NASA/TM—2000-209944

A Very High Order, Adaptable MESA Implementation for Aeroacoustic Computations

Rodger W. Dyson and John W. Goodrich
Glenn Research Center, Cleveland, Ohio

April 2000

NASA/TM—2000-209944

A Very High Order, Adaptable MESA Implementation for Aeroacoustic Computations

Rodger W. Dyson and John W. Goodrich
Glenn Research Center, Cleveland, Ohio

National Aeronautics and
Space Administration

Glenn Research Center

April 2000

This report is a formal draft or working paper, intended to solicit comments and ideas from a technical peer group.

This report contains preliminary findings, subject to revision as analysis proceeds.

Available from

NASA Center for Aerospace Information
7121 Standard Drive
Hanover, MD 21076
Price Code: A03

National Technical Information Service
5285 Port Royal Road
Springfield, VA 22100
Price Code: A03

A Very High Order, Adaptable MESA Implementation for Aeroacoustic Computations

Rodger W. Dyson and John W. Goodrich
National Aeronautics and Space Administration
Glenn Research Center
Cleveland, Ohio 44135

Abstract

Since computational efficiency and wave resolution scale with accuracy, the ideal would be infinitely high accuracy for problems with widely varying wavelength scales. Currently, many of the computational aeroacoustics methods are limited to 4^{th} order accurate Runge-Kutta methods in time which limits their resolution and efficiency. However, a new procedure for implementing the Modified Expansion Solution Approximation (MESA) schemes, based upon Hermitian divided differences, is presented which extends the effective accuracy of the MESA schemes to 57^{th} order in space and time when using 128 bit floating point precision. This new approach has the advantages of reducing round-off error, being easy to program, and is more computationally efficient when compared to previous approaches. Its accuracy is limited only by the floating point hardware. The advantages of this new approach are demonstrated by solving the linearized Euler equations in an open bi-periodic domain. A 500^{th} order MESA scheme can now be created in seconds, making these schemes ideally suited for the next generation of high performance 256-bit (double quadruple) or higher precision computers. This ease of creation makes it possible to adapt the algorithm to the mesh in time instead of its converse: this is ideal for resolving varying wavelength scales which occur in noise generation simulations. And finally, the sources of round-off error which effect the very high order methods are examined and remedies provided that effectively increase the accuracy of the MESA schemes while using current computer technology.

1 Introduction

Predicting the sources of jet noise requires computational methods that are orders of magnitude more efficient and that provide very high resolution. This is accomplished numerically with very high accuracy, adaptable, explicit methods whose benefits are as follows:

- High accuracy methods are more efficient and provide finer resolution of the physics [5];
- Adaptable methods can adjust their accuracy to resolve steep gradients while avoiding the complexities of mesh adaptation;
- And, explicit methods permit highly parallel/scalable computations by minimizing interprocessor communication [4].

The proposed approach enables the MESA schemes [5] to accomplish those objectives by making them simple to program, adapt, and compile while simultaneously reducing floating point operations and round-off error.

MESA schemes can be viewed as a multidimensional, higher order extension of Lax-Wendroff schemes that incorporate more of the physics (via cross-derivative information) necessary to more accurately propagate waves along their characteristic surfaces. The MESA schemes require essentially two procedures; a spatial interpolation followed by a time advance. Previously, the second step was implemented using a recursive definition [4] which enabled arbitrarily high accuracy in time. However, the spatial interpolation step required computer algebra [3] for the symbolic creation of one-dimensional interpolants, effectively limiting the accuracy of MESA schemes to 29^{th} order because of the computational complexity in producing its symbolic form. This final limitation is removed in this work by replacing the one-dimensional symbolic interpolant with a simple and efficient form of the Hermitian divided difference interpolant. A key finding of this paper is that all the spatial derivatives of a Hermitian divided difference interpolant, at the midpoint of a two-point, multidimensional stencil, have a simple algebraic expression which eliminates the need for computer algebra tools. Since small stencils have many advantages [7] such as better resolution and ease of boundary implementations, using this new form of Hermitian divided differences with two-point Hermitian MESA schemes would appear ideal.

Divided differences have been used to interpolate data for many years dating back to Isaac Newton, [11] and [2]. With the advent of digital computers, divided differences have been replaced by splines since polynomial interpolations tend to oscillate at higher orders. In addition, Hermitian methods have not been extensively used due to the difficulty of obtaining the derivatives of the function being approximated [14]. However, in this work, polynomial oscillations are eliminated since only a single interval is used and the derivatives of the function at the endpoints of this interval are completely prescribed. Also, round-off error is reduced for high-order interpolations since guard digits are introduced into the tableau [13]. And Hermitian divided difference interpolations coincidentally use the same data found in the stencil of a $2 * s + 1$ order MESA scheme $(f, \frac{\partial f(x)}{\partial x}, \frac{\partial^2 f(x)}{\partial x^2}, \ldots, \frac{\partial^s f(x)}{\partial x^s})$. For these reasons, Hermitian (Birkhoff [10]) divided differences are used to great advantage here.

This paper first describes the new approach to spatial interpolation in one-dimension and then extends it to 2 × 2 Hermitian stencils. Next, the linearized Euler equations are solved in a bi-periodic open domain by applying this new interpolation method to the MESA schemes. The error and efficiency of the new approach is then compared with the previously best known approach. And finally, various techniques are shown for improving the effective accuracy of very high order methods (> 30) when computer precision is limited.

2 Two-Point One-Dimensional Hermitian Divided-Difference Interpolation

We will now provide an overview of Newton's interpolation method based upon divided differences.

Let $x_i : i = 0, 1, \ldots, n$ be any (n+1) distinct points of [a,b] and let f be a differentiable function, $C^d[a, b]$. The coefficient of x^n in the polynomial $p \in P_n$ that satisfies the conditions

$$\frac{\partial p(x_i)}{\partial x} = \frac{\partial f(x_i)}{\partial x}, i = 0, 1, \ldots, n, \tag{1}$$

is defined to be a divided difference of order n [13]. These divided differences, once determined, completely define the spatial interpolant satisfying equation 1.

A convenient mnemonic commonly used to determine the divided differences is shown in figure 3 and is referred to as a divided difference tableau. In this figure is the tableau for a fifth order one-dimensional interpolant. When Hermitian data (the data and their spatial derivatives) are used for interpolation [13], the tableau is formed by inserting all the known data into each column of the tableau as shown in figure 4 for the case of a two-point Hermitian stencil with three data elements per grid point $(f(x, y), f_x(x, y), f_{xx}(x, y))$. In general, for a two-point

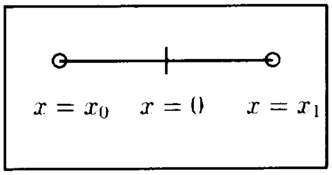


Figure 1: Two Point One-Dimensional Stencil

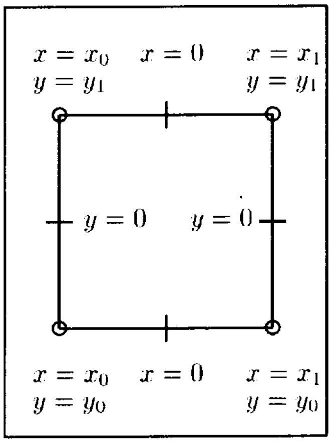
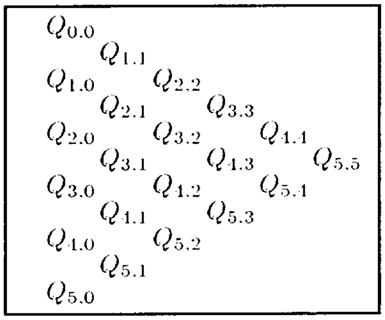


Figure 2: Two Point Two-Dimensional Stencil

Hermitian stencil in which the data: $f(x,y), f_x(x,y), f_{xx}(x,y), \ldots, \frac{\partial^s f(x,y)}{\partial x^s}$ is known at both grid points (x=x_0 and x=x_1) in figure 1, the following procedure will correctly place this data into the tableau.

