

PORTLAND STATE UNIVERSITY
DEPARTMENT OF MATHEMATICAL SCIENCES

507 COLLOQUIUM

TITLE: "Solving $\frac{dy}{dx} = F(x, y)$:
Global Interpolation Via The
Splicing of Local Interpolants"

SPEAKER: Michael Runstein

TIME: Friday, May 15, 1987, 3:00 p.m. - 4:00 p.m.

PLACE: Room 350 Neuberger Hall

ABSTRACT

"Solving $\frac{dy}{dx} = F(x, y)$:

GLOBAL INTERPOLATION VIA THE
SPLICING OF LOCAL INTERPOLANTS"

When a Runge-Kutta method is applied to solving a differential equation the problem, "How do we connect the dots?," becomes apparent. For dealing with this question the popular, "Adams and Backward Difference Formula," codes use polynomial approximation on each $[x_n, x_{n+1}]$. Although these classical methods can achieve arbitrarily good accuracy by using sufficiently small steps, we are still confronted with, "How do we obtain a globally smooth curve?" When one or multistep methods generate a continuous approximation, there will be jumps in the first derivative at the mesh points. On the other hand, if the first derivative is continuous at the mesh points, then there will be jumps in the functions value at the mesh points. In short, jumps or corners will occur at the mesh points.

In this lecture a method is discussed for circumventing this problem.

"SOLVING $\frac{dy}{dx} = F(x, y)$:

GLOBAL INTERPOLATION VIA THE
SPLICING OF LOCAL INTERPOLANTS"

PROJECT

for

MTH 507

Literature Problem Seminar

Michael Runstein

Department of Mathematical Sciences

Portland State University

Portland, Oregon

May, 1987

TO MY FATHER

WITH LOVE AND APPRECIATION

This is a report on the article by Lawrence Shampine titled, "Interpolation for Runge-Kutta Methods." Dr. John Butler, a professor at Portland State University, provided the article from The SIAM Journal of Numerical Analysis. It is being written to satisfy the requirements of Math 507 as specified.

The interpolating method developed at Sandia National Laboratories, of Albuquerque, New Mexico, approximates a solution to the differential equation $\frac{dy}{dx} = F(x, y)$. Globally, a Runge-Kutta method is used to generate a discrete approximation to $f(x)$ over an interval. Hermitian interpolation is then applied successively to each sub-interval $[x_0, x_2]$, $[x_2, x_4]$, etc., where x_0 represents the initial x , x_2 represents $x_0 + 2h$, h represents the Runge-Kutta step, and so on.

When a Runge-Kutta method is applied to solving a differential equation the problem, "How do we connect the dots?," becomes apparent. For dealing with this question the popular, "Adams and Backward Difference Formula," codes use polynomial approximation on each $[x_n, x_{n+1}]$. Although these classical methods can achieve arbitrarily good accuracy by using sufficiently small steps, we are still confronted with, "How do we obtain a globally smooth curve?" When classical methods generate a continuous approximation there will be jumps in the first derivative at the mesh points. On the other hand, if the first derivative is continuous at the mesh points, then there will be jumps in the functions value at the mesh points. In short, jumps or corners will occur at the mesh points.

In Shampine's paper, and here, a method is discussed for circumventing this problem. Shampine gives the theoretical results for higher orders and provides a particular fourth order schema using this method of interpolation. This paper will use

the interpolating schema Shampine wrote about in the article.

In order to obtain a smooth function, $C[0]$ and $C[1]$, over $[A, B]$ Shampine looks at interpolation from two levels, the global and the local. By global we mean $[A, B]$ and by local we mean over $[x_n, x_{n+2}]$ where n refers to the n^{th} Runge-Kutta approximation. In the article, Shampine refers to a $1/2$ step. It is not a $1/2$ step for the Runge-Kutta method. It is a $1/2$ step in the following sense. Each Hermitian approximating polynomial is computed using the interval $[x_n, x_{n+2}]$ where n is an even integer. The first Hermitian approximation is applied over $[x_0, x_2]$, the second Hermitian polynomial approximates over $[x_2, x_4]$, etc. The $1/2$ step which Shampine refers to is a $1/2$ step with respect to the Hermitian polynomial in the sense that it advances the independent variable x from x_n to $x_n + h$ while the Hermitian approximates y from x_n to $x_n + 2h$.

The Algorithm

The algorithm uses the classical Runge-Kutta, Lagrange, and Hermitian integration and interpolations methods. Their formulas and accuracies follow a general explanation of the algorithm.

The algorithm approximates the solution to $\frac{dy}{dx} = F(x, y)$ from two levels, the global and the local.

