NASA Technical Memorandum 107313

Designing ROW Methods

Alan D. Freed
Lewis Research Center
Cleveland, Ohio

September 1996

National Aeronautics and
Space Administration

Designing ROW Methods

Alan D. Freed*

June 26, 1996

Abstract

There are many aspects to consider when designing a Rosenbrock-Wanner-Wolfbrandt (ROW) method for the numerical integration of ordinary differential equations (ODE's) solving initial value problems (IVP's). The process can be simplified by constructing ROW methods around good Runge-Kutta (RK) methods. The formulation of a new, simple, embedded, third-order, ROW method demonstrates this design approach.

This paper outlines a recipe used by the author for developing ROW methods. Being an engineer, my approach differs somewhat from that of the mathematician (e.g., Hairer and Wanner [8]). This engineer breaks down the complex problem or ROW construction into two simpler problems; whereas, the mathematician typically retains all complexity and solves a single problem. The objective, in either case, is to be able to develop efficient and robust formulæ for integrating systems of ODE's that possess stiff regions in their solution paths.

Let us consider the numerical integration of an IVP described by the non-autonomous system of ODE's

$$\frac{d\mathbf{y}}{dx} \equiv \dot{\mathbf{y}} = \mathbf{f}(x, \mathbf{y}(x)) \qquad \text{with} \qquad \mathbf{y}(x_0) = \mathbf{y}_0\,, \tag{1}$$

where $\mathbf{y}$ is a vector composed of m dependent variables $\{y^1 \;\; y^2 \;\; \cdots \;\; y^m\}^\mathsf{T}$ with x being the single independent variable.

1 Design

Selecting a good RK method first simplifies the overall process of ROW design. Specifically, this author selects embedded methods to facilitate run-time error analysis, developed after the manner of Merson [10], but more commonly referred to as Fehlberg

*Computational Materials Laboratory, MS 105–1, NASA Lewis Research Center, 21000 Brookpark Road, Cleveland, OH 44135-3191. Email: afreed@lerc.nasa.gov.

[3, 4] methods. Such methods advance a solution via the formulæ

$$\left.\begin{aligned} \mathbf{y}_{n+1} &= \mathbf{y}_n + h\sum_{i=0}^{s-1} c_i\mathbf{k}_i + \mathcal{O}(h^{p+1}) \\ \widehat{\mathbf{y}}_{n+1} &= \mathbf{y}_n + h\sum_{i=0}^{s-1} \widehat{c}_i\mathbf{k}_i + \mathcal{O}(h^{q+1}) \end{aligned}\right\}. \tag{2}$$

The order of accuracy belonging to approximation $\mathbf{y}$ (the solution advanced) is p, while the order of the embedded approximation $\widehat{\mathbf{y}}$ (used in error assessment via $\mathbf{y}-\widehat{\mathbf{y}}$) is q, where $q \neq p$. The summation limit s denotes the number of stages in the integrator. Subscripts n and $n+1$ signify the n^{th} and $(n+1)^{\text{th}}$ integration steps. Implied in Eqn. (2) is the relationship $x_{n+1} = x_n + h$, where h is the size of the integration step. Parameters c_i and $\widehat{c}_i$ are the weights of integration, typically lying within the interval $[0, 1]$.

The Runge-Kutta derivatives $\mathbf{k}_i$ are evaluated according to

$$\mathbf{k}_i = \mathbf{f}(x_n + a_i h, \mathbf{y}_n + h\sum_{j=0}^{i-1} A_{ij}\mathbf{k}_j), \tag{3}$$

where vector components a_i are the quadrature points of integration, and matrix components A_{ij} are the *explicit* coupling coefficients. Our intuition and experiences have taught us that these quadratures ought to lie within the interval $[0, 1]$; in other words, all $\mathbf{k}_i$ derivatives should be evaluated *locally* within the interval $[x_n, x_{n+1}]$ of integration.