$$Do[Do[Q[i,j] = \frac{f^j(x_0)}{j!}; Q[i+s+1,j] = \frac{f^j(x_1)}{j!};, \{j,0,i\}], \{i,0,s\}]; \qquad (2)$$

With this initial data inserted into the tableau and the distance between both points defined by Δh, the rest of the tableau is constructed using [2]:



$Q_{0.0}$					
	$Q_{1.1}$				
$Q_{1.0}$		$Q_{2.2}$			
	$Q_{2.1}$		$Q_{3.3}$		
$Q_{2.0}$		$Q_{3.2}$		$Q_{4.4}$	
	$Q_{3.1}$		$Q_{4.3}$		$Q_{5.5}$
$Q_{3.0}$		$Q_{4.2}$		$Q_{5.4}$	
	$Q_{4.1}$		$Q_{5.3}$		
$Q_{4.0}$		$Q_{5.2}$			
	$Q_{5.1}$				
$Q_{5.0}$					

Figure 3: Divided-Difference Tableau for c2o2 MESA scheme

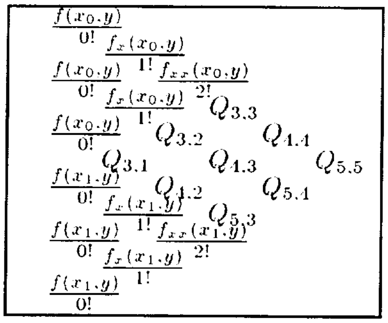


Figure 4: Loaded Divided-Difference Tableau for c2o2 MESA scheme, X-Direction

$$Do[Do[Q[i,j] = \frac{(Q[i,j-1] - Q[i-1,j-1])}{\Delta h}, \{j, i-s, i\}], \{i, s+1, 2*s+1\}]; \tag{3}$$

And with this tableau, the interpolating Hermitian polynomial on a Hermitian stencil with two points and $s+1$ data elements (primitive variable and its spatial derivatives) per grid point may be evaluated with Newton's interpolatory divided difference formula [2]:

$$f(x) = \sum_{i=0}^{2*(s+1)-1} Q(i,i) \prod_{j=0}^{i-1} (x - x_j) \tag{4}$$

which may be rewritten as:

$$f(x) = \sum_{i=0}^{s} Q(i,i)(x - x_0)^i + \sum_{i=s+1}^{2(s+1)-1} Q(i,i)(x - x_0)^{s+1}(x - x_1)^{i-(s+1)} \tag{5}$$

Hermitian MESA schemes require evaluating this polynomial at the center of each stencil to interpolate the solution data and their spatial derivatives. In the general case in which a MESA scheme of arbitrary accuracy is used, the spatial derivatives of equation 4 become very complicated and require computer algebra tools for their solution. This limits the accuracy and adaptability of these methods.

For example, a 49^{th} order MESA scheme in one-dimension requires 25 data elements per grid point. But to time advance all those elements requires determining the derivatives of equation 4 up to the 50^{th} order. The product terms in that equation will double after each differentiation by the product rule of differentiation; this produces equations with approximately $2^{50} \approx 10^{16}$ terms. Besides quickly exhausting memory resources of most computers, this also takes too much time to calculate and makes it difficult to modify the accuracy of the numerical scheme in real time to accommodate steep gradients.

Fortunately, by using a two-point stencil and a reformulation of equation 5, it is possible to efficiently calculate all the necessary spatial derivatives without the use of computer algebra.

3 Fundamental Result: Direct Interpolation of Spatial Derivatives at Center of Two Point Hermitian Stencil

A $2*s+1$ order Hermitian MESA scheme (labeled c2os) will contain $(s+1)$, $(s+1)^2$, or $(s+1)^3$ data elements per grid point for each primitive variable in one, two, or three-dimensions respectively. The MESA scheme requires all these data elements to be advanced in time. Accomplishing this requires interpolating $2(s+1), 4(s+1)^2$, or $8(s+1)^3$ spatial derivatives at the center of the stencil for each primitive variable in one, two, or three spatial dimensions respectively. Each of those interpolations normally requires evaluating the derivative of equation 4 at $x = 0$. Because of the product term in this equation, higher order derivatives of this equation become complicated, as mentioned. However, the following main result of this paper provides an alternative formulation for the interpolation of the spatial derivatives at the center of a two-point Hermitian stencil.

$$\frac{\partial^{dx} f(x=0)}{\partial x^{dx}} = \sum_{i=dx}^{s} \left[Q(i,i) \left(\frac{i!}{(i-dx)!} \right) (-x_0)^{(i-dx)} \right] + \sum_{i=s+1}^{2(s+1)-1} \left[Q(i,i) Z(i,s,dx,x_0,x_1) \right] \quad (6)$$

where Z is defined as:

$$Z(i,s,dx,x_0,x_1) = \sum_{r=0}^{dx} \left[\left(\frac{dx!}{(dx-r)!r!} \right) (-x_0)^{(s+1-r)} (-x_1)^{(i-(s+1)-dx+r)} P_1(i,s,dx,r) P_2(s,r) \right] \quad (7)$$

with P_1 and P_2 defined as:

$$P_1(i,s,dx,r) = \prod_{e=0}^{dx-1-r} [i-(s+1)-e] \quad (8)$$

and

$$P_2(s,r) = \prod_{k=0}^{r-1} [s+1-k] \quad (9)$$

,respectively.

This form can be rewritten as :

$$\frac{\partial^{dx} f(x=0)}{\partial x^{dx}} = \sum_{i=dx}^{2s+1} Q(i,i)\, coef(i,dx) \quad (10)$$

where the function coef(i,dx) is predefined as:

$$\begin{aligned}
&Do[Do[coef[i,dx] = \frac{i!}{(i-dx)!}(-x_0)^{(i-dx)}, \{i,dx,s\}], \{dx,0,(2s+1)\}] \\
&Do[Do[\\
&coef[i,dx] = \\
&\quad \textstyle\sum_{r=0}^{dx}[\frac{dx!}{(dx-r)!r!}(-x_0)^{(s+1-r)}(-x_1)^{(i-s-1-dx+r)} * \qquad (11) \\
&\qquad Product[(i-s-1-e), \{e,0,dx-1-r\}] * \\
&\qquad\qquad Product[(s+1-k), \{k,0,r-1\}]] \\
&\qquad\quad , \{i,s+1,2s+1\}], \{dx,0,(2s+1)\}];
\end{aligned}$$

The function coef(i,dx) in equation 10 is independent of space and time and needs computed only once. Thereafter only the divided difference terms Q(i,i) must be evaluated for each stencil at each time step using equation 3.

4 Cost Comparison

As mentioned, this new approach eliminates the need for using computer algebra to produce high order MESA schemes; this reduces the initial cost of programming the MESA schemes and increases its possible accuracy and adaptability. In addition, another unanticipated benefit of this new approach is fewer floating point operations are required as compared to the best previous approaches. For example, the total cost to evaluate all of the spatial derivatives required in a $2 * s + 1$ order MESA scheme (c2os) is the sum of the cost to compute the divided difference terms, Q's, using equation 3 and the cost of evaluating all of the spatial derivatives ($\frac{\partial^{d_x} f(x=0)}{\partial x^{d_x}}$) using equation 10. Therefore the total cost of the new approach is:

$$((1+s)^2) + (3 + 5s + 2s^2) = 4 + 7s + 3s^2 \qquad (12)$$

multiplications.

In the previously best known procedure [4], in which computer algebra is used to generate the one-dimensional interpolant, each spatial derivative required at most $2s + 2$ multiplications, and there were $2s + 2$ terms to be evaluated in one dimension resulting in a total cost of:

$$(2s+2)^2 = 4 + 8s + 4s^2 \qquad (13)$$

multiplications.

