem for the stiff system of ordinary differential equations

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

where y и f are real N-dimantional vector functions, t is an independent variable. In [2] the authors propose to solve (1) with explicit Runge-Kutta methods

$$y_{n+1} = y_n + \sum_{i=1}^{m} p_{mi} k_i,\ k_i = hf\Big(t_n + \alpha_i h, y_n + \sum_{j=1}^{i-1} \beta_{ij} k_j\Big), \tag{2}$$

where k_i, $1 \le i \le m$, are the stages of the method, h is an integration step, p_{mi}, α_i, β_{ij}, $1 \le i \le m$, $1 \le j \le i-1$, are numerical coefficients that define stability and accuracy characteristics of the scheme (2). For the sake of simplicity further we consider the Cauchy problem for the autonomous system of ordinary differential equations

$$y' = f(y),\ y\big(t_0\big) = y_0,\ t_0 \le t \le t_k, \tag{3}$$

For solving (3) we also may write formulas (2) in the following form:

$$y_{n,i} = y_n + \sum_{j=1}^{i} \beta_{i+1,j} k_j,\ 1 \le i \le m-1, y_{n+1} = y_n + \sum_{i=1}^{m} p_{mi} k_i, \tag{4}$$

where $k_i = hf\Big(y_{n,i-1}\Big), 1 \le i \le m, y_{n,0} = y_n$. The results given below can be used for non-autonomous systems if in (2) we assume

$$\alpha_1 = 0,\ \alpha_i = \sum_{j=1}^{i-1} \beta_{ij},\ 2 \le i \le m. \tag{5}$$

Below we need matrix B_m with elements b_{ij} in the form [2]

$$b_{1i} = 1, 1 \le i \le m, b_{ki} = 0, 2 \le k \le m, 1 \le i \le k-1,$$

$$b_{ki} = \sum_{j=k-1}^{i-1} \beta_{ij} b_{k-1,j}, 2 \le k \le m, k \le i \le m, \tag{6}$$

where β_{ij} are numerical coefficients of the scheme (2) or (4).

The stability of one-step methods is usually investigated by applying a Runge–Kutta method to a linear scalar equation known as Dahlquist's equation

$$y' = \lambda y,\ y\big(0\big) = y_0,\ t \ge 0, \tag{7}$$

with complex λ, $Re\big(\lambda\big) < 0$. The variable λ is considered as a certain eigenvalue of the Jacobi matrix of the problem (1) or (3). Applying numerical scheme (4) to solve the Dahlquist's equation we get

$$y_{n+1} = Q_m(z) y_n,\ z = h\lambda,\ Q_m(z) = 1 + \sum_{i=1}^{m} c_{mi} z^i, c_{mi} = \sum_{j=1}^{m} b_{ij} p_{mj}, 1 \le i \le m. \tag{8}$$

Denoting $C_m = \big(c_{m1}, ..., c_{mm}\big)^T$ и $P_m = \big(p_{m1}, ..., p_{mm}\big)^T$, the latter equality (8) we can rewrite in the form

$$B_m P_m = C_m, \tag{9}$$

where the elements of the matrix B_m are defined in (6). For internal numerical schemes (4) we have

$$y_{n,k} = Q_k\big(z\big) y_n,\ Q_k\big(z\big) = 1 + \sum_{i=1}^{k} c_{ki} z^i,\ c_{ki} = \sum_{j=1}^{k} b_{ij} \beta_{k+1,j},\ 1 \le k \le m-1. \tag{10}$$

Using denotions $\beta_k = \big(\beta_{k+1,1}, ..., \beta_{k+1,k}\big)^T$ и $c_k = \big(c_{k1}, ..., c_{kk}\big)^T$ we get that the coefficients β_{ij} of internal schemes (4) and the coefficients of corresponding stability polynomials are connected with formulas

$$B_k \beta_k = c_k,\ 1 \le k \le m-1. \tag{11}$$

From the comparison of (6) and (10) it follows that $b_{ki} = c_{i-1,k-1}$, i.e. the elements of (k+1)–th column of matrix B_m equal to coefficients of stability polynomial $Q_k\big(z\big)$. Hence, if the coefficients of the stability polynomials of the basic and intermediate numerical schemes are defined, then the coefficients of methods (4) are unambiguously determined from linear systems (9) and (11) with upper triangular matrices $B_i, 1 \le i \le m$.