One does not have complete freedom in selecting the parameters of Eqns. (2 & 3). They are constrained by a set of equations known as order conditions (e.g., see Hairer, Nørsett and Wanner [7, pp. 143–155]). Even so, there remains an aspect of design in RK construction, this because there are an insufficient number of order conditions to uniquely determine all the unknowns. The author chooses to apply this flexibility to specify good quadratures and weights, in the sense of classical integration theory. An additional flexibility arises in the next stage of development, which the author has used to minimize the error constants.

This completes my first stage in ROW design.

The second step of ROW construction is to fabricate a one-parameter family (in d, the Calahan [1] stability parameter) of Rosenbrock [13] integrators of the Wanner-Wolfbrandt [9, 16][1] type, i.e., a ROW method, holding the RK parameters fixed. Embedded ROW methods are described by the same formulæ for the dependent variables $\mathbf{y}_{n+1}$ and $\widehat{\mathbf{y}}_{n+1}$ as are used by explicit RK methods, viz., Eqn. (2). What distinguish ROW integrators from RK integrators are their derivative-like functions;

[1]Wanner was a visiting professor at Chalmers University of Technology in Göteborg, Sweden in 1975–1976 where he lectured on stiff methods. Wolfbrandt was a student there at that time.

specifically,

$$\mathbf{k}_i = \underbrace{\mathbf{f}\Big(x_n + a_i h, \mathbf{y}_n + h\sum_{j=0}^{i-1} A_{ij}\mathbf{k}_j\Big)}_{\text{RK (explicit) part}} + \underbrace{h\Big(b_i \frac{\partial \mathbf{f}_n}{\partial x} + \frac{\partial \mathbf{f}_n}{\partial \mathbf{y}} \cdot \sum_{j=0}^{i} B_{ij}\mathbf{k}_j\Big)}_{\text{semi-implicit part}} \tag{4}$$

with semi-implicit coupling

$$B_{ii} = d \;\forall\; i\,. \tag{5}$$

The evolution of state residing in $\mathbf{k}_i$, i.e., function $\mathbf{f}$, is the same here as it is in the RK derivatives of Eqn. (3). It is the presence of the non-autonomous gradient $\partial \mathbf{f}_n/\partial x$ with quadrature-like coefficients b_i, and the Jacobian $\partial \mathbf{f}_n/\partial \mathbf{y}$ with *semi-implicit* coupling coefficients B_{ij}, that provide the additional contributions of a ROW method. Implicit RK methods do not possess the semi-implicit part of Eqn. (4); rather, they extend the upper limit of the summation (appearing in the argument of $\mathbf{f}$) to i for the semi-implicit case, or to $s-1$ for the fully implicit case.

Like RK methods, one does not have complete freedom in selecting the additional parameters of Eqn. (4), viz., the b_i, B_{ij} and d. They too must satisfy a set of order conditions (e.g., see Hairer and Wanner [8, pp. 110–127]), some of which will have been satisfied a priori when adopting this design approach because the a_i, A_{ij}, c_i and $\widehat{c}_i$ are considered known and satisfy the RK order conditions. It is useful to express the b_i and B_{ij} as functions of d at this stage of development. For methods with four or more stages, there are insufficient order conditions to uniquely determine all remaining unknowns. This extra flexibility, when it occurs, has been used by the author to design for a more optimal performance by reducing the error constants.

This completes my second stage in ROW design.

The final aspect of constructing a ROW method is the selection of d. This parameter governs the stability characteristics of the method. Also affecting stability are the orders of the method's integrators and the number of stages the method contains (e.g., see Hairer and Wanner [8, pp. 103–107]). Ideally, one would like both integrators in an embedded ROW method to be L-stable, although seldom will this be possible. As a minimum requirement, both integrators should be no less than A-stable. Some stability has been sacrificed by designers "since, roughly, smaller values of d give better error constants but increase instability, there is no optimal choice which optimizes simultaneously error constant and stability" [9].

One may want to iterate on steps two and three to seek an optimal compromise between performance and stability. The library of IVP's cataloged by Enright and Pryce [2] is a useful proving ground for this purpose. This author encourages developers to utilize stiff and non-stiff problem sets when assessing ROW performance.