Therefore the new Hermitian (Birkhoff) divided difference form requires

$$s + s^2 \qquad (14)$$

fewer multiplications. However, the assumption that calculating each spatial derivative under the old approach required $2s + 2$ operations is an upper limit and in practice certain algebraic cancellations may reduce that number. At higher accuracy this upper limit is more likely since the equations become large and cancellations are more difficult.

5 Two-Point Two-Dimensional Hermitian Divided-Difference Interpolation

It is necessary to perform a multidimensional interpolation for the MESA schemes in two and three dimensions. While this could possibly be accomplished using other multidimensional divided difference techniques [12], we will use the tensor product approach as in Dyson [4] since this permits a detailed comparison of the new and old approaches.

5.1 Tensor Product Approach Overview

The tensor product approach interpolates all the spatial derivatives required for the MESA schemes by performing a series of one-dimensional interpolations [4]. Each one-dimensional interpolation requires a new divided difference tableau to be generated using equation 3. A set of one-dimensional interpolations is first performed in the x-direction to interpolate the data $\frac{\partial^{i+j} f(x=0,y)}{\partial x^i \partial y^j} \forall i : i = 0, 1, \ldots, s,\ and\ j = 0, 1, 2, \ldots, 2 * s + 1$ at $y = y_0$ and $y = y_1$. And then, by interpolating in the y-direction using only these interpolated values, the following terms are found:

$$\frac{\partial^{i+j} f(x=0, y=0)}{\partial x^i y^j} \forall i, j : i, j = 0, 1, 2, \ldots, 2 * s + 1 \qquad (15)$$

using the coordinate system in figure 2. These terms are required for time advancing all the data on each grid point, namely:

$$\frac{\partial^{i+j} f(x, y)}{\partial x^i y^j} \forall i, j : i, j = 0, 1, 2, \ldots, s \qquad (16)$$

at grid points $(x_0, y_0), (x_0, y_1), (x_1, y_0), and (x_1, y_1)$ as shown in figure 2.

This tensor product procedure is extensible to any size stencil, but the Hermitian divided difference one-dimensional interpolation as developed in this paper is limited to two-point interpolations.

5.2 Fifth Order Two-Dimensional Interpolation Example

The combination of one-dimensional two-point Hermitian divided difference interpolation with multidimensional tensor product extensions is best understood with a simple example. We will completely describe the process required to interpolate the data on a two point two-dimensional stencil shown in figure 2 for the 5^{th} order MESA scheme, c2o2.

The c2o2 MESA scheme in two dimensions, will contain the data: f, f_x, f_{xx}, f_y, f_{yx}, f_{yxx}, f_{yy}, f_{yyx}, f_{yyxx} at all four grid point locations as shown in figure 2 for each primitive variable (pressure, u-velocity, v-velocity). Using those 36 pieces of information for each primitive variable, the two-dimensional spatial interpolant of the form:

$$f(x, y) = \sum_{i=0}^{5}\sum_{j=0}^{5} cf(i, j)x^i y^j \tag{17}$$

is created by finding the values of the $cf(i, j)$ coefficients where

$$cf(i, j) = \frac{f^{(i,j)}}{i!j!} \tag{18}$$

These coefficients are found by first performing one-dimensional interpolations of the form:

$$f(x) = \sum_{i=0}^{5} cf(i)x^i \tag{19}$$

where

$$cf(i) = \frac{f^{(i)}(x = 0, y)}{i!} = \frac{\partial f^i(x = 0, y)}{\partial x^i}\left(\frac{1}{i!}\right) \tag{20}$$

By letting f(x) in equation 19 be replaced by $f(x)$, $f_y(x)$, and $f_{yy}(x)$, respectively, then

$$cf(i) = \frac{\partial^j f^{(i)}(x = 0, y)}{\partial y^j}\frac{1}{i!} = \frac{\partial^{(i+j)} f(x = 0)}{\partial x^i \partial y^j}\left(\frac{1}{i!}\right) \tag{21}$$

for j=0, 1, and 2, respectively; And, i=0,1,2,3,4, and 5.

Thus, it is possible to interpolate all the values, cf(i) for j=0,1, and 2 at the location $(x = 0, y = y_0)$ shown in figure 2 using the three one-dimensional interpolants of the form of equation 19. This is repeated for the cf(i) values at the location $(x = 0, y = y_1)$ for each j=0, 1, and 2 for a total of six one-dimensional interpolations.

At this point, we have the two sets of interpolated values:

$$\frac{\partial^{i+j} f(x = 0, y)}{\partial x^i \partial y^j}\left(\frac{1}{i!}\right) \tag{22}$$

for y=y_0 and y_1; and for i=0,1,2,3,4,5; and j=0,1, and 2.

This data may be used to evaluate the derivatives in equation 15 at the center of the stencil. This requires a second set of one-dimensional divided difference interpolations in the y-direction using only this data.

Interpolating in the y-direction is accomplished with the polynomial:

$$f(x=0,y)=\sum_{j=0}^{5} cf(j)y^{j} \tag{23}$$

where

$$cf(j)=\frac{\partial^{j} f(x=0,y=0)}{\partial y^{j}}\left(\frac{1}{j!}\right) \tag{24}$$

This time however, we will interpolate the functions: $f, f_x, f_{xx}, f_{xxx}, f_{xxxx}$, and f_{xxxxx} by performing the following substitution into equation 24:

$$f(x=0,y)=\frac{\partial^{i} f(x=0,y)}{\partial x^{i}}\left(\frac{1}{i!}\right) \tag{25}$$

for i=0,1,2,3,4, and 5 so that

$$cf(j)=\frac{\partial^{i+j} f(x=0,y=0)}{\partial x^{i}\partial y^{j}}\left(\frac{1}{i!j!}\right) \tag{26}$$

Therefore this requires six more one-dimensional divided difference interpolations for a total of twelve when including the six horizontal interpolations. The substitution of equation 25 into equation 24 reuses the data from the x-direction interpolation, thereby introducing efficiencies not found with other multidimensional interpolation procedures. Note that this efficiency is only possible if the data in equation 16 are available at each grid point so that a symmetry of the spatial derivatives exists in all dimensions because both the tensor product and Hermitian divided differences require this complete set of derivative information. Fortunately, the two-point Hermitian MESA schemes (c2os) provide exactly this information.

5.2.1 Horizontal Interpolation Procedures

Applying the above concepts can be reduced to a few simple steps. Interpolating the intermediate values at $(0, y_0)$ and $(0, y_1)$ in figure 2 is accomplished by:

- First, loading the known data from the stencil into the tableau shown in figure 4 to interpolate the data at $y = y_0$. The tableau is loaded with the function f, f_x, and f_{xx} data contained at the two grid point locations x_0 and x_1 as indicated in figure 1.
- Second, build the divided difference tableau as described in equation 3.
- Third, evaluate the spatial derivatives $\frac{\partial^{i} f(x=0,y=y0)}{\partial x^{i}}$ for i=0,1,2,3,4, and 5 using equation 10.
- Fourth, repeat these three steps by substituting $f(x)$ with $f = f$, f_y, and f_{yy}.
- Fifth, repeat these four steps at $y = y_1$.

After these procedures, we have calculated the data:

$$\frac{\partial^{i+j} f(x=0,y)}{\partial x^{i}\partial y^{j}} \forall i : i = 1,2,3,4,5 and \forall j : j = 0,1,2 \tag{27}$$

at grid coordinates $(0, y_0)$ and $(0, y_1)$ as labeled in figure 2.