Expansions of the exact and approximate solutions in the Taylor series in powers of h have the form

$$y(t_{n+1}) = y(t_n) + hf + \frac{h^2}{2} f'f + O(h^3),$$

$$y_{n+1} = y_n + \Big(\sum_{j=1}^{m} b_{1j} p_{mj}\Big) hf + \Big(\sum_{j=2}^{m} b_{2j} p_{mj}\Big) h^2 f'_n f_n + O(h^3), \tag{12}$$

where the elementary differentials are computed on the exact $y(t_n)$ and approximate y_n solutions, respectively. Comparison between relations (12) under assumption that $y(t_n) = y_n$ shows that numerical formula (4) has the first order of accuracy, if $\sum\limits_{j=1}^{m} b_{1j} p_{mj} = 1$. Hence, to design m-stage methods of the first accuracy order, it is necessary to set $c_{m1} = 1$.

2. Stability Polynomials

Let two integer number k и m, $k \le m$ be given. Consider the polynomial

$$Q_{m,k}(x) = 1 + \sum_{i=1}^{k} c_i x^i + \sum_{i=k+1}^{m} c_i x^i, \tag{13}$$

where the coefficients c_i, $1 \le i \le k$, are given and c_i, $k+1 \le i \le m$, are free. The coefficients c_i, $1 \le i \le k$, are usually defined from the approximation requirements. Therefore, for definiteness we assume $c_i = 1/i!$, $1 \le i \le k$ below.

Denote extremum points of (13) as $x_1, \ldots, x_{m-1}$, where $x_1 > x_2 > \cdots > x_{m-1}$. Unknown coefficients $c_i, k+1 \le i \le m$, can be obtained from the condition that the polynomial (13) takes on predefined values at extremum points $x_i, k \le i \le m-1$, i.e.

$$Q_{m,k}(x_i) = F_i,\ k \le i \le m-1, \tag{14}$$

where $F(x)$ is a preassigned function, $F_i = F(x_i)$. For this purpose consider the algebraic system of equations in variables $x_i, k \le i \le m-1$, and $c_j, k+1 \le j \le m$,

$$Q_{m,k}(x_i) = F_i,\ Q'_{m,k}(x_i) = 0,\ k \le i \le m-1,\ Q'_{m,k} = \sum_{i=1}^{m} i c_i x^{i-1}. \tag{15}$$

We rewrite (15) in the form that is convenient for computations. Let y, z, g и r denote vectors with components

$$y_i = x_{k+i-1}, z_i = c_{k+i}, g_i = F_{k+i-1} - 1 - \sum_{j=1}^{k} c_j y_i^j,$$

$$r_i = -\sum_{j=1}^{k} j c_j y_i^{j-1},\ 1 \le i \le m-k, \tag{16}$$

Let $E_1, ..., E_5$ denote diagonal matrices with elements on the mail diagonal in the form

$$e_1^{ii} = k + i,\ e_2^{ii} = 1/y_i,\ e_3^{ii} = \sum_{j=1}^{k} j c_j y_i^{j-1} + \sum_{j=1}^{m-k} (k+j) z_j y_i^{k+j-1},$$

$$e_4^{ii} = \sum_{j=2}^{k} j(j-1)c_j y_i^{j-2} + \sum_{j=1}^{m-k} (k+j)(k+j-1)z_j y_i^{k+j-2}, \tag{17}$$

$$e_5^{ii} = (-1)^{k+i-1},\ 1 \le i \le m-k,$$

Let A denote matrix $a^{ij} = y_i^{k+j}, 1 \le i, j \le m-k$. The elements of vectors (16), matrices (17) and A depend on numbers m and k, where

$$g = g\big(y\big),\ r = r\big(y\big),\ E_2 = E_2\big(y\big),\ E_3 = E_3\big(y,z\big),\ E_4 = E_4\big(y,z\big),\ A = A\big(y\big).$$

Then, we can rewrite the problem (15) in the form

$$Az - g = 0,\ E_2 A E_1 z - r = 0. \tag{18}$$

System (18) is ill-conditioned that leads to some difficulties on applying for its solution the fixed-point iterations. For convergence of the Newton's method it is necessary to obtain somehow good initial values that in this case is a separate difficult problem.

