High-Order Implicit-Explicit Multi-Block Time-stepping Method for Hyperbolic PDEs

Tanner B. Nielsen*
Jacobs ESSSA Group, NASA Marshall Space Flight Center, Huntsville, AL, 35812, USA

Mark H. Carpenter†
NASA Langley Research Center, Hampton, VA, 23681, USA

Travis C. Fisher‡
Sandia National Laboratories, Albuquerque, NM, 87123, USA

Steven H. Frankel§
Purdue Universiy, West Lafayette, IN, 47907, USA

This work seeks to explore and improve the current time-stepping schemes used in computational fluid dynamics (CFD) in order to reduce overall computational time. A high-order scheme has been developed using a combination of implicit and explicit (IMEX) time-stepping Runge-Kutta (RK) schemes which increases numerical stability with respect to the time step size, resulting in decreased computational time. The IMEX scheme alone does not yield the desired increase in numerical stability, but when used in conjunction with an overlapping partitioned (multi-block) domain significant increase in stability is observed. To show this, the Overlapping-Partition IMEX (OP IMEX) scheme is applied to both one-dimensional (1D) and two-dimensional (2D) problems, the nonlinear viscous Burger's equation and 2D advection equation, respectively. The method uses two different summation by parts (SBP) derivative approximations, second-order and fourth-order accurate. The Dirichlet boundary conditions are imposed using the Simultaneous Approximation Term (SAT) penalty method. The 6-stage additive Runge-Kutta IMEX time integration schemes are fourth-order accurate in time.

An increase in numerical stability 65 times greater than the fully explicit scheme is demonstrated to be achievable with the OP IMEX method applied to 1D Burger's equation. Results from the 2D, purely convective, advection equation show stability increases on the order of 10 times the explicit scheme using the OP IMEX method. Also, the domain partitioning method in this work shows potential for breaking the computational domain into manageable sizes such that implicit solutions for full three-dimensional CFD simulations can be computed using direct solving methods rather than the standard iterative methods currently used.

Nomenclature

ARK	Additive Runge-Kutta
CFD	Computational Fluid Dynamics
CFL	Courant-Friedrichs-Lewy condition
CDR	Convection-Diffusion-Reaction
DFL	Diffusive condition

*Computational Fluid Dynamicist, Fluid Dynamics Branch-ER42, MSFC, Huntsville, AL, 35812, AIAA Member
†Senior Research Scientist, Computational Aerosciences Branch, Hampton, VA, 23681, AIAA Member
‡Postdoctoral Researcher, Department?, 1515 Eubank SE, Albuquerque, NM, 87123, AIAA Member
§Professor, School of Mechanical Engineering, Hovde Hall of Administration, West Lafayette, IN 47907, AIAA Member

DGFEM	Discontinuous Galerkin Finite Element Method
ERK	Explicit Runge-Kutta Method
ESDIRK	Explicit Singly Diagonally Implicit Runge-Kutta Method
FD	Finite-difference
IMEX	Implicit-Explicit Method
IVP	Initial Value Problem
OP IMEX	Overlapping-Partition Implicit-Explicit Method
ODE	Ordinary Differential Equation
NSE	Navier-Stokes Equations
PDE	Partial Differential Equation
RK	Runge-Kutta Method
RK4	Classic Fourth-order Runge-Kutta Method
SAT	Simultaneous Approximation Term
SBP	Summation by parts
1D	One-dimensional
2D	Two-dimensional
3D	Three-dimensional

I. Introduction and Background

The full three-dimensional (3D) Navier-Stokes equations (NSE), which govern fluid flow, are composed of unsteady, convective, and diffusive terms. These equations are nonlinear, due to the convective term, making them challenging to solve. Time-dependent computational fluid dynamics (CFD) solutions require numerical methods with the ability to efficiently and accurately model each of the mathematical terms, yielding insight into the flow physics which cannot or may be very difficult to gain otherwise.