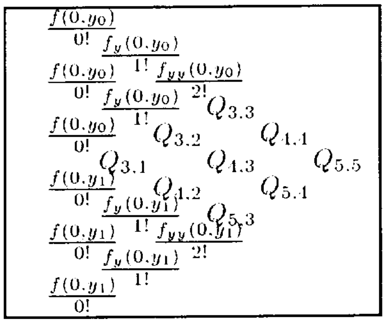


Figure 5: Loaded Divided-Difference Tableau for c2o2 MESA scheme, Y-Direction

5.2.2 Vertical Interpolation Procedures

Next, the data shown in equation 27 is used to perform the Hermitian divided difference interpolation in the y-direction by:

- First loading that data into the tableau as shown in figure 5 to interpolate the intermediate data of the last step along the line $x = 0$. The tableau is loaded with the function f, f_y, and f_{yy} data contained at the two grid point locations $(0, y_0)$ and $(0, y_1)$ as indicated in figure 2.
- Second, build the divided difference tableau as described in equation 3.
- Third, evaluate the spatial derivatives $\frac{\partial^j f(x=0,y=0)}{\partial y^j}$ for j=0,1,2,3,4, and 5 using equation 10.
- Fourth, repeat these three steps by substituting $f(x)$ with $f = f$, f_x, f_{xx}, f_{xxx}, f_{xxxx}, and f_{xxxxx} which are the intermediate data previously evaluated using the horizontal interpolation procedures.

After these steps, all the spatial derivatives of f(x,y) necessary for advancing the data on the grid is available for the MESA scheme.

6 Results

For the purposes of comparing this new approach with the old approach, it is necessary to divide the results from the new approach by a factorial term. This is because the coefficients for the previous method [4] were equivalent to using the equation:

$$f(x) = \sum_{i=0}^{2*s+1} cf(i)x^i \tag{28}$$

in which the factorial term is included in these cf(i) coefficients instead of the form:

$$f(x) = \sum_{i=0}^{2*s+1} \frac{cf(i)}{i!}x^i \tag{29}$$

in which the factorial term is not included and therefore $cf(i) = \frac{\partial^i f(x)}{\partial x^i}$.

With this correction factor a direct comparison is possible between both approaches for very high accuracy by using automatic code generation [6]. In particular, the relative efficiency and accuracy of applying both interpolation methods to the same wave propagation problem are measured.

6.1 Problem Definition

Wave propagation is described by the linearized Euler equations and its correct simulation in time is important in many aeroacoustic applications. We will solve the bi-periodic open domain problem in which the physical domain is a unit square ($[-1,1] \times [-1,1] \times [0,T]$). The solution of the linearized Euler equations in this case is assumed to be y-periodic (top and bottom of box repeat) and x-periodic (left and right sides of box repeat). Using separation of variables with periodic boundary conditions, on the following linearized Euler equation system with a constant mean convection velocity vector (M_x, M_y):

$$\begin{aligned} \frac{\partial u}{\partial t} + M_x \frac{\partial u}{\partial x} + M_y \frac{\partial u}{\partial y} + \frac{\partial p}{\partial x} &= 0, \\ \frac{\partial v}{\partial t} + M_x \frac{\partial v}{\partial x} + M_y \frac{\partial v}{\partial y} + \frac{\partial p}{\partial y} &= 0, \\ \frac{\partial p}{\partial t} + M_x \frac{\partial p}{\partial x} + M_y \frac{\partial p}{\partial y} + \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} &= 0, \end{aligned} \tag{30}$$

with the boundary conditions :

$$\begin{aligned} p(1,y,t) &= p(-1,y,t) \\ u(1,y,t) &= u(-1,y,t) \\ v(1,y,t) &= v(-1,y,t) \\ p(x,1,t) &= p(x,-1,t) \\ u(x,1,t) &= u(x,-1,t) \\ v(x,1,t) &= v(x,-1,t) \end{aligned}$$

provides the following analytical solution:

$$p(x,y,t) = \cos(\pi t\sqrt{2})\, \sin(\pi\,(-\,(M_x t) + x))\, \sin(\pi\,(-\,(M_y t) + y)) \tag{31}$$

$$u(x,y,t) = -\frac{\cos(\pi\,(-\,(M_x t) + x))\sin(\pi t\sqrt{2})\sin(\pi\,(-\,(M_y t) + y))}{\sqrt{2}} \tag{32}$$

$$v(x,y,t) = -\frac{\cos(\pi\,(-\,(M_y t) + y))\sin(\pi t\sqrt{2})\sin(\pi\,(-\,(M_x t) + x))}{\sqrt{2}} \tag{33}$$

6.2 Numerical Results

The results of these numerical experiments with 8 grid points per wavelength are shown for double-precision (64-bit reals) in tables 1 and 2 and for quadruple precision (128-bit reals) in table 3. The time to complete the simulation using each approach is a measure of relative efficiency and is shown for each MESA scheme tested. In addition, the results for both approaches are placed above and below each other for easy comparison. The seven column headings are defined as:

N Number of time steps.

T Total time elapsed in simulation.

maxperr The maximum absolute error in the pressure.

l1perr A measure of the average error (L1 norm) .

phmax Maximum pressure occurring in domain.

phmin Minimum pressure occurring in domain.

e-ratio The change in the total energy content of the system as a ratio $\frac{new}{initial}$ (should be one for no change).

Despite the cost analysis in section 4 showing a slight advantage to the new approach for all MESA schemes, numerical experiments show the old approach is actually faster for methods less than about 15^{th} order. This is likely due to the length of the one-dimensional equations being less than the worst case used in the analysis. However, for higher order methods the new approach is modestly faster while using double-precision calculations and it is significantly faster when quadruple-precision is used (see table 3).

Notice in table 1 that the new approach begins demonstrating less round-off error at approximately 11^{th} order accuracy. And at 15^{th} order accuracy the new approach still maintains essentially no growth in the energy compared to the old approach in table 2. And at 17^{th} order both methods begin introducing significant round-off errors into the overall energy.

In quadruple precision, both approaches appear similar at 15^{th} order, however the new approach is about 50 percent faster. And at 21^{st} order accuracy the round-off errors are higher in the old approach while the new approach is more efficient. By the time 27^{th} order accuracy is reached, both approaches are showing significant round-off error in the total system energy.

It was not possible to create a 33^{rd} order or higher MESA scheme using the old approach, but the new approach appears to maintain accuracy up to 57^{th} order in quadruple precision before round-off error becomes too excessive. The sources of this round-off error are explained in the appendix A.

N	T	maxperr	l1perr	phmax	phmin	e-ratio
OLD APPROACH ABOVE, NEW APPROACH BELOW						
c2o2: Old Approach Time=254.79 , New Approach Time=290.39						
200	1.00000E+01	9.07553E-04	1.36338E-03	8.95490E-01	-8.95490E-01	9.97694E-01
200	1.00000E+01	9.07553E-04	1.36338E-03	8.95490E-01	-8.95490E-01	9.97694E-01
2000	1.00000E+02	2.06528E-03	3.24237E-03	2.40990E-01	-2.40990E-01	9.79382E-01
2000	1.00000E+02	2.06528E-03	3.24237E-03	2.40990E-01	-2.40990E-01	9.79382E-01
20000	1.00000E+03	7.40171E-02	1.11612E-01	7.04408E-01	-7.04408E-01	8.14061E-01
20000	1.00000E+03	7.40171E-02	1.11612E-01	7.04408E-01	-7.04408E-01	8.14061E-01
c2o3: Old Approach Time=572.39, New Approach Time=641.52						
200	1.00000E+01	2.04781E-06	2.99855E-06	8.96396E-01	-8.96396E-01	9.99996E-01
200	1.00000E+01	2.04781E-06	2.99855E-06	8.96396E-01	-8.96396E-01	9.99996E-01
2000	1.00000E+02	5.51277E-06	1.08654E-05	2.43050E-01	-2.43050E-01	9.99952E-01
2000	1.00000E+02	5.51277E-06	1.08654E-05	2.43050E-01	-2.43050E-01	9.99952E-01
20000	1.00000E+03	1.77378E-04	2.67937E-04	7.78247E-01	-7.78247E-01	9.99543E-01
20000	1.00000E+03	1.77378E-04	2.67937E-04	7.78247E-01	-7.78247E-01	9.99543E-01
c2o4: Old Approach Time=1108.65, New Approach Time=1311.10						
200	1.00000E+01	2.86266E-09	4.24757E-09	8.96398E-01	-8.96398E-01	1.00000E+00
200	1.00000E+01	2.86266E-09	4.24757E-09	8.96398E-01	-8.96398E-01	1.00000E+00
2000	1.00000E+02	8.95464E-09	1.86641E-08	2.43055E-01	-2.43055E-01	1.00000E+00
2000	1.00000E+02	8.95463E-09	1.86641E-08	2.43055E-01	-2.43055E-01	1.00000E+00
20000	1.00000E+03	2.45441E-07	3.84170E-07	7.78425E-01	-7.78425E-01	9.99999E-01
20000	1.00000E+03	2.45441E-07	3.84169E-07	7.78425E-01	-7.78425E-01	9.99999E-01
c2o5: Old Approach Time=2034.22, New Approach Time=2533.79						
200	1.00000E+01	2.70040E-12	4.09499E-12	8.96398E-01	-8.96398E-01	9.99999E-01
200	1.00000E+01	2.69773E-12	4.09191E-12	8.96398E-01	-8.96398E-01	1.00000E+00
2000	1.00000E+02	2.00462E-11	3.45419E-11	2.43055E-01	-2.43055E-01	1.00000E+00
2000	1.00000E+02	2.00432E-11	3.45456E-11	2.43055E-01	-2.43055E-01	1.00000E+00
20000	1.00000E+03	1.53291E-09	2.87923E-09	7.78425E-01	-7.78425E-01	1.00000E+00
20000	1.00000E+03	1.53274E-09	2.87917E-09	7.78425E-01	-7.78425E-01	1.00000E+00