If we assume in (15) that $F_i = (-1)^i, k \le i \le m-1$, we find the polynomial with the maximal length of stability interval. In this case the problem of computation of initial value y_0 is solved using values of the Chebyshev polynomial at extremum points over interval $[-2m^2, 0]$, where m is the degree of polynomial (13). That values can be computed using the formula

$$y_i = m^2[cos(i\pi/m) - 1],\ 1 \le i \le m-1. \tag{19}$$

Substituting (19) in the system (17), we get coefficients of the Chebyshev polynomial, for which $\big|Q_{m1}\big(x\big)\big| \le 1$ on $x \in [-2m^2, 0]$. For any k (19) can be taken as the initial values and according to numerical computations there is good convergence rate in this case. If $F_i \ne (-1)^i, k \le i \le m-1$, then the choice of initial values is quite a difficult problem.

Let us describe a way to solve (18) that does not require good initial values. Apply the relaxations for the numerical solution of (18). The main idea of the relaxations is that for steady-state problem we run unsteady-state process which solution settles to the solution of the initial problem. Consider the Cauchy problem

$$y' = E_5\big(E_2 A E_1 A^{-1} g - r\big),\ y\big(0\big) = y_0. \tag{20}$$

Apparently, after the determination of stationary point of (20) the coefficients of a stability polynomial can be computed from the system (18). Notice, that due to using matrix E_5 all the eigenvalues of the Jacobi matrix of (20) have negative real components, i.e. problem (7) is stable. From the numerical results it follows that (20) is a stiff problem. Methods for solving such problems use calculation of the Jacobi matrix which cause difficulties on solving (20). Therefore, let us apply the second accuracy order method using numerical calculation and freezing the Jacobi matrix [5–6] to solve (20).

With some work it can be shown that with m moving higher the polynomial coefficients tend to zero. Coefficients $c_i, k+1 \le i \le m$, were computed using algorithm [2] up to the polynomial degree $m = 13$. Moreover, algorithm of obtaining polynomial coefficients on the interval $[-1, 1]$ is described in [7]. In this case coefficients c_i grow with slower speed and it is possible to build polynomials of degrees $m > 13$.

3. The implementation of the algorithm for obtaining stability polynomials coefficients using the GMP library

It is not difficult to see that the coefficient c_m of stability polynomial (13) tends to zero as m increases and in particular if $m = 13$ and $k = 1$ the value of c_m is about 10^{-26}. Computation of problem (20) where $m > 13$ with double precision is very hard to do because of round-off errors. In order to compute polynomial coefficients of higher degrees m in [8] algorithm was implemented using *qd* library that is described in [9].

The *qd* library allows to perform computations with higher accuracy. Standard data type *double* which allows to perform computations with double precision is restricted by 53 bites of binary mantissa and provides accuracy of 16 decimal digits, whereas *qd* data type dd_real has 106–bit mantissa that provides accuracy of 32 decimal digits. In fact, the number of data type dd_real is a programmed concatenation of two *double* numbers, where mantissa becomes doubled, but the range of values that can be represented using new data type stays the same (from 10^{-308} to 10^{308}). Despite this restriction accuracy of representing number increases.

With the use of this library the coefficients of polynomials up to degree $m = 35$ were computed in [8]. Nevertheless, the *qd* library has some disadvantages. Firstly, accuracy of representing numbers is restricted because of program implementation of data types. Secondly, it can be used only in Unix systems. Moreover, the *qd* library is written in the $C++$ programming language. That is why algorithms using this library could be slower than the ones realised in low-level programming languages (for example, C).