The general algorithm for numerical integration of time-dependent equations has remained essentially the same for the last 3 or 4 decades.[1] Explicit schemes are the most popular and widely used method due to their easy implementation and computational efficiency. However, explicit schemes are restricted by the Courant-Friedrichs-Lewy (CFL) condition for convective terms and the equivalent diffusive condition (DFL) for diffusive terms, which dictate the time step size based on grid size and flow velocity or viscosity, respectively. When systems become stiff, the CFL and DFL conditions severely limit the time step size and explicit schemes become inefficient. A stiff system, as defined by Chapra and Canale,[2] is a system which involves both rapidly and slowly changing components. An example of a geometry-induced stiff system important in CFD is that of a boundary layer on an airfoil. The cells near the airfoil wall are very small in order to capture the small boundary layer, whereas the cells away from the wall can be orders of magnitude larger. One other example resulting in a stiff system is the reaction terms in convection-diffusion-reaction (CDR) problems. These examples require solutions to stiff equations which are not efficiently solved using explicit schemes. Stiff systems are most efficiently solved using unconditionally stable implicit schemes, which are not dependent upon the CFL and DFL conditions.

One approach to improving the severe time step stability of explicit schemes is to solve individual cells or elements using local time-steps, which is often called multi-rate integration. The local time-stepping allows for a relaxed stability condition in areas that are less stiff, improving the overall efficiency of the method compared to global explicit time-stepping methods. There exist many examples of such schemes, some of which will be discussed next. Osher and Sanders[3] introduced a local time stepping method for one-dimensional conservation laws. Berger and Oliger[4] present an adaptive mesh refinement method that refines the domain in space and time based upon local truncation error. Another adaptive mesh refinement algorithm based on cell size and the stability condition is given by Flaherty et al.[5] Dawson and Kirby present a first- and second-order accurate time discretization scheme based on local CFL condition, rather than a global CFL condition.[6] Tan et al.[7] present a similar approach using a moving mesh. Bernacki et al. use an explicit leap-frog time scheme for use in solving non-dissipative wave propagation problems.[8] While these methods improve the explicit stability condition, most of them are implemented at a lower order accuracy, second-order accurate in time or less. There exist multi-rate integration methods using high-order accuracy, however, they suffer from increased implementation complexity.

Another approach to improve the time step stability limitations of explicit schemes was originally developed to solve the stiff term in CDR equations implicitly while solving the non-stiff terms explicitly as noted by Ascher et al.,[9] referred to as Implicit-Explicit (IMEX) methods. They compare various IMEX methods and find that they are not a universal cure for all problems. However, they show that IMEX schemes can be very effective on certain problems when chosen prudently. Further work from Ascher et al.[10] focuses strictly on IMEX-RK methods for solving CDR problems, concluding that they exhibit undesirable time-step restrictions unless diffusion strongly dominates. Calvo and Novo[11] present two variable step linear IMEX schemes for CDR problems, which is competitive with alternative stiffly accurate time integrators. Fritzen and Wittekindt[12] show a family of partitioned RK schemes for integration of semi-discrete equations. Second- and third-order semi-implicit schemes are used by Zhong[13] for the simulations of hypersonic flows. Similar high-order semi-implicit methods are used in Ref. 14. Some limitations of the methods described above include: errors in the coupling of the IMEX schemes which lead to less robust simulations, poor stability properties, and no error control.

Kennedy and Carpenter[15] develop additive Runge-Kutta (ARK) methods which allow domain-partitioned IMEX time-splitting, meaning that portions of the domain are solved implicitly while other portions are solved explicitly. Regions of the domain are implicitly integrated using an L-stable, stiffly-accurate explicit, singly diagonally implicit Runge-Kutta (ESDIRK) method, while other parts of the domain are integrated using an explicit Runge-Kutta (ERK) method. These are the Implicit-Explicit Runge-Kutta (IMEX-RK) schemes that are used in this work. Kanevsky et al.[1] use these additive IMEX-RK schemes to solve systems with large amounts of geometry-induced stiffness using a discontinuous Galerkin finite element method (DGFEM) discretization. Their work shows that the IMEX-RK method is more efficient than ERK schemes when solving geometry-induced stiff systems. They alleviate the time step restriction by solving the very small elements in the domain using the ESDIRK scheme while the larger elements are solved using the ERK scheme.