2 Implementation

Direct implementation of the semi-implicit equations in (4) is not very practical. Fortunately, a change in variable introduced by Wolfbrandt [16, pp. 106–107] removes the diagonal contribution from the right-hand side of Eqn. (4) resulting in the transformed formulæ

$$\left.\begin{aligned} \mathrm{y}_{n+1} &= \mathrm{y}_n + h\sum_{i=0}^{s-1}\chi_i\boldsymbol{\kappa}_i + \mathcal{O}(h^{p+1}) \\ \widehat{\mathrm{y}}_{n+1} &= \mathrm{y}_n + h\sum_{i=0}^{s-1}\widehat{\chi}_i\boldsymbol{\kappa}_i + \mathcal{O}(h^{q+1}) \end{aligned}\right\} \tag{6}$$

and

$$\left(\frac{1}{d}\mathbf{I} - h\,\frac{\partial \mathrm{f}_n}{\partial \mathrm{y}}\right)\cdot\boldsymbol{\kappa}_i = \mathrm{f}\Big(x_n + a_i h, \mathrm{y}_n + h\sum_{j=0}^{i-1}\alpha_{ij}\boldsymbol{\kappa}_j\Big) + h\,b_i\,\frac{\partial \mathrm{f}_n}{\partial x} + \sum_{j=0}^{i-1}\beta_{ij}\boldsymbol{\kappa}_j\,. \tag{7}$$

These equations contain the change in variables: $\boldsymbol{\beta} = \frac{1}{d}\mathbf{I} - \mathbf{B}^{-1}$, $\boldsymbol{\alpha} = \mathbf{A}\cdot\mathbf{B}^{-1}$, $\boldsymbol{\chi}^{\mathsf{T}} = \mathbf{c}^{\mathsf{T}}\cdot\mathbf{B}^{-1}$ and $\widehat{\boldsymbol{\chi}}^{\mathsf{T}} = \widehat{\mathbf{c}}^{\mathsf{T}}\cdot\mathbf{B}^{-1}$. Because of the constraints $B_{ii} = d > 0\ \forall\ i$ and $B_{ij} = 0\ \forall\ i < j$, matrix $\mathbf{B}$ has an inverse.

There are several aspects worthy of comment that help transform a ROW method into an efficient and robust algorithm suitable for code implementation, which we have discussed more fully in [5]. In summary: use LU-factorization with full pivoting to decompose the matrix $[\frac{1}{d}\mathbf{I} - h(\partial \mathrm{f}_n/\partial \mathrm{y})]$; employ a local interpolater, made popular by Shampine [15], to permit dense output without requiring step-size reductions; use a stable step-size controller, like the proportional-integral (PI) controller of Gustafsson et al. [6], to monitor step size; estimate the initial step-size internally; construct an error estimate using some blend, e.g., 50/50, of relative and absolute errors, where the weights of absolute error are dynamically updated; and finally, utilize numerical evaluations for the non-autonomous gradient and Jacobian, thereby reducing the potential for programming errors, plus significantly simplifying the client module one must write to make use of such integrators. When combined with a ROW method, these various procedures will produce an efficient and robust algorithm for numerical integration.

3 Example

The first stage in my recipe is to choose or construct an embedded RK method with good quadrature and weights of integration. As a means for demonstration, consider the simple, 3 stage, third-order method given in Table 1, which has good quadrature, i.e., $\{0\ \ \frac{1}{2}\ \ 1\}^{\mathsf{T}}$, plus Simpson's weights of integration, viz., $\{\frac{1}{6}\ \ \frac{2}{3}\ \ \frac{1}{6}\}^{\mathsf{T}}$. The coefficients listed in this table satisfy the order conditions of a 3(2) RK method—a third-order solution with an embedded second-order integrator for error assessment.