Here we show numerical results of implementation of the algorithm of obtaining polynomial coefficients with help of the library for arbitrary precision arithmetic GMP. This library provide accuracy of computations that is restricted only by the random access memory size. It is cross-platform and support operations on integer, rational and real numbers. Besides, the GMP library is written in the C programming language, which potentially increase the speed of computations.

Using the GMP library we managed to compute the polynomial coefficients up to degree $m = 40$. At higher degrees there are some difficulties that may be related to the initial conditions choice for the problem (20).

4. Stability Regions Construction

Let us now describe the effect of the function F on the size and shape of the stability region. If we assume $F_i = (-1)^i$, $k \le i \le m-1$, than the stability interval length is known to be $|\gamma_m| = 2m^2$. In this case, we get the maximum stability interval length along the real axis for given m. The stability region of such methods is almost multiply connected which leads to reducing of stability interval length because of some potential rounding errors provoking appearing of small imaginary parts of Jacobi matrix eigenvalues. Figure 1 shows the stability region of 5-stage method, where the stability interval length is $|\gamma_m| = 50$.

In order to avoid stability region reducing because of rounding errors it should be "stretched" along the imaginary axis at the extremum points of stability polynomial. For that we can assume $F_i = (-1)^i\mu, 1 \le i \le m-1, 0 < \mu < 1$. For example, if we choose $\mu = 0.95$, then the stability interval length will be reduced by only $3-4\%$ in comparison with maximal possible

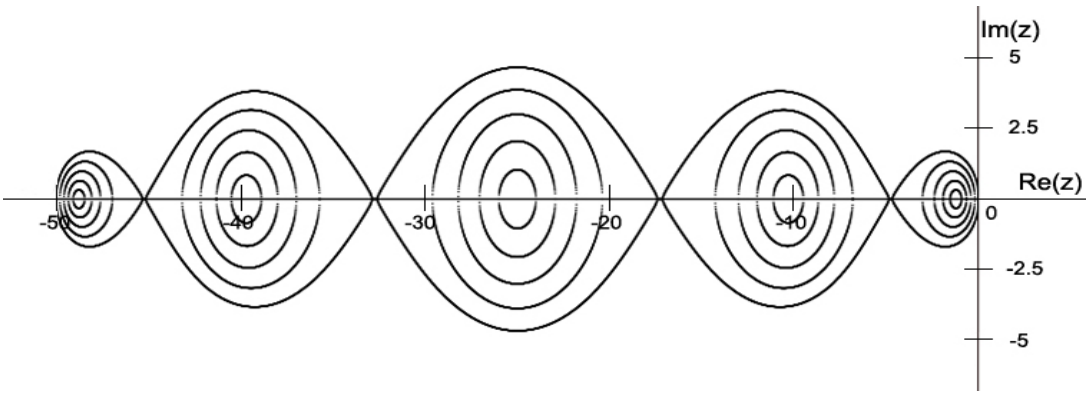

Figure 1: The stability region at $m = 5$, $k = 1$, $F = \{-1, 1, -1, 1\}$, $\left|\gamma_m\right| = 50$

length that equals $2m^2$ and it will be $\left|\gamma_m\right| = 48.39$ (fig. 2). The stability region of the 5–stage method at $\mu = 0.8$ is shown in figure 3. In this situation, the stability interval length is reduced to $\left|\gamma_m\right| = 43.55$ with conjoined "stretching" along the imaginary axis. For better visualization of the polynomial roots (13) in the complex plane all figures provide level lines $\left|Q_{m,k}(x)\right| = 1$, $\left|Q_{m,k}(x)\right| = 0.8$, $\left|Q_{m,k}(x)\right| = 0.6$, $\left|Q_{m,k}(x)\right| = 0.4$, $\left|Q_{m,k}(x)\right| = 0.2$.