The focus of this work is not to separate the solution of stiff and non-stiff terms as was the focus of most the IMEX methods previously described. This work uses the IMEX-RK methods from Kennedy and Carpenter[15] to split up the computational domain into partitions or blocks. The independent block partitions are advanced in time using the implicit scheme, while the terms coupling the blocks together are advanced in time using the explicit scheme. In addition, the partitions are overlapped at the block interfaces, where portions of the finite-difference (FD) matrix are solved twice in separate implicit solve blocks. The added cost of solving portions of the domain twice are justified by the significant increase in numerical stability, as compared to purely explicit schemes, resulting from what will be called the Overlapping-Partition Implicit-Explicit (OP IMEX) method. The novel OP IMEX method presented in this work not only yields an increase numerical stability, but provides a way by which implicit methods used in CFD can potentially be solved using direct solving methods rather than iterative methods.

II. Methodology

The numerical methods, theory, and methodology that make up the OP IMEX method will be described in this section. This will include discussion of the FD operators formulated using the summation by parts (SBP) rule,[16] as well at the Simultaneous Approximation Term (SAT) penalty method of imposing the Dirichlet boundary conditions in the weak sense.[17] A general OP IMEX algorithm will also be presented, describing how the method was implemented in the code used to produce these results. The partitioning and overlapping of the domain is an important aspect of this work and the specific one-dimensional (1D), and two-dimensional (2D) domain partitioning will be presented in detail.

The governing equations of interest in CFD are typically partial differential equations (PDEs), that is, equations that are dependent upon multiple variables and their partial derivatives.[18] In order to numerically solve such equations, one must discretize them such that the original PDE is transformed into a system of algebraic equations. One way to accomplish this is by semi-discretizing the computational domain of interest using FD approximations of the spatial derivatives, resulting in a semi-discrete system of ordinary differential equations (ODEs). The ODE, by definition, is an equation that is a function of a single variable and its derivatives. The semi-discrete scheme is written as an initial value problem (IVP)

$$\frac{\mathrm{d}\mathbf{U}}{\mathrm{d}t} = \mathbf{F}(t, \mathbf{U}(t)), \quad \mathbf{U}(t_0) = \mathbf{U}_0, \tag{1}$$

where $\mathbf{U}$ is the unknown variable in vector form of length N, and N is the number of ODEs resulting from the spatial discretization. The system of ODEs are integrated in time using a high-order Runge-Kutta method (see section C). The way in which the boundary conditions are imposed play a vital role when solving PDEs and will be discussed in section B.

A. Finite Difference: Summation by parts

In this section, the formulation of the FD-based spatial derivative approximations, both first and second derivatives, will be developed. The SBP property allows one to obtain a discrete energy estimate, resulting in desired stability properties.

1. *First Derivative Approximation*

The domain, $\Omega \in [x_L, x_R]$, is discretized using N equidistant points, where $\mathbf{x}$ represents the discrete solution points,

$$\mathbf{x} = (x_1, x_2, ..., x_N)^T, \quad x_i = x_L + (i-1)\mathrm{d}x, \quad i = 1, 2..., N, \quad \mathrm{d}x = \frac{x_R - x_L}{N-1}. \tag{2}$$

The name summation by parts comes from the fact that the derivative operators mimic integration by parts,

$$\int_{x_L}^{x_R} \phi u_x \,\mathrm{d}x = \phi u|_{x_L}^{x_R} - \int_{x_L}^{x_R} \phi_x u \,\mathrm{d}x. \tag{3}$$