Global:

At the global level a Runge-Kutta method is applied. We will advance the approximate solution from x_i to x_{i+1} . This is over $[A, B]$. We set $x_0 = A$, $x_1 = A + h$, $x_2 = A + 2h$, and so on. The step size is $h = (B-A)/n$, where n is the number of points generated by the Runge-Kutta method over $[A, B]$.

$$x_n = A + nh = B.$$

Local:

Hermitian interpolation is applied at the local level.

Therefore, we must also generate the Lagrange interpolating polynomials. Using the Runge-Kutta method we obtain

$(x_{2i-2}, Y_{2i-2}, Y'_{2i-2})$, $(x_{2i-1}, Y_{2i-1}, Y'_{2i-1})$, and $(x_{2i}, Y_{2i}, Y'_{2i})$, where $0 < i \leq \frac{n}{2}$.

From these values the Lagrange and Hermite interpolating polynomials are computed for $[x_{2i-2}, x_{2i}]$.

This is the part of the algorithm that is unique:

Splicing $[x_{2i-2}, x_{2i}]$ to $[x_{2i}, x_{2i+2}]$, $0 < i < \frac{n}{2}$,

in order to obtain a globally smooth function. The Hermitian interpolating polynomials meet the specified values and their derivatives at the mesh points. We will exploit this fact to obtain our smooth function.

First, calculate the Hermitian polynomial over $[x_0, x_2]$. This can be used to approximate y over $[x_0, x_2]$. Then calculate the Hermitian polynomial over $[x_2, x_4]$. This can be used to approximate y over $[x_2, x_4]$. And so on.

The accuracy of the Hermitian interpolation is fifth order. The accuracy of the algorithm used in this paper will be that of the Runge-Kutta method or fourth order. The approximation will be smooth since both the function values and the derivative values have been specified at the mesh points. This technique can be applied to any Runge-Kutta method. If a second order Runge-Kutta method is applied globally, then the algorithm will produce results which are accurate to the second order. If a fifth order Runge-Kutta method is applied, then the algorithm will produce results which are accurate to the fifth order. We must remember that the Hermitian interpolation is completely local in nature and is used by the algorithm to obtain a smooth approximation at non-mesh points over $[A, B]$.

Keeping the preceding in mind we can conclude the following:

- i) The mesh points generated by the Runge-Kutta method will have the accuracy associated with the Runge-Kutta integration.
- ii) The non-mesh points obtained from the local Hermite interpolation will be of fifth order accuracy or the accuracy associated with the Runge-Kutta scheme, whichever is less.

Runge-Kutta Method

To approximate the solution of the initial value problem

$$y' = \frac{dy}{dx} = F(x, y), \quad A \leq x \leq B, \quad y(A) = \alpha$$

at $n + 1$ equally spaced points in $[A, B]$.

Input: A, B , integer n , initial condition α

Output: (x_i, y_i, y'_i)

(1) Input: A, B, n, α

(2) Set $h = (B - A)/n$

$$x_0 = A$$

$$y_0 = \alpha$$

(3) For $i = 0$ to n do

$$y'_i = F(x_i, y_i)$$

$$\text{Set } K_1 = F\left(x_i + \frac{h}{2}, y_i + \frac{hy'_i}{2}\right)$$

$$K_2 = F\left(x_i + \frac{h}{2}, y_i + \frac{h}{4}(y'_i + K_1)\right)$$

$$K_3 = F(x_i + h, y_i + h(-K_1 + 2K_2))$$

(4) output (x_i, y_i, y'_i)

(5) $x_{i+1} = x_i + h$

$$y_{i+1} = y_i + h\left(\frac{1}{6}y'_i + \frac{2}{3}K_2 + \frac{1}{6}K_3\right)$$

(6) increment loop counter

(7) repeat loop if counter is less than n .

This Runge-Kutta scheme has fourth order accuracy. When the results from Shampine's were compared to the results in the table of "Numerical Analysis", by Burden et al, [p. 205-207], they were very close to the values obtained using the traditional fourth order Runge-Kutta algorithm.

As with all Runge-Kutta schemes the errors will accumulate. $O(h^4)$ refers to a step. At $x_0 + 10,000h$ the algorithms per step errors will have accumulated and the accuracy of $y_{10,000}$ will be substantially diminished.

Hermite Interpolation

Osculating polynomials are polynomials $P(x)$ which approximate $f(x)$ and satisfy the following:

$x_0, x_1, \dots, x_n$ are $n + 1$ distinct numbers in $[A, B]$ and m_i is a non-negative integer associated with x_i for $i = 0, 1, \dots, n$,

where $m = \max m_i$,

$$0 \leq i \leq n$$

$$f \in C^{[m]}[A, B],$$

and

$$\frac{d^k P(x_i)}{dx^k} = \frac{d^k f(x_i)}{dx^k}$$

for $i = 0, 1, 2, \dots, n$ and $k = 0, 1, \dots, m_i$

When $m_i = 0$ the polynomials are the Lagrange polynomials. When $m_i = 1$ they are the Hermite polynomials. The following pages will give the general interpolating formulas, the specific interpolating formulas used by the algorithm, and their accuracies.