Table 1: Cost and Round-off Error Comparison of New and Old 2D Interpolation Methods With 8 Grid Points Per Wavelength: Double Precision Computations

N	T	maxperr	l1perr	phmax	phmin	c-ratio
OLD APPROACH ABOVE. NEW APPROACH BELOW						
c2o6: Old Approach Time=3372.85. New Approach Time=3978.52						
200	1.00000E+01	4.72955E-14	5.58290E-14	8.96398E-01	-8.96398E-01	1.00094E+00
200	1.00000E+01	2.56462E-14	4.69778E-14	8.96398E-01	-8.96398E-01	1.00002E+00
2000	1.00000E+02	1.47465E-11	2.62678E-11	2.43055E-01	-2.43055E-01	9.98917E-01
2000	1.00000E+02	1.47443E-11	2.62938E-11	2.43055E-01	-2.43055E-01	1.00004E+00
20000	1.00000E+03	1.45736E-09	2.73856E-09	7.78425E-01	-7.78425E-01	1.00010E+00
20000	1.00000E+03	1.45717E-09	2.73851E-09	7.78425E-01	-7.78425E-01	1.00001E+00
c2o7: Old Approach Time= 6557.54. New Approach Time=5824.79						
200	1.00000E+01	7.13873E-14	1.17613E-13	8.96398E-01	-8.96398E-01	6.61955E+02
200	1.00000E+01	2.69784E-14	4.96476E-14	8.96398E-01	-8.96398E-01	1.04582E+00
2000	1.00000E+02	1.48251E-11	2.63057E-11	2.43055E-01	-2.43055E-01	7.81400E+02
2000	1.00000E+02	1.47365E-11	2.62886E-11	2.43055E-01	-2.43055E-01	1.08933E+00
20000	1.00000E+03	1.45761E-09	2.73870E-09	7.78425E-01	-7.78425E-01	6.05388E+02
20000	1.00000E+03	1.45698E-09	2.73832E-09	7.78425E-01	-7.78425E-01	1.03607E+00
c2o8: Old Approach Time=7424.47 , New Approach Time=6443.41						
200	1.00000E+01	2.19713E-13	3.11883E-13	8.96398E-01	-8.96398E-01	8.09897E+09
200	1.00000E+01	2.47580E-14	4.59314E-14	8.96398E-01	-8.96398E-01	1.23692E+05
2000	1.00000E+02	1.46503E-11	2.62826E-11	2.43055E-01	-2.43055E-01	8.25499E+09
2000	1.00000E+02	1.47489E-11	2.63057E-11	2.43055E-01	-2.43055E-01	1.10213E+05
20000	1.00000E+03	1.45885E-09	2.73874E-09	7.78425E-01	-7.78425E-01	7.70854E+09
20000	1.00000E+03	1.45711E-09	2.73847E-09	7.78425E-01	-7.78425E-01	1.38970E+05
c2o9: Old Approach Time=10112.20 , New Approach Time=11168.74						
200	1.00000E+01	8.13960E-13	1.04409E-12	8.96398E-01	-8.96398E-01	5.13309E+17
200	1.00000E+01	2.58682E-14	4.62761E-14	8.96398E-01	-8.96398E-01	2.81314E+11
2000	1.00000E+02	1.52230E-11	2.65629E-11	2.43055E-01	-2.43055E-01	5.05941E+17
2000	1.00000E+02	1.47471E-11	2.63039E-11	2.43055E-01	-2.43055E-01	4.05821E+11
20000	1.00000E+03	1.46161E-09	2.73741E-09	7.78425E-01	-7.78425E-01	4.91307E+17
20000	1.00000E+03	1.45712E-09	2.73848E-09	7.78425E-01	-7.78425E-01	3.22536E+11
c2o10: Old Approach Time=17783.09 , New Approach Time=11433.52						
200	1.00000E+01	2.76479E-12	4.08680E-12	8.96398E-01	-8.96398E-01	3.58499E+25
200	1.00000E+01	2.62013E-14	4.66777E-14	8.96398E-01	-8.96398E-01	2.85350E+18
2000	1.00000E+02	2.05443E-11	2.66944E-11	2.43055E-01	-2.43055E-01	2.28614E+25
2000	1.00000E+02	1.47421E-11	2.63037E-11	2.43055E-01	-2.43055E-01	3.55611E+18
20000	1.00000E+03	1.47549E-09	2.73275E-09	7.78425E-01	-7.78425E-01	2.72428E+25
20000	1.00000E+03	1.45712E-09	2.73848E-09	7.78425E-01	-7.78425E-01	3.55940E+18
c2o11: Old Approach Time=1803.33 , New Approach Time= 1869.41						
200	1.00000E+01	9.20091E-12	1.09857E-11	8.96398E-01	-8.96398E-01	4.76527E+32
200	1.00000E+01	2.85327E-14	4.66790E-14	8.96398E-01	-8.96398E-01	7.71540E+24
2000	1.00000E+02	3.32458E-11	4.56052E-11	2.43055E-01	-2.43055E-01	6.81984E+32
2000	1.00000E+02	1.47503E-11	2.63034E-11	2.43055E-01	-2.43055E-01	9.88028E+24

Table 2: Cost and Round-off Error Comparison of New and Old 2D Interpolation Methods With 8 Grid Points Per Wavelength: Double Precision Computations