The SBP rule is held by constructing the first derivative approximation, $\mathbf{u}_x(\mathbf{x}) = \mathcal{D}\mathbf{u} + \mathcal{T}_{p\,2p\,p}$, where $\mathcal{T}_{p\,2p\,p}$ is the truncation error associated with the order of the scheme, p. The scheme has the following properties:

$$\begin{gathered} \mathcal{D} = \mathcal{P}^{-1}\mathcal{Q}, \quad \mathcal{P} = \mathcal{P}^T, \quad \zeta^T \mathcal{P} \zeta > 0, \quad \zeta \neq \mathbf{0}, \\ \mathcal{Q}^T = \mathcal{B} - \mathcal{Q}, \quad \mathcal{B} = \mathrm{diag}(-1, 0, ..., 0, 1). \end{gathered} \tag{4}$$

The diagonal $\mathcal{P}$ matrix incorporates the grid spacing into the derivative approximation. The $\mathcal{Q}$ matrix is nearly skew-symmetric, all rows sum to zero, and the first and last columns sum to -1 and 1, respectively. One can show the summation by parts property by numerically integrating the following,

$$\int_{x_L}^{x_R} \phi u_x \,\mathrm{d}x \approx \phi^T \mathcal{P} \mathcal{D} \mathbf{u}. \tag{5}$$

An inner product of ϕ and the approximate derivative $\mathcal{D}\mathbf{u}$ is formed, where $\phi = (\phi(x_1), \phi(x_2), ..., \phi(x_N))^T$ and the integration weights are in the $\mathcal{P}$ norm. Using the properties from equation 4 the summation by parts rule is shown as follows,

$$\begin{aligned} \phi^T \mathcal{P} \mathcal{D} \mathbf{u} &= \phi^T \mathcal{P} \mathcal{P}^{-1} \mathcal{Q} \mathbf{u} = \phi^T (\mathcal{B} - \mathcal{Q}^T) \mathbf{u} \\ &= \phi_N u_N - \phi_1 u_1 - \phi^T \mathcal{Q}^T \mathbf{u}. \end{aligned} \tag{6}$$

The final form

$$\phi^T \mathcal{P} \mathcal{D} \mathbf{u} = \underbrace{\phi_N u_N - \phi_1 u_1}_{\phi u|_{x_L}^{x_R}} - \underbrace{\phi^T \mathcal{D}^T \mathcal{P} \mathbf{u}}_{\int_{x_L}^{x_R} \phi_x u \mathrm{d}x}, \tag{7}$$

mimics integration by parts shown above in equation 3.

The truncation error, $\mathcal{T}_{p\,2p\,p}$, of the approximation has $2p$ interior accuracy and p accuracy at the boundary. The specific SBP approximations used in the work are SBP 1-2-1 and SBP 2-4-2, where the 1-2-1 scheme is first-order accurate at the boundary and second-order accurate in the interior, and the 2-4-2 scheme is second-order accurate at the boundary and fourth-order accurate in the interior. Refer to appendix A for the full SBP operators, which come from Mattsson and Nordström.[16]

2. *Second Derivative Approximation*

The viscous terms in the equations require a second derivative approximation as well. Integration by parts leads to

$$\int_{x_L}^{x_R} \phi u_{xx} \mathrm{d}x = \phi u_x|_{x_L}^{x_R} - \int_{x_L}^{x_R} \phi_x u_x \mathrm{d}x. \tag{8}$$

It is possible to use the first derivative approximation to construct the second derivative approximation. However, there are several drawbacks to doing this described in,[16] one of which is the width of the operator would increase from $2p+1$ to $4p+1$, resulting in a less efficient scheme. The properties of the second derivative operator are defined as

$$\mathcal{D}_2 = \mathcal{P}^{-1}(-\mathcal{M} + \mathcal{B}\mathcal{D}), \quad \mathcal{M} = \mathcal{M}^T, \quad \zeta^T \mathcal{M}\zeta \geq 0, \quad \forall \zeta, \tag{9}$$

where $\mathcal{M}$ is restricted to

$$\mathcal{M} = \mathcal{D}^T\mathcal{P}\mathcal{D} + \mathcal{R}, \quad \mathcal{R} = \mathcal{R}, \quad \zeta^T\mathcal{R}\zeta \geq 0, \quad \forall\zeta, \tag{10}$$