Table 1: A 3(2) Runge-Kutta method.

a_i	A_{ij}		
0	0		
$\frac{1}{2}$	$\frac{1}{2}$	0	
1	-1	2	0
$\widehat{c}_j$	0	1	0
c_j	$\frac{1}{6}$	$\frac{2}{3}$	$\frac{1}{6}$

Table 2: A 3(2) Rosenbrock-Wanner-Wolfbrandt method.

a_i	α_{ij}			b_i	β_{ij}		
0	0			d	0		
$\frac{1}{2}$	$1/2d$	0		0	$-1/d$	0	
1	$1/d$	$2/d$	0	$-d$	$-2/d$	$-4(2-3d)/d(1-2d)$	0
$\widehat{\chi}_j$	$1/d$	$1/d$	0	χ_j	$7/6d$	$2(3-5d)/3d(1-2d)$	$1/6d$

For a 3(2) ROW method in 3 stages, where the a_i and A_{ij} are specified a priori, the number of remaining order conditions happens to be one less than the number of unknowns to be solved for. Consequently, the b_i and B_{ij} can be uniquely expressed as a one-parameter family of Calahan's [1] stability parameter d. Their values, after transformation, are listed in Table 2. This completes my second stage of ROW design.

The final stage of design is to select a stability parameter d. This example is a rare exception where both integrators in a ROW method can be L-stable. A second-order integrator in 3 stages (the embedded integrator) will be L-stable if $d \in [0.18042531, 2.18560010]$ (see Hairer and Wanner [8, Table 6.4]). A third-order integrator in 3 stages will be L-stable if $d = 0.43586652$, which is the reciprocal of the second zero in the third-order Laguerre polynomial; therefore, selecting

$$d = 0.43586652$$

could, arguably, complete the design.

Not yet addressed is the issue of the error constants. Requiring the integrator to be at least A-stable leads to the constraint that $d \in [1/3, 1.06857902]$ (see Hairer and Wanner [8, Table 6.3]). The embedded integrator is L-stable over this region. One now seeks an answer to the question: Is there an A-stable value for d whose selection would result in a significant performance gain over its L-stable value, thereby warranting a compromise in the integrator's stability properties?

In most developments of ROW methods there are an insufficient number of order conditions to uniquely define the b_i and B_{ij} as functions of d alone. When this is the case, the designer should make attempts to minimize the error constants. Error constants are nothing more than the order conditions for that order which is one greater than the actual integrator. For example, in the 3(2) ROW method of Table

2, the embedded integrator fails to satisfy the third-order order conditions, which are two in number, resulting in the two error constants

$$e_1^{(2)} = -1/12 \qquad \text{and} \qquad e_2^{(2)} = -(1 - 6d + 6d^2)/6\,. \tag{8}$$

Likewise, the approximation advanced as the solution arises from a third-order integrator that fails to satisfy the fourth-order order conditions, which are four in number, resulting in the four error constants

$$\left.\begin{array}{ll} e_1^{(3)} = 0 & e_2^{(3)} = -d^2/6(1-2d) \\ e_3^{(3)} = (1+8d)/24 & e_4^{(3)} = -(1 - 12d + 36d^2 - 24d^3)/24 \end{array}\right\}. \tag{9}$$

These are the error constants for the integrator of Table 2 that one would want to minimize in an attempt to optimize performance. Figure 1 presents the norm for these error constants, calculated according to

$$\|\mathbf{e}^{(p)}\| = \left(\sum_i \left(e_i^{(p)}\right)^2\right)^{1/2}. \tag{10}$$

Viewing Fig. 1, it is immediately apparent that there is no advantage to considering values for d larger than the L-stable value. Choosing $d = 2/5$ leads to a 29% improvement in the norm for the error constants with only a minimal sacrifice in stability, while choosing $d = 1/3$ gives a 48% improvement but places stability at the boundary of the A-stable region. The compromise value of $d = 2/5$ seems, at first glance, to be a good design selection, but as we'll see, this is premature.

3.1 Numerical Exercise

As a demonstration of capability, the performance of this new, 3-stage, 3(2) ROW method is compared against the performances of two, 4-stage, ROW methods of higher order—the 3(4) method Wolfbrandt [16, 11], MROS3, and the 4(3) method of Shampine [14], S-ROW, which is the Rosenbrock integrator promoted by Press et al. [12, pp. 738–742]. These constitute tough competitors. Also compared are the performances of the RK methods contained within these three ROW methods.