N	T	maxperr	l1perr	phmax	phmin	e-ratio
OLD APPROACH ABOVE, NEW APPROACH BELOW						
c2o7: Old Approach Time=127.48 , New Approach Time=89.34						
2	1.00000E-01	2.09573E-20	3.26076E-20	8.53711E-01	-8.53711E-01	1.00000E+00
2	1.00000E-01	2.09573E-20	3.26076E-20	8.53711E-01	-8.53711E-01	1.00000E+00
20	1.00000E+00	9.75947E-20	1.53880E-19	2.64616E-01	-2.64616E-01	1.00000E+00
20	1.00000E+00	9.75947E-20	1.53880E-19	2.64616E-01	-2.64616E-01	1.00000E+00
c2o10: Old Approach Time=359.67, New Approach Time= 230.74						
2	1.00000E-01	1.76261E-30	1.51294E-30	8.53711E-01	-8.53711E-01	9.99954E-01
2	1.00000E-01	5.54668E-31	8.26561E-31	8.53711E-01	-8.53711E-01	1.00000E+00
20	1.00000E+00	3.00584E-29	4.99977E-29	2.64616E-01	-2.64616E-01	9.99980E-01
20	1.00000E+00	4.39112E-30	8.13865E-30	2.64616E-01	-2.64616E-01	1.00000E+00
c2o13: Old Approach Time=806.21, New Approach Time= 376.07						
2	1.00000E-01	2.87010E-29	4.01517E-29	8.53711E-01	-8.53711E-01	1.88808E+16
2	1.00000E-01	3.38964E-32	4.22501E-32	8.53711E-01	-8.53711E-01	4.69234E+06
20	1.00000E+00	1.53766E-27	1.79257E-27	2.64616E-01	-2.64616E-01	4.68978E+16
20	1.00000E+00	9.39854E-32	1.21469E-31	2.64616E-01	-2.64616E-01	8.26702E+06
c2o16: Old Approach Time=NA, New Approach Time=686.30						
2	1.00000E-01	4.31408E-32	3.50760E-32	8.53711E-01	-8.53711E-01	8.95892E+28
20	1.00000E+00	1.17097E-31	1.35263E-31	2.64616E-01	-2.64616E-01	1.74818E+29
c2o19: Old Approach Time=NA, New Approach Time=1439.74						
2	1.00000E-01	3.69779E-32	4.17927E-32	8.53711E-01	-8.53711E-01	8.80805E+51
20	1.00000E+00	1.21719E-31	1.51134E-31	2.64616E-01	-2.64616E-01	1.46755E+52
c2o22: Old Approach Time=NA, New Approach Time=2195.83						
2	1.00000E-01	5.54668E-32	4.66316E-32	8.53711E-01	-8.53711E-01	2.19468E+76
20	1.00000E+00	1.21719E-31	1.30786E-31	2.64616E-01	-2.64616E-01	4.81574E+76
c2o25: Old Approach Time=NA, New Approach Time= 2605.29						
2	1.00000E-01	4.93038E-32	4.33575E-32	8.53711E-01	-8.53711E-01	1.21860+101
20	1.00000E+00	1.37126E-31	1.55859E-31	2.64616E-01	-2.64616E-01	5.71654+101
c2o28: Old Approach Time=NA, New Approach Time= 3653.88						
2	1.00000E-01	5.23853E-32	5.09168E-32	8.53711E-01	-8.53711E-01	2.81365+126
20	1.00000E+00	7.85779E-32	1.28541E-31	2.64616E-01	-2.64616E-01	1.62764+127
c2o31: Old Approach Time=NA, New Approach Time= 4826.52						
2	1.00000E-01	6.77927E-32	5.62612E-32	8.53711E-01	-8.53711E-01	3.49609+152
20	1.00000E+00	7.43198E-27	4.69403E-27	2.64616E-01	-2.64616E-01	3.60786+168
c2o34: Old Approach Time=NA, New Approach Time= 42884.60						
2	1.00000E-01	6.77927E-32	6.47594E-32	8.53711E-01	-8.53711E-01	1.62494+179
10	5.00000E-01	1.25506E-16	5.60704E-17	6.01971E-01	-6.01971E-01	1.19710+216
12	6.00000E-01	Infinity	NaN	1.08423E-01	-Infinity	NaN

Table 3: Cost and Round-off Error Comparison of New and Old 2D Interpolation Methods With 8 Grid Points Per Wavelength: Quadruple Precision Computations

7 Conclusions and Future Research

It is now possible to develop 200^{th} order or higher wave propagation algorithms whose practical utility is limited only by the precision of the computer. This is made possible by a change in the procedure used to implement the spatial interpolation step of the MESA schemes. With this new approach, the MESA schemes are far simpler to program, more efficient, and incur less round-off error. Automation is no longer necessary for algorithm development, but is still necessary for testing the MESA schemes and for applying multidimensional wall boundary conditions [4].

This new implementation of the MESA schemes makes it possible to adapt the method to the mesh instead of the common approach of adapting the mesh to the method which adversely effects the CFL constraints and complicates grid generation. By adapting the method using estimates of the gradients from the divided difference tableau, an efficient procedure for resolving many different wavelength scales may be possible. In particular, problems which contain a wide range of wavelength scales, such as fan and jet noise generation, may benefit from adaptive algorithms such as these.

In short, this new approach is an improvement to the implementation of the MESA schemes in every regard for methods of 15^{th} order or higher, and has many advantages for the lower order methods as well. Two-point Hermitian stencils have many desirable properties [4], [7] such as:

- They simplify boundary treatments;
- They are more computationally efficient;
- And, they provide higher resolution.

In addition, they may now enable adaptive algorithm implementations and be implemented in arbitrarily high accuracy using the key result from this paper, equation 10. However, a limit to the effective accuracy is encountered with this implementation due to the round-off errors incurred from subtractive cancellation as discussed in the appendix A. Some different approaches for reducing round-off error include:

- Reformulate the time advance step of the MESA schemes to eliminate the factorial term correction discussed in section 6 which multiplies the round-off errors.
- Modify the divided differences to central differences to avoid the divisions by grid point distance, h, in each column of the tableau which also magnifies round-off error [11]
- Reformulate the MESA schemes to propagate complex variables in time and calculate derivatives without subtractive cancellation [15]
- Use divided difference pipelining [1].
- Use the Stirling or Bessel form of the central differences to minimize the coefficients in Newton's interpolatory equation 4.
- Predict roundoff error using the relationship between columns of the tableau (sum of column j = difference of first and last terms in column j-1) [11], and then set to zero all higher order divided differences in the tableau.

The last item limits the accuracy of the MESA scheme, but prevents contamination from round-off error. This also improves efficiency since it prevents unnecessary calculations which will only introduce error and computational overhead. The ability to detect round-off error is a unique advantage of this new approach that may make using very high order methods on available technology practical.

It is necessary to use Hermitian stencils for very high order algorithms since traditional Lagrangian stencils become excessively large in the limit. This paper has provided one approach for this. In addition, it has provided some of the tools necessary for exploring the capabilities of adaptive algorithms for adaptive resolution of steep gradients (shocks).

In the future, as very high precision, large-scale parallel systems become common, Hermitian divided difference implementations of the MESA schemes will be very useful since they offer minimal interprocessor communications [4] and accuracy limited only by machine precision. For practical applications, this places a greater need on developing very high order wall boundary and radiation conditions.

A Sources of Round-off Error

The benefits of very high order accuracy begin to diminish as round-off error grows. The predominant source of this round-off error is subtractive cancellation error which occurs when two numbers close in value are subtracted from one another. This results in a loss of most of the significant digits in the mantissa.

The subtractive cancellation errors will occur in the construction of the divided difference tableau as the standard deviation of the values in each column decreases. Each column, numbered j, in figure 3 is constructed for all i using:

$$Q_{i,j} = \frac{Q_{i,j-1} - Q_{i-1,j-1}}{x_i - x_{i-j}} \tag{34}$$

where for a two point Hermitian spatial interpolation, this relation reduces to dividing by $x_i - x_{i-j} = h$, which is the distance between both grid points in figure 1.

Each divided difference term can be proved using the generalized Rolle theorem to be equivalent to [2]:

$$Q_{i,j} = \frac{f^j(\xi_{i,j})}{j!}, \quad \frac{-h}{2} \leq \xi_{i,j} \leq \frac{h}{2} \tag{35}$$

That is, each divided difference j-column contains data representing the j^{th} derivative at some point on the interval between both points. If we knew it applied at the center (x=0) of the stencil, we could use it directly in the MESA scheme. However, the exact location of the function is not known.

From equations 34 and 35, the following is actually being calculated in each column of the tableau:

$$f^j(\xi_{i,j}) = \frac{f^{j-1}(\xi_{i,j-1}) - f^{j-1}(\xi_{i-1,j-1})}{h} \tag{36}$$

where $\xi_{i,j-1}$ and $\xi_{i-1,j-1}$ are contained within the interval between both grid points, ie. $x_0 \leq \xi_{i,j-1}, \xi_{i-1,j-1} \leq x_1$.