in order to show the SBP property (see below). Integrating again using the $\mathcal{P}$ norm as the integration weights yields

$$\phi^T\mathcal{P}\mathcal{D}\mathbf{u} = \phi^T\mathcal{P}\mathcal{P}^{-1}(-\mathcal{M} + \mathcal{B}\mathcal{D})\mathbf{u} = \phi^T\mathcal{B}\mathcal{D}\mathbf{u} - \phi^T\mathcal{M}\mathbf{u}. \tag{11}$$

Using the restricted definition of $\mathcal{M}$ from equation 10, the final form of the equation is

$$\phi^T\mathcal{P}\mathcal{D}\mathbf{u} = \underbrace{\phi_N u_N - \phi_1 u_1}_{\phi u|_{x_L}^{x_R}} - \underbrace{\phi^T\mathcal{D}^T\mathcal{P}\mathcal{D}\mathbf{u}}_{\int_{x_L}^{x_R}\phi_x u_x \mathrm{d}x} - \phi^T\mathcal{R}\mathbf{u}. \tag{12}$$

The $\mathcal{R}$ matrix is known as the remainder matrix which scales with the grid spacing at the same order as the error of the second derivative operator.[19] Therefore, the last term of equation 12 is small and decreases with increasing resolution so the SBP property is preserved. The SBP second derivative operators used in this work can be found in appendix A.

B. Boundary Conditions: SAT Penalty Method

The SAT method, coined by Carpenter et al.,[17] is the boundary condition used in conjunction with SBP operators. They describe why the SBP property, by itself, is not guaranteed to be stable or time stable. The reader is referred to the original paper for an in-depth discussion. In summary, imposing the physical boundary condition directly, $u(x_N) = g(t)$, can destroy the SBP property and an energy estimate becomes unobtainable. Rather, the boundary condition is imposed as an additional 'penalty' term in the differential equation, preserving the SBP property and allowing an energy estimate to be obtained.

The hyperbolic scalar equation

$$\frac{\partial u}{\partial t} = \lambda\frac{\partial u}{\partial x}, \quad x \in [0,1], \quad t \in [0,\infty), \tag{13}$$

will be used to illustrate the formulation and general theory of the method in this section. For $\lambda \geq 0$, the wave is traveling to the left and the boundary condition is $u(1,t) = g(t)$. The energy rate is

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_0^1 u^2(x,t)\mathrm{d}x = \lambda(u^2(1,t) - u^2(0,t)), \tag{14}$$

and is used with the SBP property to show the necessity of the SAT method for imposing the boundary conditions.

The equation is semi-discretized using the SBP first derivative approximation (see section A)

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = \lambda\mathcal{P}^{-1}\mathcal{Q}\mathbf{u}, \tag{15}$$

where $\mathbf{u}$ represents the approximate solution at the discrete locations, $\mathbf{x}$, from equation 2. It is important to note that the SBP properties from equation 4 give the following energy rate,

$$\frac{\mathrm{d}E}{\mathrm{d}t} = q_{0,0}u_0^2 + q_{N,N}u_N^2, \tag{16}$$

where

$$E(t) = \frac{1}{\lambda}(\mathbf{u}(t), \mathcal{H}\mathcal{P}\mathbf{u}(t)). \tag{17}$$

The boundary is imposed as a penalty term as follows,

$$\mathcal{P}\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = \lambda \mathcal{Q}\mathbf{u} - \tau q_{N,N}\mathcal{S}(u_N - q(t)), \tag{18}$$

where $\mathcal{S} = \mathcal{H}^{-1}(0, 0, ...0, 1)^T$. Multiplying by the matrix $\mathcal{H}$

$$\mathcal{H}\mathcal{P}\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = \lambda \mathcal{H}\mathcal{Q}\mathbf{u} - \tau q_{N,N}(0, 0, ...0, 1)^T u_N, \tag{19}$$

which can be written as an energy rate

$$\frac{\mathrm{d}E(t)}{\mathrm{d}t} = q_{0,0}u_0^2 + q_{N,N}u_N^2 - \tau q_{N,N}u_N^2. \tag{20}$$

Immediately it is clear that if $\tau \geq 1$ then scheme is both stable and time stable. It can also be proven that the SAT method maintains the order of accuracy of SBP approximation (refer to the original paper for the proof).