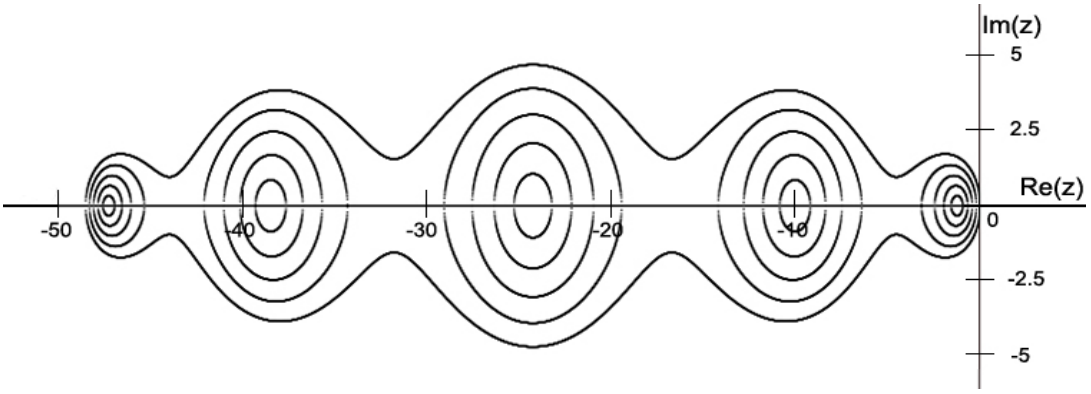

Figure 2: The stability region at $m = 5$, $k = 1$, $F = \{-0.95, 0.95, 0.95, 0.95\}$, $\left|\gamma_m\right| = 48.39$

5. First-Order Method

For numerical solution of Cauchy problem (1) we consider the explicit five-stage Runge-Kutta method

$$\begin{gathered} y_{n+1} = y_n + p_1 k_1 + p_2 k_2 + p_3 k_3 + p_4 k_4 + p_5 k_5, \\ k_1 = hf\big(y_n\big),\ k_2 = hf\big(y_n + \beta_{21} k_1\big), \\ k_3 = hf\big(y_n + \beta_{31} k_1 + \beta_{32} k_2\big), \end{gathered} \tag{21}$$

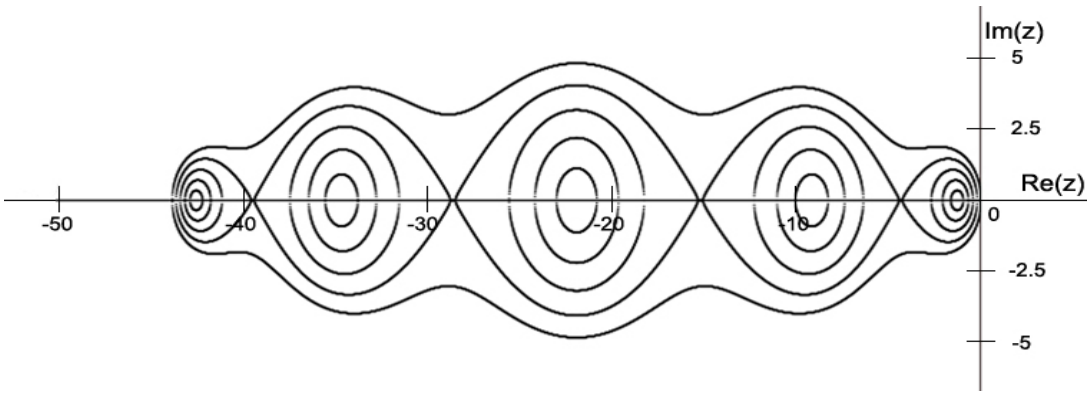

Figure 3: The stability region at $m = 5$, $k = 1$, $F = \{-0.8, 0.8, -0.8, 0.8\}$, $\left|\gamma_m\right| = 43.55$

$$k_4 = hf\big(y_n + \beta_{41}k_1 + \beta_{42}k_2 + \beta_{43}k_3\big),$$
$$k_5 = hf\big(y_n + \beta_{51}k_1 + \beta_{52}k_2 + \beta_{53}k_3 + \beta_{54}k_4\big),$$

