Another Approach to Runge-Kutta Methods

C.R.Traas
University of Twente, P.O.Box 217
7500AE, Enschede, The Netherlands.
c.r.traas@utwente.nl

Abstract

The condition equations are derived by the introduction of a system of *equivalent differential equations*, avoiding the usual formalism with trees and elementary differentials.
Solutions to the condition equations are found by direct numerical optimization, during which simplifying assumptions upon the Runge-Kutta coefficients may or may not be used. Depending on the optimization criterion, different types of optimal Runge-Kutta methods can be pursued. In the present article the emphasis is on rounding minimization.

Key words: Runge-Kutta, condition equations, direct optimization, rounding errors
AMS Subject Classification: 65L06

1 Introduction

Deriving Runge-Kutta methods of high order is a heavy task. The standard way for deriving them is based on expansion of the formal system of differential equations on the one hand, and expansion of one Runge-Kutta step on the other hand, matching equal powers of the stepsize. The expansion of the formal system of differential equations leads to a complicated set of equations, involving all partial derivatives, up to a certain order, of the right-hand members of the differential equations, which process has been treated in literature in a systematic way by the introduction of *trees* and *elementary differentials* [B63, HNW93]. One aim of the present work is to avoid this formalism and to present a simpler way to arrive at the condition equations.
Although the process of deriving Runge-Kutta methods can be considered, more or less, as being completed, it does not hurt to consider alternatives, which can be instructive and interesting in itself. In addition, the proposed approach may be of interest to those who work on other types of continuous problems such as delay differential equations, integral equations and, possibly, partial differential equations. The presentation of this approach, therefore, is also an aim of the present work.
The process for deriving Runge-Kutta methods, from literature, can be simplified, indeed, by deriving a system of *equivalent* differential equations. Using formula manipulation software (MAPLE) the condition equations for the coefficients of a Runge-Kutta method then follow easily.

Simplifying assumptions upon the coefficients are usually introduced for reducing the complexity of solving the set of condition equations, and/or for generating a specific, desirable structure in the Runge-Kutta method. Examples are the methods of Fehlberg, Verner and Dormand-Prince [HNW93]. In the present article solutions to the condition equations are obtained by *direct* numerical optimization. Simplifying assumptions will lighten this numerical process, but they are not really needed to obtain a solution. The optimization criterion determines in what sense the result will be optimal.
A further aim, therefore, is to find, by direct optimization, Runge-Kutta methods with specific optimality properties, which are specified a priori. In the present work the emphasis is on better rounding error behaviour. This has resulted in the generation of a 5th order Runge-Kutta method with embedded 4th order method, valid for general systems of differential equations, using just a plain 2.3 GHz Personal Computer. Also a number of solutions without embedding have been obtained, and a method of order 6, though the latter only for a scalar autonomous differential equation. For higher order methods for general systems of differential equations, however, more, to much more, computing power is required because the numerical optimization process requires much computing effort.
Concerning the point of *deriving* the order conditions, avoiding trees and elementary differentials, another alternative can be found in [A87]. Here RK-methods are treated as composite linear multistep schemes and this article, therefore, contributes to the efforts towards a unified treatment of numerical methods for solving ordinary differential equations.

2 The condition equations.

Let be given the autonomous system of differential equations

$$\begin{aligned} \frac{dy_1}{dx} &= f_1(y_1, y_2) \\ \frac{dy_2}{dx} &= f_2(y_1, y_2), \end{aligned} \tag{1}$$

where $\mathbf{f} = (f_1, f_2)$ is supposed to be sufficiently differentiable. We here consider a system of *two* autonomous differential equations. This gives no loss of generality for deriving Runge-Kutta methods up to a certain order which includes, anyway, the fifth order. For higher order methods the necessity of extending the system (1) with one or more equations should be considered.