The Brusselator from chemical kinetics [7, pp. 115–116] serves as a good example problem, i.e.,

$$\left.\begin{array}{l} \dot{y}_0 = c_0 + y_0^2 y_1 - (c_1 + 1) y_0 \\ \dot{y}_1 = c_1 y_0 - y_0^2 y_1 \end{array}\right\} \tag{11}$$

which, by varying the values for constants c_0 and c_1, varies its stiffness properties. This is illustrated in Table 3, wherein λ denotes an eigenvalue from the Jacobian of Eqn. 11. Case i has a periodic solution; it is not stiff. Cases ii, iii and iv successively increase in stiffness, and have monotonic solutions over the interval $x \in [0, 100]$, which is the interval of integration for all reported results, with initial condition $y_0(0) = 1.5$ and $y_1(0) = 3.1$.

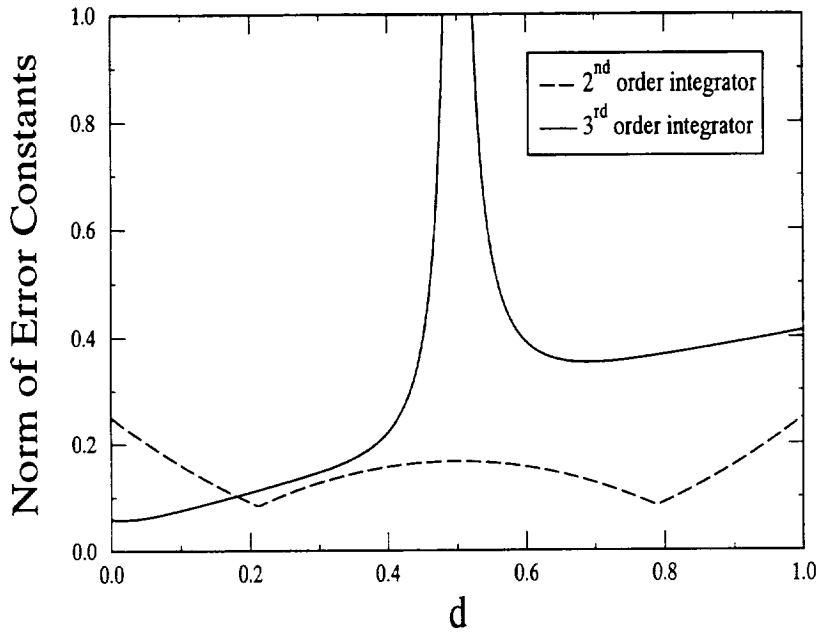

Figure 1: Error constant norms belonging to second- and third-order integrators in 3(2) ROW method of Table 2.

Table 3: Stiffness Ratios of the Brusselator.

case	c_0	c_1	$\lambda_{max}/\lambda_{min}$
i	1	5	7
ii	1	50	2 500
iii	1	500	250 000
iv	1	5 000	25 000 000

Table 4: Performance of Three RK Methods.

case	RK 3(2)	RK in MROS3	RK in S-ROW
	$\epsilon = 0.01$		
i	363/588	491/569	333/1 090
ii	1 955/2 441	2 723/3 256	980/1 129
iii	19 941/24 927	32 451/163 073	9 875/12 339
iv	198 791/248 493	321 204/1 616 498	98 522/123 151
	$\epsilon = 0.001$		
i	638/1 563	696/1 650	763/4 883
ii	1 969/1 984	2 675/2 762	1 004/1 016
iii	19 955/24 942	31 921/161 011	9 899/12 370
iv	198 803/248 497	311 520/1 420 202	98 544/123 174
	$\epsilon = 0.0001$		
i	1 188/3 613	1 589/10 889	1 881/16 996
ii	1 992/2 007	2 715/2 773	1 060/1 071
iii	19 979/24 965	28 370/73 977	9 963/13 587
iv	198 828/248 520	316 566/1 545 963	98 602/123 235

The performance of the three, contained, RK methods is tabulated in Table 4. Reported are the number of steps required over the number of attempts made, each at three different accuracies for step control, i.e., 0.01, 0.001 and 0.0001. RK 3(2) out performs the other two RK methods for the non-stiff ODE's of case *i*, which should not be a surprise since neither of the other two constructions gave consideration to the RK method within. When the ODE's of Eqn. (11) become stiff, the RK method in S-ROW is superior, although one would not normally use an RK method for such stiff problems. Accuracy of solution, ϵ, has little influence on the solution time in any of the stiff cases. Stability controls the solution time here.