where y and f are real N-dimensional vector function, t is an independent variable, h is the integration step, k_1, k_2, k_3, k_4 and k_5 are stages of the method, $p_1, p_2, p_3, p_4, p_5, \beta_{21}, \beta_{31}, \beta_{32}, \beta_{41}, \beta_{42}, \beta_{43}, \beta_{51}, \beta_{52}, \beta_{53}, \beta_{54}$ are numerical coefficients, defining accuracy and stability properties of (21). Applying the algorithm, we get stability polynomials coefficients:

$$c_{5,1} = 0.1e1,\ c_{5,2} = 0.164341322127140896342e0,\ c_{5,3} = 0.948975952580473808808e - 2,$$

$$c_{5,4} = 0.223956930863224544258e - 3,\ c_{5,5} = 0.185097275222353341 53e - 5$$

In this case the stability interval length is $\left|\gamma_m\right| = 48.39$. Writing and resolving (9) and (11), we obtain the coefficients of the first-order method:

$$\beta_{21} = 0.0413243016210550,\ \beta_{31} = 0.0805823881610573,$$

$$\beta_{32} = 0.0805823881610573,\ \beta_{41} = 0.11916681511228434,$$

$$\beta_{42} = 0.1597820013984078,\ \beta_{43} = 0.0819394878966193,$$

$$\beta_{51} = 0.1570787892802991,\ \beta_{52} = 0.2379583021959820,$$

$$\beta_{53} = 0.1631711307360486,\ \beta_{54} = 0.0822916178203657,$$

$$p_1 = 0.1945277188657676,\ p_2 = 0.3151822878089125,$$

$$p_3 = 0.2437005934695969,\ p_4 = 0.1641555613805598,$$

$$p_5 = 0.0824338384751631.$$

Without formula output we reduce formulas for accuracy control of numerical scheme, based on local error estimation and described in [10].

As tentative assessment of local error we take the value

$$A'_n = [(0.5 - c_{m2})/\alpha_2](k_2 - k_1). \tag{22}$$

We make the final decision on accuracy by estimating

$$A''_n = (0.5 - c_{m2})(hf\big(y_{n+1}\big) - k_1). \tag{23}$$

Thus, we apply the following inequalities for accuracy control and as criterion of integration step choice

$$A'_n \le \epsilon,\ A''_n \le \epsilon. \tag{24}$$

As k_1 linearly depends on integration stepsize, then omission of inequality (24) leads just to one additional computation of the right part of the problem. If the integration on step is successful, the second inequality (24) does not lead to increase of computational costs, because $f\big(y_{n+1}\big)$ is not used at the next step. At the same time if the second inequality (24) is used for accuracy control, the backout in case of violating accuracy criterion is quite expensive, moreover the more m the higher computational costs are. Nevertheless, in most cases tentative assessment of A'_n allows to avoid recomputations of solution. We use the following inequality for stability control of methods (2)

$$\nu_n \le \gamma_{m,1}, \tag{25}$$

where

$$\nu_n = \Big|\alpha_2\beta_{32}\Big|^{-1} \max_{1\le j\le N}\Big|[\alpha_2 k_3 + \alpha_3 k_2 - (\alpha_2+\alpha_3)k_1]_j/[k_2 - k_1]_j\Big|, \tag{26}$$

and positive constants $\gamma_{m,1}$ define the size of stability regions [10].

6. Merson Method

One of the most effective explicit fourth-order Runge-Kutta type methods is the Merson method [8]

$$\begin{gathered} y_{n+1} = y_n + \frac{1}{6}k_1 + \frac{2}{3}k_4 + \frac{1}{6}k_5, \\ k_1 = hf\big(y_n\big),\ k_2 = hf\big(y_n + \frac{1}{3}k_1\big), \\ k_3 = hf\big(y_n + \frac{1}{6}k_1 + \frac{1}{6}k_2\big), \\ k_4 = hf\big(y_n + \frac{1}{8}k_1 + \frac{3}{8}k_3\big), \\ k_5 = hf\big(y_n + \frac{1}{2}k_1 - \frac{3}{2}k_3 + 2k_4\big). \end{gathered} \tag{27}$$