2.1 Deriving the equations.

Introducing the notation $x - x_0 = X,\ y_i - y_{0,i} = Y_i,\ i = 1, 2$, the system (1) can be rewritten as

$$\frac{dY_i}{dX} = F_i(Y_1, Y_2),\ i = 1, 2. \tag{2}$$

Expanding formally around $X = 0$ we get

$$\frac{dY_i}{dX} = \mathcal{C}_{i00} + \mathcal{C}_{i10}Y_1 + \mathcal{C}_{i01}Y_2 + \mathcal{C}_{i20}Y_1^2 + \ldots, \ i = 1, 2. \tag{3}$$

We use the system (3) as a model for the system (2), valid over the interval $[0, h]$ (equivalent differential equations), where h is a typical stepsize for numerical integration. The aim is to give the coefficients $\mathcal{C}_{ijk}$ values such that, in the limit of infinite series, the solution of (3) over the interval $[0, h]$ would be formed exactly by (collocation) polynomials of degree n,

$$Y_i = \sum_{j=1}^{n} a_{ij} X^j, \ i = 1, 2, \tag{4}$$

which are associated with the n-th order Runge-Kutta method.
Using a computer algebra package (MAPLE), it is straightforward to invert these polynomials:

$$X = \sum_{j=1}^{N>n} c_{ij} Y_i^j, \ i = 1 \text{ or } 2, \tag{5}$$

with

$$\begin{aligned}
c_{i1} &= 1/a_{i1} \\
c_{i2} &= -a_{i2}/a_{i1}^3 \\
c_{i3} &= (2a_{i2}^2 - a_{i1}a_{i3})/a_{i1}^5 \\
c_{i4} &= (5a_{i3}a_{i1}a_{i2} - 5a_{i2}^3 - a_{i1}^2 a_{i4})/a_{i1}^7 \\
c_{i5} &= \ldots \text{ etc.}
\end{aligned}$$

Now

$$\frac{dY_i}{dX} = \sum_{j=1}^{n} j a_{ij} X^{j-1}, \ i = 1, 2. \tag{6}$$

Defining

$$X = \alpha \sum_{j=1}^{N} c_{1j} Y_1^j + (1 - \alpha) \sum_{j=1}^{N} c_{2j} Y_2^j, \ \alpha \in \mathbb{R}, \tag{7}$$

then, upon comparing (6) with (3), we find

$$\begin{aligned}
\mathcal{C}_{i00} &= a_{i1} \\
\mathcal{C}_{i10} &= 2\alpha a_{i2}/a_{11} \\
\mathcal{C}_{i01} &= 2(1-\alpha)a_{i2}/a_{21} \\
\mathcal{C}_{i20} &= \alpha(3\alpha a_{i3}a_{11} - 2a_{12}a_{i2})/a_{11}^3 \\
\mathcal{C}_{i11} &= 6\alpha(1-\alpha)a_{i3}/(a_{21}a_{11}) \\
\mathcal{C}_{i02} &= (1-\alpha)(3(1-\alpha)a_{i3}a_{21} - 2a_{22}a_{i2})/a_{21}^3 \\
\mathcal{C}_{i30} &= \ldots \text{ etc.}
\end{aligned} \tag{8}$$

Remark. In the case of m equations (1) the definition of X should be

$$X = \sum_{i=1}^{m} \alpha_i \sum_{j=1}^{N} c_{ij} Y_i^j, \ \alpha_i \in \mathbb{R}, \text{ with } \sum \alpha_i = 1. \qquad \blacksquare$$

In the explicit method of Runge-Kutta for (2) we have

$$k_{ij} = F_i(h \sum_{\ell=1}^{j-1} A_{j\ell} k_{1\ell}, \ h \sum_{\ell=1}^{j-1} A_{j\ell} k_{2\ell}), \tag{9}$$

with $j = 2, \ldots, s, \quad k_{i1} = F_i(0, 0)$, and

$$Y_i = h \sum_{j=1}^{s} b_j k_{ij}, \ i = 1, 2, \tag{10}$$

where s is the stage number and (A_{ij}), (b_i) are the Runge-Kutta coefficients.
Using the computer algebra package the expressions (10), with (9) inserted and F_i replaced by the righthand member of (3) with coefficients (8), can be Taylor-expanded in powers of h. Matching equal powers of h between these expressions and (4) with X replaced by h, we get a system of condition equations:

$$M\mathbf{b} = \mathbf{g}, \tag{11}$$

where the matrix M depends on the coefficients A_{ij}, and $\mathbf{b} = (b_1, \ldots, b_s)^T$.
We are also interested in an embedded method of order $n - 1$, useful for stepsize control. The function evaluation for the last stage of the embedded method should be the same as the one for the first stage of the n-th order method for the next step. The reason for this choice is the saving, effectively, of one function evaluation per step. The system of condition equations for the embedded method is written as

$$\hat{M}\hat{\mathbf{b}} = \hat{\mathbf{g}}. \tag{12}$$

The matrix $\hat{M}$ only contains rows that correspond with the Taylor expansions up to and including h^{n-1}, and $\hat{\mathbf{b}}$ is the vector $(\hat{b}_1, \ldots, \hat{b}_{s+1})^T$.
Let $\mathbf{b}$ be a solution to (11). Then

$$\hat{\mathbf{b}}_0 = (\mathbf{b}^T \ \ 0)^T. \tag{13}$$

is a solution to (12). The matrix $\hat{M}$ has fewer rows than M, and it has (in this case) one more column. For a true embedded method a solution different from (13) is needed i.e., $\hat{M}$ should have rank deficiency.

We illustrate the above process for fifth order integration. Then $n = 5$ in (4). We take $N = 6$, and the stagenumber $s = 6$.
Introducing the notation

$$C_j = \sum_{i=1}^{j-1} A_{ji}, \ j = 2, \ldots, s, \tag{14}$$

we find

$$Y_1(h) = ha_{11} \sum_{i=1}^{6} b_i + 2h^2 a_{12} \sum_{i=2}^{6} b_i C_i + \ldots + \mathcal{O}(h^7) \tag{15}$$

and similar for $Y_2(h)$. In this expansion of $Y_i(h)$ the general term, above the one of order h, has the form

$$h^p \sum_k \beta_k^{(p)}(b_2, \ldots, b_6, A_{21}, \ldots, A_{65}) \frac{\prod_{j=2}^{n} a_{1j}^{r_{kj}} \prod_{\ell=2}^{n} a_{2\ell}^{s_{kj}}}{a_{11}^{p_k} a_{21}^{q_k}}, \quad p_k, q_k, r_{kj}, s_{kj} \in \mathbb{N}_0, \tag{16}$$

where $\beta_k^{(p)}$, p_k, q_k, r_{kj}, s_{kj} depend on p and i. For the purpose of illustration we write out the term for $p = 3$ in the expansion of $Y_1(h)$:

$$h^3 \Big(a_{13} \sum_{i=2}^{6} r_i^{(1)} b_i + a_{12}^2 \sum_{i=2}^{6} r_i^{(2)} b_i + a_{12} a_{22} \sum_{i=2}^{6} r_i^{(3)} b_i \Big),$$

where the $r_i^{(j)}$ depend on the $A_{\ell m}$ and on p. We have put $a_{11} = a_{21} = 1$, which gives no loss of generality. In the notation of (16) we thus have:

$$\beta_k^{(3)} = \sum_{i=2}^{6} r_i^{(k)} b_i.$$

In the case of $p = 4$, in the expansion of $Y_1(h)$, the following combinations $\prod_j a_{1j}^{r_{kj}} \prod_\ell a_{2\ell}^{s_{kj}}$ occur:

$$a_{14}, \ a_{12}a_{13}, \ a_{12}a_{23}, \ a_{13}a_{22}, \ a_{12}^3, \ a_{12}^2 a_{22}, \ a_{12}a_{22}^2.$$

The expansion (15) should match (4), with h as value of X, for all real values of a_{1j} and $a_{2\ell}$, $j = 2, \ldots, n$, $\ell = 2, \ldots, n$. Therefore the condition equations follow by putting, for each h^p, $p = 2, \ldots, 5$, the $\beta_k^{(p)}$ equal to 1 or 0, namely 1 if $\prod_j a_{1j}^{r_{kj}} \prod_\ell a_{2\ell}^{s_{kj}} \equiv a_{ip}$, $i = 1, 2$, and 0 otherwise. For each p, there is only one $\beta_k^{(p)}$ in $Y_i(h)$ that has to be put to 1, because $a_{ip}X^p$ is the only term of order p in (4).
In the following we will fix $C_6 = 1$. This is a usual choice for a Runge-Kutta method with an embedded method.