The performance of the three ROW methods is listed in Table 5. Without exception, all three ROW methods were more efficient then their embedded RK counterparts for all four cases, as measured by the number of integration steps required. There are slight improvements in the non-stiff case where accuracy controls step size, and dramatic improvements in the three stiff cases where stability controls step size, as one would expect for both situations. At the tighter tolerance of $\epsilon = 0.0001$, the fourth-order integrators MROS3 and S-ROW begin to out perform my third-order method, also an expected result. Since the 3(2) integrator of Table 2 has one less stage in it, it can make up for slight performance losses (as measured by the number of steps taken) due to the smaller system required for LU decomposition and the forward/backward substitution process accompanying it. The very slight performance gain obtained in an effort to minimize the error constants, as reported in Table 5, does not warrant compromising stability in this case, answering the question posed in the previous section.

Table 5: Performance of Three ROW Methods.

case	ROW 3(2)			MROS3	S-ROW
	$d = 1/3$	$d = 2/5$	L-stable		
	$\epsilon = 0.01$				
i	289/344	249/367	315/403	231/268	287/347
ii	25/25	25/25	25/25	21/21	24/24
iii	28/28	27/27	27/27	24/24	29/29
iv	30/30	30/30	30/30	26/26	31/31
	$\epsilon = 0.001$				
i	481/880	516/1220	544/999	324/424	441/554
ii	36/36	37/37	37/37	27/27	34/34
iii	40/40	40/40	41/41	31/31	39/39
iv	42/42	43/43	43/43	33/33	42/42
	$\epsilon = 0.0001$				
i	897/2576	976/3059	1 021/3 159	494/632	714/1 318
ii	57/57	59/59	60/60	38/38	49/49
iii	62/62	64/64	64/64	43/43	56/56
iv	65/65	68/68	68/68	46/46	61/61

4 Remarks

The author finds it much simpler to follow the design approach outlined in this paper when constructing ROW methods than to use the more brute force approach of his predecessors.

The 3(2) ROW method presented herein is L-stable and computationally efficient. It was designed for use in simulation and optimization codes, where robust performance matters and the accuracy of solution can oftentimes be relaxed, hence the lower order. It is an excellent integrator for these purposes.

Because of the linear system of equations that must be solved, more CPU work is required per step from a ROW method than from a RK method of comparable size. RK methods are therefore still preferred for non-stiff ODE's. For stiff ODE's, however, ROW methods are clearly superior. If one is uncertain about the stiffness properties belonging to a system of ODE's, then selecting a ROW method should not excessively increase the cost of computation whenever the problem turns out to be non-stiff, but whenever it ends up being stiff, it will likely save on cost, and this savings can be significant.

Acknowledgments

Over the course of this author's investigation into Rosenbrock methods, instructive comments have been furnished by Professors Kaps, Wanner and Shampine, to whom the author is grateful. He is also grateful to Dr. Iskovitz for our many valuable conversations on numerical methods. Her insistence on detail is both gratifying and refreshing.

References

[1] CALAHAN, D. A. A Stable, Accurate Method of Numerical Integration for Nonlinear Systems. *Proceedings of the IEEE,* **56** (1968), 744.

[2] ENRIGHT, W. H., AND PRYCE, J. D. Two FORTRAN Packages for Assessing Initial Value Problems. *ACM Transactions on Mathematical Software,* **13** (1987), 1–27.

[3] FEHLBERG, E. Classical Fifth-, Sixth-, Seventh-, and Eighth-Order Runge-Kutta Formulas With Stepsize Control. NASA Technical Report TR R-287, NASA, Marshall Space Flight Center, Huntsville, AL, 1968.