C. IMEX-RK Time Integration

The general PDE is reduced to an ODE by discretizing the equation and approximating the spatial derivatives using the SBP operators discussed in section A. This results in an ODE at each discrete solution point, and therefore, a system of ODEs. Recall that the boundary conditions are included in the differential equation using the SAT method (see section B). The semi-discrete equation is now a system of ODEs and is written as an initial value problem (IVP)

$$\frac{d\mathbf{U}}{dt} = \mathbf{F}(t, \mathbf{U}(t)), \quad \mathbf{U}(t_0) = \mathbf{U}_0(x), \tag{21}$$

where $\mathbf{U}$ is the solution vector of length N, and N is the number of ODEs corresponding to the number of discrete solution points in the domain. The left hand side of the equation, $\frac{d\mathbf{U}}{dt}$, can now be advanced in time by numerical integration, $\mathbf{U}(t + \Delta t) = \mathbf{U}^{(n+1)}$. There are several forms of numerical integration, however, fundamentally they are either explicit, implicit, or some combination of explicit and implicit. This work is focused on a newly developed OP IMEX method, which implements high-order ARK schemes by Kennedy and Carpenter.[15]

RK schemes can be written in a tabular form, known as the Butcher tableau,[20] as seen below.

$$\begin{array}{c|c} c_i & a_{ij} \\ \hline & b_i \end{array}$$

The c_i vector contains the RK intermediate time levels, the a_{ij} matrix contains the weights for the ith RK stage, and the b_i vector contains the integration weights for the final stage.

The RK schemes used in this work are written in a slightly modified form of the Butcher tableau,

$$\begin{array}{c|c} c_i & a_{ij} \\ \hline & b_i \\ \hline & \hat{b}_i \end{array}$$

where $\hat{b}_i$ contains the integration weights for the embedded scheme. The embedded scheme is one order less accurate than the main scheme and is used for the adaptive time step controller. The adaptive time step controller is used to assess the stability of the 2D OP IMEX method and will be discussed in detail in section G.

This method uses the ARK schemes of Kennedy and Carpenter,[15] which include a 6-stage fourth-order accurate ERK scheme used for the explicit time stepping, and a 6-stage fourth-order accurate ESDIRK scheme used for implicit time stepping (see appendix A for the schemes in Butcher tableau form). The two schemes use the same b_i and c_i vectors, such that the intermediate time levels and the final time step weights are the same. This is vital for maintaining accuracy when combining the two different schemes for the IMEX temporal advancement. Note that the explicit intermediate weights, a_{ij}^{ex}, are zero on and above the main diagonal, $a_{ij}^{ex} = 0, j \geqslant i$. The implicit intermediate weights, A_{ij}^{im}, are $\frac{1}{4}$ on the main diagonal, $A_{ij}^{im} = \frac{1}{4}, j = i(i > 1)$, and zeros above the main diagonal, $A_{ij}^{im} = 0, j > i$. The ESDIRK scheme contains explicit contributions to each implicit stage, however, it is important to note that it is an implicit scheme and behaves as such.