In the expansion of $Y_i(h)$, the coefficient of h^3 contains 3 terms $\prod a_{1j}^{r_{kj}} \prod a_{2\ell}^{s_{kj}}$, that of h^4 contains 7 terms and that of h^5 15 terms. So for Y_1 and Y_2 together we get 54 condition equations, of which many occur more than once.
By analysis of the matrix M we find that from the 6 equations corresponding to the h^3 part in the expansion of Y_1 and Y_2 only 2 are independent. In the h^4 part only 4 are independent and in the h^5 part only 9, the latter provided that α in (7) is chosen $\neq \frac{1}{2},\ 0$ or 1, so that from the total of 54 conditions only 17 independent relations remain.
Such an independent set of condition equations out of the total system is obtained by comparing all of them mutually. To start with, first all sets of *two* equations are considered. This eliminates already a lot of equations. Of the remaining system, all sets of *three* equations are tested, etc. This concludes with the testing of all sets of five equations (in this case) from a, in the mean time, considerably reduced system of equations. The remaining system is a system of independent equations sought after. This process is suitable for automation.

Remark. For the choice $\alpha = \frac{1}{2}$ we get a system of 16 independent relations, which corresponds with the situation of a scalar, non-autonomous differential equation. Apparently, this is a consequence of the symmetry in the system for this choice of α. The choices $\alpha = 0$ and $\alpha = 1$ correspond with the situation of a scalar, autonomous differential equation.
In the case of m equations (1) the following inequalities seem to be safe:

$$\alpha_i \neq \alpha_j \text{ for } i \neq j, \quad \alpha_i \neq 0, 1, \quad i, j = 1, \ldots, m.$$

2.2 Necessary and Sufficient Set of Condition Equations.

It is known that a Runge-Kutta method of order up to and including the 4^{th}, derived for a scalar, non-autonomous DE, is also a 4^{th}-order method for a *system* of differential equations [Lam91]. This is not true for 5^{th} and higher order methods. From the analysis given above it appears that a Runge-Kutta method of order 5, derived for a system of 2 autonomous DE's (using $\alpha \neq \frac{1}{2}, 0, 1$), is also a 5^{th} order method for a general system of DE's. This is true because a set of 17 independent condition equations is found which is known, from the classical approach to RK-methods, to be the necessary and sufficient set for getting a general 5^{th} order method. This is confirmed within the present context by considering the matrix M corresponding with a set of 3 equations (1), which does *not* lead to more condition equations.

3 Computing the Runge-Kutta coefficients.

The systems of condition equations are written as

$$M\mathbf{b} = \mathbf{g}, \tag{17}$$

and

$$\hat{M}\hat{\mathbf{b}} = \hat{\mathbf{g}}, \tag{18}$$

respectively, where the vector $\mathbf{b} = (b_1, \ldots, b_6)^T$ in the chosen case of fifth order integration, and $\hat{\mathbf{b}} = (\hat{b}_1, \ldots, \hat{b}_7)^T$ for the embedded process. M and $\hat{M}$ contain the coefficients A_{ij}.
We now define an optimization process:

Find coefficients (A_{ij}) *such that* $\lambda \parallel M\mathbf{b} - \mathbf{g} \parallel_2 + (1-\lambda) \parallel \hat{M}\hat{\mathbf{b}} - \hat{\mathbf{g}} \parallel_2$ *is "sufficiently close to zero".*