The general Lagrange interpolating polynomial is given by

$$L_{n,k} = \frac{(x-x_0)(x-x_1) \dots (x-x_{k-1})(x-x_{k+1}) \dots (x-x_n)}{(x_k-x_0)(x_k-x_1) \dots (x_k-x_{k-1})(x_k-x_{k+1}) \dots (x_k-x_n)}$$

$$= \prod_{\substack{i=0 \\ i \neq k}}^n \frac{(x-x_i)}{(x_k-x_i)} = L_{n,k}$$

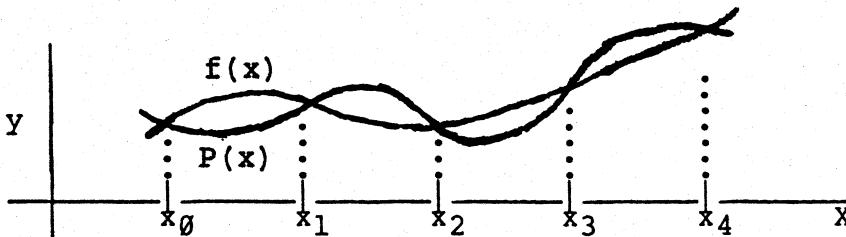
The approximating polynomial must have the property

$P(x) = f(x)$ at all x_i , which is the case when

$$P(x) = \sum_{i=0}^n y_i L_i(x).$$

The ratio $\frac{P(x)}{f(x)} = 1$ when $x = x_i$.

To see this substitute x_i into $P(x)$ and look at the graph representing a typical $P(x)$ and $f(x)$.



If $x_0, x_1, \dots, x_n$ are distinct points in $[A, B]$ and if $f \in C^{[n+1]}[A, B]$, then there exists $\xi(x) \in (A, B)$ such that

$$\begin{aligned} f(x) &= P(x) + \frac{f^{(n+1)}(\xi(x))}{(n+1)!} (x-x_0)(x-x_1) \dots (x-x_n) \\ &= P(x) + \frac{f^{(n+1)}(\xi(x))}{(n+1)!} \prod_{i=0}^n (x-x_i) \end{aligned}$$

The algorithm we have used computes the Lagrange interpolating polynomials using x_0, x_1 , and x_2

hence

$$f(x) - P(x) = \frac{f^{(3)}(\xi(x))}{(3)!} (x-x_0)(x-x_1)(x-x_2)$$

Since the step size is h , it follows

$$x_i = A + ih,$$

$$x_{i+2} = A + (i+2)h, \text{ and}$$

$$|f(x) - P(x)| \leq \frac{|f^{(3)}(\xi(x))|}{3!} |(x-x_0)(x-x_1)(x-x_2)|$$

and

$$|f(x) - P(x)| \leq \frac{|f^{(3)}(\xi(x))|}{6} |x-x_0| |x-x_1| |x-x_2|$$

which is $O(h^3)$ since each $|x-x_i|$ is $O(h)$.

The Lagrange polynomials used by the algorithm are:

$$L_0(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} ,$$

$$L_1(x) = \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} ,$$

and

$$L_2(x) = \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}$$

The set of osculating polynomials, $P(x)$, which agree with $f \in C^{[1]}[A, B]$, where

$x_0, x_1, \dots, x_n \in [A, B]$ are distinct,

$P(x_i) = f(x_i)$ and $P'(x_i) = f'(x_i)$ for $0 \leq i \leq n$, are called the Hermite polynomials. Their general form is:

$$H(x) = \sum_{j=0}^n f(x_j) H_j(x) + \sum_{j=0}^n f'(x_j) \hat{H}_j(x)$$

where

$$H_j(x) = [1 - 2(x-x_j)L'_j(x_j)][L_j(x)]^2 ,$$

and

$$\hat{H}_j(x) = (x-x_j)[L_j(x)]^2.$$

The order of accuracy of $H(x)$, as defined above, is $2n+1$. The algorithm we have used employs these formulas when $n = 2$. The local Hermitian interpolation is accurate to the fifth order.

References

1. Shampine, Lawrence, "Interpolation For Runge-Kutta Methods," SIAM Journal of Numerical Analysis, Vol. 22, No. 5, October, 1985.
2. Burden, Faires, and Reynolds, Numerical Analysis, 2nd edition, 1981, pgs. 102-106, and 205-207.