Now that the RK schemes used to integrate the system of ODEs in time have been explained, the details of how the implicit and explicit contributions to what will be called the 'residual' will be discussed. The residual is referred to the right hand side of equation 21, $\mathbf{F}(t, \mathbf{U}(t))$, which is dependent upon explicit and/or implicit contributions depending on the location in the computational domain. The intermediate stage values are computed using

$$\begin{aligned}\mathbf{U}^{(i)} = \mathbf{U}^{(n)} + \Delta t \sum_{j=1}^{i-1} a_{ij}^{ex}\, \mathbf{F}^{ex}(t + c_j \Delta t, \mathbf{U}^{(j)}) + \Delta t \sum_{j=1}^{i-1} A_{ij}^{im}\, \mathbf{F}^{im}(t + c_j \Delta t, \mathbf{U}^{(j)}) \\ + \Delta t\, A_{ii}^{im}\, \mathbf{F}^{im}(t + c_i \Delta t, \mathbf{U}^{(i)}),\end{aligned} \tag{22}$$

where $\mathbf{F}^{ex}$ is the residual with contributions from the explicit terms, $\mathbf{F}^{im}$ is the residual with contributions from the implicit terms, $\mathbf{U}^{(i)}$ is the approximate solution at the intermediate time level, $\mathbf{U}^{(n)}$ is the approximate solution at the previous stage, and Δt is the chosen time step. The system of ODEs must be advanced to each stage simultaneously, meaning that all the implicit and explicit residuals are advanced to the first stage before advancing to the second stage and so forth.

The final stage value is computed using all the stored explicit and implicit residuals for each stage,

$$\mathbf{U}^{(n+1)} = \mathbf{U}^{(n)} + \Delta t \sum_{i=1}^{s} b_i\, \mathbf{F}^{ex}(t + c_i \Delta t, \mathbf{U}^{(i)}) + \Delta t \sum_{i=1}^{s} b_i\, \mathbf{F}^{im}(t + c_i \Delta t, \mathbf{U}^{(i)}). \tag{23}$$

D. Nonlinear Implicit Solutions

Newton's method is implemented in the IMEX method for nonlinear implicit solves. One must calculate the residual vector as follows:

$$\begin{aligned}\mathbf{G}(\mathbf{U}^{(i)}) = \mathbf{U}^{(i)} - \mathbf{U}^{(n)} - \Delta t \sum_{j=1}^{i-1} a_{ij}^{ex}\, \mathbf{F}^{ex}(t + c_j \Delta t, \mathbf{U}^{(j)}) \\ - \Delta t \sum_{j=1}^{i-1} A_{ij}^{im}\, \mathbf{F}^{im}(t + c_j \Delta t, \mathbf{U}^{(j)}) - \Delta t\, A_{ii}^{im}\, \mathbf{F}^{im}(t + c_i \Delta t, \mathbf{U}^{(i)});\end{aligned} \tag{24}$$

by taking the derivative of equation 24 with respect to the solution stage value, $\mathbf{U}^{(i)}$, one arrives at the Jacobian matrix,

$$\frac{\partial \mathbf{G}}{\partial \mathbf{U}^{(i)}} = [I] - \Delta t\, A_{ii}^{im} \frac{\partial \mathbf{F}^{im}}{\partial \mathbf{U}^{(i)}}, \tag{25}$$

where $[I]$ is the identity matrix. A system of linear equations is formed with the Jacobian matrix, residual vector, and the change in the solution vector as follows,

$$-\mathbf{G}(\mathbf{U}_{k+1}^{(i)}) = \frac{\partial \mathbf{G}}{\partial \mathbf{U}_k^{(i)}} \delta \mathbf{u}, \tag{26}$$

and $\delta \mathbf{u}$ is obtained by solving the system equations. The change in the solution vector, $\delta \mathbf{u}$, is used to update the velocity vector,

$$\mathbf{U}_{k+1}^{(i)} = \mathbf{U}_k^{(i)} + \delta \mathbf{u}, \tag{27}$$

and Newton's method is iterated until convergence is reached. The number of Newton iterations required to obtain convergence is dependent upon the nonlinearity of the system of equations and the desired convergence tolerance. This solution procedure is repeated for each stage in the RK scheme.

As a side note, this nonlinear system of equations solution procedure can also be applied to linear systems of equations, where Newton's method will converge in one iteration for the linear set of equations. In this work, the nonlinear Burger's equation and linear 2D advection equation were studied using the same nonlinear implicit solution procedure described above.