Here $\mathbf{b}$ and $\hat{\mathbf{b}}$ are, at any stage of the optimization process, the least squares solutions to (17) and (18), respectively, (and hence they are expressed in terms of the (A_{ij})), and $0 < \lambda < 1$ is a weight. The value of $\hat{b}_7$ is chosen arbitrarily.
Since the optimization process is fully numerical, the result is obtained in the form of approximating reals. Some authors prefer a representation in terms of exact rationals. This is elegant indeed, but not essential because in any *actual* computation these rationals are immediately transformed to approximating reals. Of even less use is the translation of coefficients which were obtained as (finite precision) reals into approximating rationals.
In general, the systems of condition equations have an infinit number of solutions. Therefore we have the freedom, to some extent, to control the process such that more desirable solutions arise. This is done (in the present context with emphasis on rounding behaviour) by constraining during the optimization process (in particular in the beginning of the process), the coefficients (A_{ij}) and the solution vectors $\mathbf{b}$ and $\hat{\mathbf{b}}$. Once a solution is found with sufficient accuracy, another solution $\hat{\mathbf{b}}$ to (18) can be found by adding to this solution a vector from the null-space of $\hat{M}$.
The numerical optimization process is a heavy one. Most processes, known from non-linear programming, fail. The dimension of the search-space is at least 14 for the more interesting cases, i.e. Runge-Kutta methods of order 5 (with 6 stages) and higher. The search point seems to move in a highly capricious capillary in such a relatively high-dimensional space. The only optimization process which led us, ultimately, to a solution was a Nelder-Mead type process [NM65, Him72]. Convergence is slow, therefore the most powerful computing equipment available is required for the more interesting cases. Using a standard personal computer with 2.3 GHz processor, we succeeded in getting a solution for the case mentioned above (order 5, with 6 stages and with an embedded process of order 4, for a general system of differential equations) accurate to 24 digits. The control of the optimization process was organized such that the components of $\mathbf{b}$ and $\hat{\mathbf{b}}$, and the coefficients (A_{ij}) are in the interval $[0,\ 1]$ or, otherwise, as near as possible to this interval.
As stated, the optimization process requires much effort. The process is slow when directly programmed in Maple. Using C^{++} (17 digits), however, a speed-up with a factor between 1000 and 2000 is obtained. This allows a computation to be done

in a reasonable time. When desiring a more accurate result (e.g. in 24 digits, as in the present article), the slowing down of the continuation of the computation must be accepted. However, the process needs to be done only once to produce a desirable Runge-Kutta method, which method then can be used for ever. For reducing the computational load it is possible to use a number of order conditions directly, reducing the dimension of the numerical optimization process.

4 A fifth-order Runge-Kutta result

Along the lines described above a Runge-Kutta method of order 5 with embedded method of order 4 was produced. The coefficients are presented in the following scheme. Here $A = (A_{ij}),\quad B = (b_1, \ldots, b_6), \hat{B} = (\hat{b}_1, \ldots, \hat{b}_7),\quad C = (C_2, \ldots, C_6)$ (see (14)).

A=

$$\begin{array}{lll}
.43164015354304871935 0737 & & \\
.18855117698629792626 2854 & .89659174430533176276 2067e-1 & \\
.50325590214249406358 0981e-3 & -.11920492548869614903 5214 & .59826196853826496117 3768 \\
.18956883142249297051 2298e-2 & -.22988235014682373923 6961 & .45239424401543552646 5412 \\
 & & .51594528437374147336 7907 \\
.12111771913426599920 7122 & .76305836117149735422 5105 & 1.13064504544581253082 070 \\
 & -2.32906156594574255527 119 & 1.31424044019416667101 827
\end{array}$$

B=

$$\begin{array}{lll}
.85978396020417673122 3368e-1 & 0 & .40215290244732446888 4692 \\
.42065369400528019257 3430e-1 & .39229901065834616642 0869 & .77504321473383672314 7596e-1
\end{array}$$

$\hat{\text{B}}$=

$$\begin{array}{lll}
.10603441819811952896 0708 & 0 & .29613268468506439370 8661 \\
.19322634931100898110 2530 & .30679103159188716187 5279 & .87815516213919934352 8221e-1 \\
 & & 0.01
\end{array}$$

C=

$$\begin{array}{lll}
.43164015354304871935 0737 & .27821035141683110253 9061 & .47956029895171130620 2135 \\
 & .74035286655657819030 1481 & 1
\end{array}$$

Comparison with the fifth order method of Dormand-Prince shows that the coefficients of the present method are closer to the unit interval:

	$\max(A_{ij})$	$\min(A_{ij})$	$\min(b_i)$
present method	1.31	-2.33	0
Dormand-Prince	9.8	-11.6	-0.32

This means that we arrived at a better *discrete condition number* [DB02]. The discrete condition κ_Δ satisfies

$$\kappa_\Delta \ \leq \ e^{\gamma LT}$$

in which Δ represents the mesh on the interval $[0,\ T]$, $\gamma \ \geq \ 1$, and L is the Lipschitz constant of the associated initial value problem $\mathbf{x}' = \mathbf{f}(t, \mathbf{x})$:

$$\| \ \mathbf{f}(t, \mathbf{x}) - \mathbf{f}(t, \bar{\mathbf{x}}) \ \| \ \ \leq \ L \ \| \ \mathbf{x} - \bar{\mathbf{x}} \ \|, \ \ \forall (t, \mathbf{x}), (t, \bar{\mathbf{x}}) \text{ in the relevant domains.}$$

For the interval condition $\kappa[0,\ T]$ of the initial value problem holds

$$\kappa[0,\ T] \ \leq \ e^{LT},$$

meaning that γ is a measure for the "worsening" of the condition of the discrete problem with respect to that of the continuous problem. For a given Runge-Kutta method γ is a constant; it depends only on the coefficients (b_i) and (A_{ij}). For computation of γ see [DB02]. For the 5th order Dormand-Prince method we find $\gamma = 5.74$. The present 5th order method gives $\gamma = 1.50$. To establish the effect of this, a number of test calculations were done for the gravitational two-body problem, which show that the present method is more accurate in the situation where the local truncation error comes close to the machine precision (smallest integration stepsize which makes sense in relation to the wordlength of the machine), in spite of the fact that the truncation error of the present method is about 2.7 times that of the Do-Pr-method (the Do-Pr-method is "error-tuned").
In the next two figures the error in the radius vector after one revolution of the gravitating bodies, and the convergence of the truncation error ratio to the factor of 2^5, respectively, are plotted. The ratio plot of the Do-Pr-method shows very capricious behaviour for the larger stepsizes. (This suggests that, for larger stepsizes, the error behaviour and step control of error-tuned methods might be rather unpredictable).

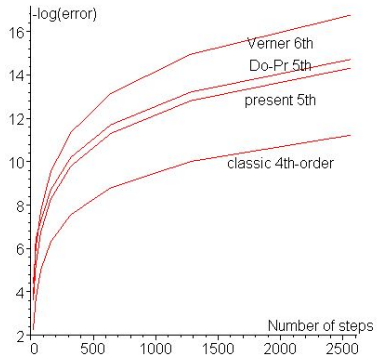

Figure 1: Error in radiusvector

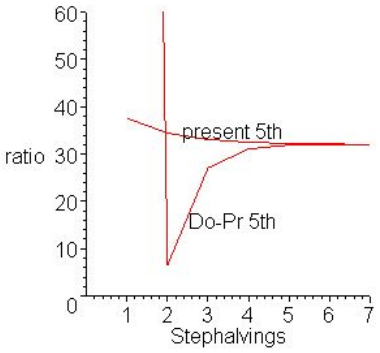

Figure 2: Order of convergence

In the next table the influence of the machine precision on the error in the radius vector, for the problem of the gravitating bodies after 10 revolutions, is presented. The truncation errors after one revolution are 0.49 10^{-14} and 0.19 10^{-14}, respectively, with 2560 steps per revolution, using 23 digits.

	23 digits	15 digits	14 digits
present method	0.83-11	0.51-10	0.79-10
Dormand-Prince	0.28-11	0.40-10	1.58-10

Remarkably, employing compensated summation in the Do-Pr-method does not change this picture so much.