[4] FEHLBERG, E. Low-Order Classical Runge-Kutta Formulas With Stepsize Control and Their Application to Some Heat Transfer Problems. NASA Technical Report TR R-315, NASA, Marshall Space Flight Center, Huntsville, AL, 1969.

[5] FREED, A. D., AND ISKOVITZ, I. S. Development and Applications of a Rosenbrock Integrator. NASA Technical Memorandum 4709, NASA, Lewis Research Center, Cleveland, OH, 1996.

[6] GUSTAFSSON, K., LUNDH, M., AND SÖDERLIND, G. A PI Stepsize Control for the Numerical Solution of Ordinary Differential Equations. *BIT,* **28** (1988), 270–287.

[7] HAIRER, E., NØRSETT, S. P., AND WANNER, G. *Solving Ordinary Differential Equations I: nonstiff problems*, second ed. Vol. 8 of *Springer Series in Computational Mathematics*, J. S. R. L. Graham and R. Varga, (eds.). Springer-Verlag, Berlin, 1993.

[8] HAIRER, E., AND WANNER, G. *Solving Ordinary Differential Equations II: stiff and differential-algebraic problems.* Vol. 14 of *Springer Series in Computational Mathematics*, J. S. R. L. Graham and R. Varga, (eds.). Springer-Verlag, Berlin, 1991.

[9] KAPS, P., AND WANNER, G. A Study of Rosenbrock-Type Methods of High Order. *Numerische Mathematik,* **38** (1981), 279–298.

[10] MERSON, R. H. An Operational Method for the Study of Integration Processes. In *Proceedings of the Symposium on Data Processing* (Salisbury, Australia, 1957), Weapons Research Establishment, pp. 110–1 to 110–25.

[11] NØRSETT, S. P., AND WOLFBRANDT, A. Order Conditions for Rosenbrock Type Methods. *Numerische Mathematik,* **32** (1979), 1–15.

[12] PRESS, W. H., TEUKOLSKY, S. A., VETTERLING, W. T., AND FLANNERY, B. P. *Numerical Recipes in C: the art of scientific computing*, second ed. Cambridge University Press, Cambridge, 1992.

[13] ROSENBROCK, H. H. Some General Implicit Processes for the Numerical Solution of Differential Equations. *Computer Journal,* **5** (1963), 329–330.

[14] SHAMPINE, L. F. Implementation of Rosenbrock Methods. *ACM Transactions on Mathematical Software,* **8** (1982), 93–113.

[15] SHAMPINE, L. F. Interpolation for Runge-Kutta Methods. *SIAM Journal on Numerical Analysis,* **22** (1985), 1014–1027.

[16] WOLFBRANDT, A. *A Study of Rosenbrock Processes With Respect to Order Conditions & Stiff Stability.* PhD thesis, Chalmers University & The University of Göteborg, Göteborg, Sweden, January 1978.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	September 1996	Technical Memorandum

4. TITLE AND SUBTITLE

Designing ROW Methods

5. FUNDING NUMBERS

WU-505-63-5A

6. AUTHOR(S)

Alan D. Freed

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Lewis Research Center
Cleveland, Ohio 44135-3191

8. PERFORMING ORGANIZATION REPORT NUMBER

E-10408

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Washington, D.C. 20546-0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

NASA TM-107313

11. SUPPLEMENTARY NOTES

Responsible person, Alan D. Freed, organization code 5110, (216) 433-8747.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified - Unlimited
Subject Category 64

This publication is available from the NASA Center for AeroSpace Information, (301) 621-0390.

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

There are many aspects to consider when designing a Rosenbrock-Wanner-Wolfbrandt (ROW) method for the numerical integration of ordinary differential equations (ODE's) solving initial value problems (IVP's). The process can be simplified by constructing ROW methods around good Runge-Kutta (RK) methods. The formulation of a new, simple, embedded, third-order, ROW method demonstrates this design approach.

14. SUBJECT TERMS

Numerical integration; Differential equations

15. NUMBER OF PAGES

12

16. PRICE CODE

A03

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	