The fifth computation of function f does not result in the fifth order of accuracy, but allows to extend the stability interval to 3.5 and estimate truncation error $\delta_{n,4}$ using stages k_i i.e.

$$\delta_{n,4} = (2k_1 - 9k_3 + 8k_4 - 2k_5)/30.$$

We apply inequality $\big\|\delta_{n,4}\big\| \le 5\epsilon^{5/4}$ for accuracy control. Despite the fact that the inequality for accuracy control is obtained on a linear equation, it shows high reliability on solving non-linear problems.

Now let us construct the inequality for stability control. Applying to $k_3 - k_2$ the first order Taylor's formula with the remainder term written in the Lagrangian form, we have

$$k_3 - k_2 = h[\partial f\big(\mu_n\big)/\partial y](k_2 - k_1)/6,$$

where vector μ_n is computed in some vicinity of solution $y\big(t_n\big)$. Taking into account, that

$$k_2 - k_1 = h^2 f'_n f_n/3 + O\big(h^3\big),$$

the inequality

$$\nu_{n,4} = 6 \cdot \max_{1\le j\le N} \Big| \frac{k_3^j - k_2^j}{k_2^j - k_1^j} \Big| \le 3.5, \tag{28}$$

can be used for stability control of (27), where 3.5 is the approximate length of stability interval. Let $\epsilon_{n,4} = \delta_{n,4}/5$. Then inequalities $\epsilon_{n,4} \le 5\epsilon^{5/4}$ and $\nu_{n,4} \le 3.5$. can be applied respectively for accuracy and stability control of scheme (27).

7. Numerical Results

The calculations were performed on Intel(R) Core(TM) i7-8550U CPU. However, stability polynomial coefficients were computed with the help of the GMP library, whereas calculation of differential problem solution was implemented with double precision. In concrete computations the norm $\|\xi_n\|$ from the inequalities for the accuracy control was calculated by formula

$$\big\|\xi_n\big\| = \max_{1\le j\le N} |\xi_n^i|/(|y_n^i| + r),$$

where i is a number of vector component, r is a positive parameter. If inequality $|y_n^i| < r$ holds on the component with number i, then the absolute error $r \cdot \varepsilon$ is controlled, otherwise we control the relative error ε, where ϵ is the required accuracy.

We chose the Van der Pol oscillator (29) as a test example. This problem has the stiffness ratio approximately equal to 10^6:

$$y'_1 = y_2,\ y'_2 = \Big((1 - y_1^2)y_2 - y_1\Big)/10^{-6}$$

$$0 \le t \le 1,\ h_0 = 10^{-3},\ y_1\big(0\big) = 2,\ y_2\big(0\big) = 0, \epsilon = 10^{-2}. \tag{29}$$

We compared the efficiency of two algorithms. The first one is the first order 5-stage Runge-Kutta method described in section 5. And the second one is the traditional 5-stage Merson method of the forth order of accuracy (27). Either of two algorithms were run in two modes: with stability control and without it. We counted total numbers of steps, recomputations of a solution (due to omission of the defined accuracy), and number computations of right parts of a differential problem. When using Merson method for computing the problem the defined accuracy $\epsilon = 10^{-2}$ was supported, whereas for the first-order method we needed to define $\epsilon = 10^{-5}$ in order to provide 10^{-2} in fact. Nevertheless, under these conditions the constructed algorithm shows better efficiency (fig. 4).

The comparison of two algorithms shows that stability control provides efficiency increasing because of eliminating extra recomputations of solution originating from instability of numerical scheme. In addition, the constructed first-order algorithm has less computational costs estimated by number of right parts computations. Simulation of other test examples gives similar tendency.