Several other criteria for measuring the quality of a Runge-Kutta method are used in literature. For a 5th order method with embedded 4th order method these read [DB02]:

$$A_6 = \| e^{(6)} \|_2 = (\sum_\beta | e_\beta^{(6)} |^2)^{\frac{1}{2}},$$

$$B_5 = \frac{\| \hat{e}^{(6)} \|_2}{\| \hat{e}^{(5)} \|_2}, \quad \text{and}$$

$$C_5 = \frac{\| \hat{e}^{(6)} - e^{(6)} \|_2}{\| \hat{e}^{(5)} \|_2},$$

which all should be as small as possible.
Referring to (16) we put

$$A_6 = (\sum_k | \beta_k^{(6)} |^2)^{\frac{1}{2}},$$

which is equivalent (but not identical) to the quantity A_6 from [DB02]. Minimizing A_6 (over all Runge-Kutta coefficients) means minimizing, on the average, the truncation error of the 5th order method.

We put

$$B_5 = \frac{(\sum_k \mid \hat{\beta}_k^{(6)} \mid^2)^{\frac{1}{2}}}{(\sum_k \mid \hat{\beta}_k^{(5)} \mid^2)^{\frac{1}{2}}}$$

and

$$C_5 = \frac{(\sum_k \mid \hat{\beta}_k^{(6)} - \beta_k^{(6)} \mid^2)^{\frac{1}{2}}}{(\sum_k \mid \hat{\beta}_k^{(5)} \mid^2)^{\frac{1}{2}}}.$$

Smallness of B_5 and C_5 gives a more accurate local error estimate and a more reliable stepcontrol. Values of A_6, B_5 and C_5 are collected in the following table:

	A_6	B_5	C_5
present method5(4)	0.118	9.1	4.5
Dormand-Prince5(4)	0.0131	6.7	7.3
Fehlberg5(4)	0.171	12.5	6.3

where for the computation of B_5 and C_5 the vector $\hat{\mathbf{b}}$ has been used as given in the beginning of this section.
The vector $\hat{\mathbf{b}}$ has one degree of freedom: an arbitrary vector from the null-space of $\hat{M}$ can be added to it. The difference $\hat{\mathbf{b}} - \hat{\mathbf{b}}_0$ (see (13)) is such a vector. Therefore we write

$$\hat{\mathbf{b}} := \hat{\mathbf{b}}_0 + \lambda(\hat{\mathbf{b}} - \hat{\mathbf{b}}_0), \quad \lambda \in \mathbb{R}.$$

The value of λ influences the value of B_5. For $\lambda = -1.2$ we find a minimum $B_5 = 2.5$ for the present method. In the Do-Pr process a minimum $B_5 = 4.8$ is attained for $\lambda = 0.1$. The value of C_5 is not sensitive to values of λ except in a narrow region around $\lambda = 0$, where C_5 becomes very big. A better vector $\hat{\mathbf{b}}$, better in view of error estimate and stepcontrol, therefore is:
$\hat{\mathrm{B}}$=

$.619111694071754461162912e-1$	0	.529377163762036559095928
$-.139327806492049134956882$	.494908585538096971875577	$.651308877847401578690846e-1$
		$-.12e-1$

The embedded method is such that the error estimator, and thus the stepcontrol, is also suitable for special right sides $\mathbf{f}$ (quadrature problems).

Remark.
The Runge-Kutta result from this section is certainly not the 'final' 5th order method. With the obtained result, however, we have demonstrated that methods can be derived in the presented alternative way, leading possibly to more favourable methods.

5 References

[A87] P.Albrecht. *A new Theoretical Approach to Runge-Kutta Methods.* SIAM J.Numer.Anal., Vol.24, pp.391-406, 1987

[B63] J.C.Butcher. *Coefficients for the study of Runge-Kutta integration processes.* J.Austral.Math.Soc., Vol.3, pp.185-201, 1963

[DB02] P.Deuflhard and F.Bornemann. *Scientific Computing with Ordinary Differential Equations.* Springer, 2002

[Him72] D.M.Himmelblau. *Applied Nonlinear Programming.* McGRAW-Hill, 1972

[HNW93] E.Hairer, S.P.Nϕrsett and G.Wanner. *Solving Ordinary Differential Equations I: Nonstiff problems.* Springer, 1993

[Lam91] J.D.Lambert. *Numerical Methods for Ordinary Differential Systems: The initial value problem.* Wiley, 1991

[NM65] J.A.Nelder and R.Mead. *A Simplex Method for Function Minimization.* Computer J., Vol.7, pp.308-313, 1965