Grateful Acknowledgements

1. Dr. Charles Balogh, PSU Faculty
2. Dr. John Butler, PSU Faculty
3. Dr. Forest Simmons, PCC Faculty
4. Ms. Virginia Weeks, PCC Faculty
5. Mr. Steve Wilkins, Boeing Company

An Example

$y' = \frac{dy}{dx} = -y + x^2 + 1$			APPROXIMATIONS		
exact		Heun	Runge-Kutta	Runge-Kutta	Runge-Kutta
x	y(x)	h = .1	Shampine	Shampine	Shampine
0	1	H(x)	h=.1, S(x)	h=.5, M(x)	h=.3, MS(x)
.1	1.000 3252	1.000 3333	1.000 3252	*1.000 3801	*1.000 3323
.2	1.002 5385	1.002 5850	1.002 5386	*1.002 6775	*1.002 5512
.3	1.008 3636	1.008 4728	1.008 3637	*1.008 5505	1.008 3756
.4	1.019 3599	1.019 5512	1.019 3601	*1.019 5494	*1.019 3735
.5	1.036 9387	1.037 2272	1.036 9390	1.037 1094	*1.036 9620
.6	1.062 3767	1.062 7739	1.062 3771	*1.062 5446	1.062 4077
.7	1.096 8294	1.097 3437	1.096 8299	*1.097 0413	*1.096 8730
.8	1.141 3421	1.141 9794	1.141 3426	*1.141 6509	*1.141 3969
.9	1.196 8607	1.197 6247	1.196 8613	*1.197 2843	1.196 9131
1.0	1.264 2411	1.265 1337	1.264 2418	1.264 7044	*1.264 2919

*Refers to approximations obtained by local Hermitian interpolation.

The Analytic Solution: $y(x) = -2e^{-x} + x^2 - 2x + 3$ was obtained using an integrating factor.



The preceding page contains the results of an example using this algorithm and compares them to the results from the second order Heun method of approximation.

The results were obtained from a package of 35 interactive programs I wrote for the HP-41 calculator and HP-71 computer.

I am submitting eight of the programs with my paper. Three of the eight are the Heun, the Funct, and the Tester programs.

The Heun approximating program and the Funct program can stand alone. Similarly, the Tester program with the Funct program work alone.

The 32 remaining programs all interact and use the Funct program to obtain the global and local results in the table.

For the example on the preceding page,

$$|M(.5) - y(.5)| < |H(.5) - y(.5)|.$$

This is not generally the case for the step sizes used in obtaining $M(.5)$ and $H(.5)$. The Heun interpolation is $O(h^2)$ where $h = .1$. The Shampine interpolation is $O(h^4)$ where $h = .5$. Using these step sizes the Heun method will typically generate more accurate results since $.1^2 = .01 < .0625 = .5^4$.

This is shown by the local approximation $M(x)$ when $x = .1, .2,$ and $.3$. The Hermitian interpolation is $O(h^5)$ having step size $.5$. Also, $.1^2 = .01 < .03125 = .5^5$. It follows that for local approximations Heun's method with $h = .1$ would generate better results than Hermite's method when $h = .5$.

To generate results that are more accurate than the second order scheme the step size of the fourth order method must be decreased until it satisfies $h_2^4 < h_1^2$, h_1 being the $O(h^2)$ step size and h_2 being the $O(h^4)$ step size, whereupon the global data will usually be more accurate using the fourth order method.

For the problem we have examined this will not occur until $h_2 < \sqrt{.1} \approx .316$, assuming $h_1 = .1$ is maintained.

When the above is satisfied, $h_2^4 < h_1^2$, the data generated by the $O(h^4)$ method typically will be more accurate than the data generated by the $O(h^2)$ method.

Also, since the step size for the Hermite interpolation and the Runge-Kutta integration is the same in this paper, the local Hermite data denoted with an asterisk will be of the same order accuracy as that of the Runge-Kutta scheme. This is because the Runge-Kutta scheme generates the data necessary to obtain the Hermite polynomials which are necessary for the local approximations.

The table illustrates that the results from Shampine's method of integration when $h = .5$ are less accurate than those obtained from Heun's method of integration when $h = .1$. When $h = .3$ for Shampine's method, $.3 < \sqrt{.1}$ and it follows that the results from Shampine's algorithm will usually be more accurate than the results from Heun's algorithm with $h = .1$. This is seen in the table for this example.

We have presented the algorithm for a fourth order Runge-Kutta approximating schema that applies the method Shampine described in the article. It is the re-initialization that makes this schema unique. We can obtain an approximation for non-mesh points that is continuous in both the 0th and 1st derivatives at the mesh points.

As Shampine stated, "Adams and BDF codes vary their order as well as their step size automatically." By, "dodging the issue," the user is forced to choose between jumps or corners at the mesh points. We have shown that the method advocated by Shampine confronts the problem and offers the user of the algorithm a viable alternative to the inadequacies of the traditional methods.