Criterion	First-order method without stability control	First-order method with stability control	Merson method without stability control	Merson method with stability control
Number of integration steps	69 433	51 414	549 241	556 114
Number of repeated solution calculations	20 001	1 052	187 120	6 464
Number of right part computations	452 683	309 948	3 494 685	2 806 426

Figure 4: Numerical results of Van der Pol problem simulation

Conclusion

Implementation of the algorithm of obtaining stability polynomial coefficients using the GMP library allowed to build stability polynomial up to degree $m = 40$, which made possible to build methods with extended stability regions with respective number of stages. The more number of stages, the larger stability interval is and therefore the higher efficiency of numerical scheme is shown when integrating stiff problems.

When comparing two five-stage methods (the constructed first-order method and Merson method), we see that at the same number of stages extending of stability interval gives decreasing of overall computational costs.

It is important to say that the first-order methods with extended stability regions allow to significantly increase the efficiency in a settling region where the step is restricted by stability. So methods described here can be used in adaptive algorithms where number of stages may vary from one integration step to another providing large stability interval where needed and saving computational efforts where there is no requirement to stability characteristics of numerical scheme.

The reported study was funded by RFBR according to the research project № 18-31-00375.

References

[1] E. Hairer and G. Wanner, Solving ordinary differential equations. II. Stiff and differential-algebraic problems, Berlin: Springer, 1996.

[2] E.A. Novikov, Explicit methods for stiff systems, Novosibirsk: Nauka, 1997 (in Russian).

[3] Novikov E.A., Novikov A.E. *Explicit-Implicit Variable Structure Algorithm for Solving Stiff Systems,* International Journal of Mathematical Models and Methods in Applied Sciences – 2015. Vol.9(1)–pp.62–70

[4] E.A. Novikov, Yu.V. Shornikov, Computer simulation of stiff hybrid systems, Novosibirsk: Publisher of NSTU, 2012 (in Russian).

[5] A.E. Novikov, E.A. Novikov, L-stable (2,1)-method for stiff nonautonomius problem solving, *Computing technologies*, **13**(2008), 477–482 (in Russian).

[6] E.A. Novikov, Yu.A. Shitov, *Integration algorithm for stiff systems based on a second-order accuracy (m, k)–method with numerical calculation of the Jacobi matrix,* Krasnoyarsk:

Preprint of the Exhibition Center of the Siberian Branch of the USSR Academy of Sciences, n.20. – 1988. – 23 p. (in Russian).

[7] E.A. Novikov, M.V. Rybkov, *The numerical algorithm of constructing stability polynomials of first order methods,* Bulletin of the Buryat State University, (2014), no.9–2, 80–85. (in Russian).

[8] E.A. Novikov, M.V. Rybkov, The numerical algorithm of constructing of stability regions for explicit methods, *Control systems and information technologies*, (2014), no.1.1(55), 173–177 (in Russian).

[9] Yozo Hida, Xiaoye S Li, David H Bailey, *Quad-double arithmetic: algorithms, implementation, and application,* Technical Report LBNL–46996, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, 2000. – 28 p.

[10] Knaub L.V., Litvinov P.S., Novikov A.E., Rybkov M.V. *Solving Problems of Moderate Stiffness Using Methods of the First Order with Conformed Stability Domains*, University Scientific Journal, (2016), no.22, – p. 49–58.

[11] Merson R.H. *An operational methods for integration processes,* Proc. of Symp. on Data Processing. Weapons Research Establishment, Salisbury, Australia. 1957. P. 331.

Методы первого порядка с расширенными областями устойчивости для расчета задач электрических цепей

Михаил В. Рыбков, Людмила В. Кнауб, Данил В. Хоров

Исследуется применение контроля устойчивости численных схем типа Рунге-Кутты для повышения эффективности при интегрировании жестких задач. Приведена реализация алгоритма определения коэффициентов полиномов устойчивости, при которых метод имеет заданную форму и размер области устойчивости, с помощью библиотеки GMP. Построены наборы методов первого порядка с расширенными областями устойчивости. Приведены результаты расчетов задач из теории электрических цепей, показывающие повышение эффективности построенных методов первого порядка точности в сравнении с методом более высокого порядка.

Ключевые слова: жесткая задача, явные методы, контроль точности, контроль устойчивости