Now let j=1 and apply the mean value theorem of calculus to this equation, then it must be the case that:

$$-\frac{h}{2} = \xi_{i-1,0} < \xi_{i,1} < \xi_{i,0} = \frac{h}{2} \tag{37}$$

Next, let j=2, then "loosely" applying the mean value theorem again to equation 36 results in the relation:

$$-\frac{h}{2} = \xi_{i-1,0} < \xi_{i-1,1} < \xi_{i,2} < \xi_{i,1} < \xi_{i,0} = \frac{h}{2} \tag{38}$$

This term "loosely" is used since the denominator in equation 36 does not change as additional columns are processed and the mean value theorem could only be directly applied if its size was $\xi_{i,1} - \xi_{i-1,1}$. However, on average, the interval will decrease for most functions since there will be one or more critical points ($\xi_{i,j}$) between $\xi_{i,j-1}$ and $\xi_{i-1,j-1}$ when simulating the oscillating functions that occur in acoustical applications.

Therefore as j is increased, the interval in which $\xi_{i,j}$ can occur decreases. This has the effect of compressing all the data in column j closer together. In fact, by definition, the last column of the tableau contains a single element representing the center of the stencil with an interval length of zero for the column. As the points, $\xi_{i-1,j}$ and $\xi_{i,j}$ become closer in equation 36, the subtractive cancellation errors increase.

For example, suppose there are 10 grid points per unit interval, then the first column has an interval length of $h = 1/10$. In addition, suppose the interval length is halved in each column, then column j will have length $\left(\frac{1}{2}\right)^{j-1}$. With a 21^{st} order MESA method (c2o10) there will be 21 columns with the intervals decreasing by six orders of magnitude in the last three columns.

In fact, by the 5^{th} column there is very little separation between both data points (.00625 units) resulting in subtracting two numbers that are very close in value.

This subtractive cancellation error is then magnified by the division of the small number, h, in equation 36. It is a simple matter to eliminate this division by using the forward, backward, or centered-difference forms [11] of the Newton interpolant. However, even without this improvement, the results of this paper show using Hermitian divided differences with two-point Hermitian MESA schemes results in less subtractive cancellation than previous methods.

B Fortran90 Implementation

```
!-------------------------------------------------------------------------
! This routine will calculate the spatial derivatives for the MESA scheme
! Using the data fdata(dx,dy,x,y) = D[f[x,y],{x,dx},{y,dy}]
! It applies only to a c2os scheme, where the s=0,1,2, ...
!      0------|------0 y=y1
!      |             |
!      |             |
!      -      +      - y=0
!      |             |
!      |             |
!      0------|------0 y=y0
!  x=x0      x=0     x=x1
!
! INPUT: fdata(dx,dy,x,y) , x=x0,x1, y=y0,y1 , dx=0,s , dy=0,s
! OUTPUT:cf(dx,dy,0) , dx=0,1, ..., 2*s + 1 , dy=0,1, ..., 2*s+1
! -----------------------------------------------------------------------

program multidimensional
implicit none

!----------------------------
! Define variables to be used
!----------------------------
integer :: digits, e,i,j,k,s,n, dx, dy,r, yp, AllocateStatus
real (kind=selected_real_kind(30)) :: z, deltah, prod1, prod2, sum, innersum
real (kind=selected_real_kind(30)) :: h, x0,x1
real (kind=selected_real_kind(30)), DIMENSION(:,:,:,:), ALLOCATABLE:: fdata
real (kind=selected_real_kind(30)), DIMENSION(:,:), ALLOCATABLE:: coef,Q
real (kind=selected_real_kind(30)), DIMENSION(:,:,:,:), ALLOCATABLE:: cf,cfdata
integer, parameter :: Prec30 = Selected_Real_Kind(30)


!-------------------------------------------------------------------------
! Select the c2os MESA scheme desired.  Order of scheme will be 2 * s + 1
!-------------------------------------------------------------------------
print *,"Enter the degree, c2os"
read *, s

!--------------------------------------------------------
! Dynamically allocate memory space for array
!--------------------------------------------------------
ALLOCATE (coef(0:2*(s+1),0:2*(s+1)), STAT = AllocateStatus)
If (AllocateStatus /= 0) STOP '*** Not Enough Memory ***'
ALLOCATE (cf(0:2*(s+1),0:2*(s+1),0:0,0:0), STAT = AllocateStatus)
If (AllocateStatus /= 0) STOP '*** Not Enough Memory ***'
ALLOCATE (cfdata(0:2*(s+1),0:2*(s+1),0:1,0:1), STAT = AllocateStatus)
If (AllocateStatus /= 0) STOP '*** Not Enough Memory ***'
ALLOCATE (Q(0:2*(s+1),0:2*(s+1)), STAT = AllocateStatus)
```

```
If (AllocateStatus /= 0) STOP '*** Not Enough Memory ***'
ALLOCATE (fdata(0:2*(s+1),0:2*(s+1),0:1,0:1), STAT = AllocateStatus)
If (AllocateStatus /= 0) STOP '*** Not Enough Memory ***'


!-----------------------------------------------------------------------
! Spacing between adjacent grid points, assume same in x and y direction
!-----------------------------------------------------------------------
h=dble(1._Prec30/4._Prec30)

!---------------------------------------------
! Location of right grid point as shown at top
!---------------------------------------------
x1=h/2._Prec30

!--------------------------------------------
! Location of left grid point as shown at top
!--------------------------------------------
x0=-h/2._Prec30

!-------------------------------------------------------
! Spacing between grid points x0, x1 , usually same as h
!-------------------------------------------------------
deltah = x1-x0


!=============================================================================!
! THIS SECTION NEEDS COMPUTED ONLY ONCE AND ITS RESULTS CAN BE REUSED AT EACH !
! STENCIL -- SIGNIFICANT COMPUTATIONAL SAVINGS                                !
! The variable array, coef( , ) is assigned next and does not depend on the   !
! stencil data, only upon the MESA scheme employed                            !
!=============================================================================!
outerloop1: do dx=0, 2 * s + 1

!--------------------------------------------------
! Compute the first set of coefficients, coef(i,dx)
!--------------------------------------------------
  do i=dx, s
    coef(i,dx) = ( Fac(i)/Fac(i-dx) )* (-x0)**(i-dx)
  end do

!--------------------------------------------------
! Compute the other set of coefficients, coef(i,dx)
!--------------------------------------------------
csetloop:  do i=s+1, 2 * s + 1

    sum = 0.0
sumloop: do r=0, dx

        prod1=1.0
        do e=0, dx-1-r
```

```
          prod1 = prod1 * (i - s - 1  - e)
        end do

        prod2=1.0
        do k=0, r-1
          prod2 = prod2 * (s + 1 - k)
        end do

    sum = sum + ( Fac(dx)/(Fac(dx-r)*Fac(r)) ) * (-x0)**(s+1-r) * &
                 ( -x1)**(i-s-1-dx+r) * prod1 * prod2
  end do sumloop
    coef(i,dx)=sum
  end do csetloop
  end do outerloop1

!=============================================================================!
! This section is repeated for each stencil in the domain                     !
!=============================================================================!

!-------------------------------------------------------------------------
! Assign known data and its derivatives from the MESA c2os 2 by 2 stencil
!-------------------------------------------------------------------------
do dx=0, s
do dy=0, s
 fdata(dx,dy,0,0)= STENCIL DATA AT LOWER LEFT
 fdata(dx,dy,0,1)= STENCIL DATA AT UPPER LEFT
 fdata(dx,dy,1,0)= STENCIL DATA AT LOWER RIGHT
 fdata(dx,dy,1,1)= STENCIL DATA AT UPPER RIGHT
end do
end do

!-----------------------------------------------------------------
! Calculate the tableau by first inserting the known data into it
!-----------------------------------------------------------------
lastloop: do yp=0, 1
do dy=0,s
do i=0, s
!------------------------------
! Load table in for x-direction
!------------------------------
 do dx=0, i
    Q(i,dx)= fdata(dx,dy,0,yp) / Fac(dx)
    Q(i+s+1,dx) = fdata(dx,dy,1,yp) / Fac(dx)
 end do
end do

!-----------------------------------------------------------------------------
! Next perform algorithm 3.2 in Burden to construct Divided Difference Tableau
! ----------------------------------------------------------------------------
n=(2 * (s+1))
do i=s+1, n-1
```

```
  do j=i-s, i
     Q(i,j) = (Q(i,j-1) - Q(i-1, j-1))/ (deltah)
  end do
end do


!-----------------------------------------------------------
! Evaluate the spatial derivatives at center using short form
!-----------------------------------------------------------
do dx=0, 2 * s + 1

sum=0.0
do i=dx, 2 * s + 1
sum = sum + Q(i,i) * coef(i,dx)
end do
cfdata(dx,dy,0,yp)=sum
end do
end do
end do lastloop

!----------------------------------------------------------------------
! Repeat this process for the y-direction, only using the cfdata( , , ,)
! as developed by Goodrich
!----------------------------------------------------------------------

!---------------------------------
! Compute Y-Direction interpolation
!---------------------------------
! Load table for Y interpolation
! --------------------------------
dxloop: do dx=0, (2*(s+1)-1)
do i=0, s
 do dy=0, i
    Q(i,dy)= cfdata(dx,dy,0,0) / Fac(dy)
    Q(i+s+1,dy) = cfdata(dx,dy,0,1) / Fac(dy)
 end do
end do

!----------------------------------------------------------------------------
! Next perform algorithm 3.2 in Burden to construct Divided Difference Tableau
! ---------------------------------------------------------------------------
n=(2 * (s+1))
do i=s+1, n-1
  do j=i-s, i
     Q(i,j) = (Q(i,j-1) - Q(i-1, j-1))/ (deltah)
  end do
end do

!-----------------------------------------------------------
! Evaluate the spatial derivatives at center using short form
!-----------------------------------------------------------
```

```
do dy=0, 2 * s + 1

sum=0.0
do i=dy, 2 * s + 1
sum = sum + Q(i,i) * coef(i,dy)
end do
cf(dx,dy,0,0)=sum/(fac(dx)*fac(dy))
end do

end do dxloop

!----------------------------------------------------------------------
! The spatial derivatives for the c2os MESA scheme at the center of the
! 2 by 2 stencil are stored in the cf(dx,dy,0,0) variables
! May need to use cf(dx,dy,0,0)/(dx! dy!) depending upon how the exact
! local propagator is evaluated
!----------------------------------------------------------------------
do dx=0,2*s+1
do dy=0,2*s+1
  print *, "cf(",dx,",",dy,",0,0)=",cf(dx,dy,0,0)
end do
end do

!------------------------------------------------------
!This process is repeated for the p, u, and v variables
!------------------------------------------------------

CONTAINS
!-----------------------------------------------------------------------
! Factorial Function
! In more efficient implementations, can precalculate all the factorial
! results and store as an array
!-----------------------------------------------------------------------
FUNCTION Fac(N2)

  REAL (kind=selected_real_kind(30)) :: Fac
  INTEGER, INTENT(IN) :: N2
  INTEGER :: I2

  Fac = 1.0
  DO I2 = 2, N2
     Fac = Fac * I2
  END DO
END FUNCTION Fac
end
```

References

[1] Al-Ayyoub, A.E. "Pipelined Algorithm for Newton's Divided Difference Interpolation", Computers & Structures, Vol. 58, No.4, pp. 689-701, 1996.

[2] Burden, R.L.; Faires, J.D. "Numerical Analysis", PWS-KENT, Boston, 1989.

[3] Cox,D.;Little,J.;O'Shea,D. "Ideals, Varieties, and Algorithms," Springer-Verlag, 1992.

[4] Dyson, R.W. "An Automated Code Generator For Three-Dimensional Acoustic Wave Propagation With Geometrically Complex Solid Wall Boundaries", Ph.D. Dissertation, Case Western Reserve University, May 1999.

[5] Goodrich,J.W. "High Accuracy Finite Difference Algorithms for Computational Aeroacoustics," AIAA 97-1584, 1997.

[6] Goodrich,J.W.;Dyson,R.W. "Automated Development of Accurate Algorithms and Efficient Codes for Computational Aeroacoustics," Computational Aerosiences Conference, NASA Ames, 1998.

[7] Dyson, R.W.;Goodrich,J.W. "An Automated Approach to Very High Order Aeroacoustic Computations in Complex Geometries", 6^{th} AIAA Aeroacoustics Conference, June 2000.

[8] Goodrich, J.W. "A Comparison of Numerical Methods for Computational Aeroacoustics", AIAA 99-1943, 1999.

[9] Goodrich, J.W. "High Order Implementations of Accurate Boundary Conditions", AIAA 99-1942, 1999.

[10] Hamming, R.W. "Numerical Methods for Scientists and Engineers", McGraw-Hill, 1973.

[11] Hildebrand, F.B. "Introduction to Numerical Analysis", McGraw-Hill, 1974.

[12] Micchelli, C.A. "A Constructive Approach to Kergin Interpolation in R^k", Technical Summary Report #1895, Army Contract# DAAG29-75-C-0024, 1978.

[13] Powell, M.J.D. "Approximation theory and methods", Cambridge University Press, 1981

[14] Schulz, M.H. (1973), Spline Analysis, Prentice-Hall, Englewood Cliffs, N.J.; 156 pp. QA221.S33

[15] Squire, W., Trapp, G. "Using Complex Variables to Estimate Derivatives of Real Functions", SIAM Review, Vol. 40, No. 1, pp. 110-112, March 1998.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (*Leave blank*)

2. REPORT DATE
April 2000

3. REPORT TYPE AND DATES COVERED
Technical Memorandum

4. TITLE AND SUBTITLE
A Very High Order, Adaptable MESA Implementation for Aeroacoustic Computations

5. FUNDING NUMBERS
WU–522–81–11–00

6. AUTHOR(S)
Rodger W. Dyson and John W. Goodrich

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
John H. Glenn Research Center at Lewis Field
Cleveland, Ohio 44135–3191

8. PERFORMING ORGANIZATION REPORT NUMBER
E–12192

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
Washington, DC 20546–0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
NASA TM—2000-209944

11. SUPPLEMENTARY NOTES

Responsible person, Rodger W. Dyson, organization code 5940, (216) 433–9083.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified - Unlimited
Subject Categories: 71, 64 and 61 Distribution: Nonstandard

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

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 words)***

Since computational efficiency and wave resolution scale with accuracy, the ideal would be infinitely high accuracy for problems with widely varying wavelength scales. Currently, many of the computational aeroacoustics methods are limited to 4th order accurate Runge-Kutta methods in time which limits their resolution and efficiency. However, a new procedure for implementing the Modified Expansion Solution Approximation (MESA) schemes, based upon Hermitian divided differences, is presented which extends the effective accuracy of the MESA schemes to 57th order in space and time when using 128 bit floating point precision. This new approach has the advantages of reducing round-off error, being easy to program, and is more computationally efficient when compared to previous approaches. Its accuracy is limited only by the floating point hardware. The advantages of this new approach are demonstrated by solving the linearized Euler equations in an open bi-periodic domain. A 500th order MESA scheme can now be created in seconds, making these schemes ideally suited for the next generation of high performance 256-bit (double quadruple) or higher precision computers. This ease of creation makes it possible to adapt the algorithm to the mesh in time instead of its converse; this is ideal for resolving varying wavelength scales which occur in noise generation simulations. And finally, the sources of round-off error which effect the very high order methods are examined and remedies provided that effectively increase the accuracy of the MESA schemes while using current computer technology.

14. SUBJECT TERMS
Jet noise; Cartesian; Grid; Generation; Hermitian; Divided difference

15. NUMBER OF PAGES
30

16. PRICE CODE
A03

17. SECURITY CLASSIFICATION OF REPORT
Unclassified

18. SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

19. SECURITY CLASSIFICATION OF ABSTRACT
Unclassified

20. LIMITATION OF ABSTRACT