E. 1D Domain Partitioning and Overlapping

The domain partitioning and overlapping used in the OP IMEX method, applied to a 1D problem, is presented in this section. This will include details of the method using both SBP 1-2-1 and SBP 2-4-2 schemes, respectively. In this work, the computational domain $\mathbf{x} \in [x_L, x_R]$ is split into three evenly spaced partitions: $\mathbf{x}_1 \in [x_L, x_A]$, $\mathbf{x}_2 \in [x_{A+1}, x_B]$, $\mathbf{x}_3 \in [x_{B+1}, x_R]$. The number of solution points is adjustable, however, it must be divisible by three in order to always maintain the three equispaced partitions. The choice of three equispaced partitions is not required by the OP IMEX methodology and was chosen simply for easier code implementation for this initial study.

1. SBP 1-2-1 Scheme

The second-order spatially accurate scheme (SBP 1-2-1) uses a three-point wide stencil, requiring the solution of a tridiagonal matrix for implicit time-stepping. Splitting the FD matrix into partitions or blocks results in portions of the domain that can be independently solved, which is desirable when running large simulations on multiple processors. The splitting and overlapping of the FD matrix is shown with the following simple example. The FD matrix results from a computational domain with eight solution points, $\mathbf{a} = (a_1, a_2, ..., a_8)^T$, which are along the main diagonal,

$$\begin{pmatrix} a_1 & c_1 & & & & & & \\ b_2 & a_2 & c_2 & & & & & \\ & b_3 & a_3 & c_3 & & & & \\ & & b_4 & a_4 & c_4 & & & \\ & & & b_5 & a_5 & c_5 & & \\ & & & & b_6 & a_6 & c_6 & \\ & & & & & b_7 & a_7 & c_7 \\ & & & & & & b_8 & a_8 \end{pmatrix}. \tag{28}$$

The domain is partitioned at a single location, between a_4 and a_5 as shown. The upper left and lower right blocks are implicitly solved (shown inside red blocks) while c_4 and b_5 (outside red blocks) are explicitly solved. The simple partitioning of the FD matrix above, along with the implicit and explicit solves, does not gain much in stability by itself. The real gain in stability comes from overlapping the partitions and solving small portions of the domain twice. The system of equations, shown in equation 28, will be used to show the structure of the FD matrix as the partitions are overlapped. The computational domain $\mathbf{a} = (a_1, a_2, ..., a_8)^T$ is again partitioned between solution points a_4 and a_5. However, this time the partition is overlapped once as follows,

$$\begin{pmatrix} a_1 & c_1 & & & & & & & & \\ b_2 & a_2 & c_2 & & & & & & & \\ & b_3 & a_3 & c_3 & & & & & & \\ & & b_4 & a_4 & c_4 & & & & & \\ & & & b_5 & a_5 & & & c_5 & & \\ & & b_4 & & & a_4 & c_4 & & & \\ & & & & & b_5 & a_5 & c_5 & & \\ & & & & & & b_6 & a_6 & c_6 & \\ & & & & & & & b_7 & a_7 & c_7 \\ & & & & & & & & b_8 & a_8 \end{pmatrix}. \tag{29}$$

The system of equations is now $10 \text{x} 10$ due to the single overlap of each partition, rather than $8 \text{x} 8$. The solution points a_4 and a_5 are solved in the upper left block and also in the lower right block. The explicit contributions are shifted to c_5 and b_4 which are also included in the implicit block solves as seen above. This shifted matrix structure, resulting from the overlapped partitions, is mathematically demonstrated using a bordering technique and matrix operations or visually by reordering the domain and applying FD discretization. The partitions can be overlapped multiple times; the explicit contributions continue to shift outward. In this final example of the tridiagonal FD matrix with eight solutions points, the matrix is partitioned again between a_4 and a_5 and overlapped twice,

$$\begin{pmatrix}
a_1 & c_1 & & & & & & & & & & \\
b_2 & a_2 & c_2 & & & & & & & & & \\
 & b_3 & a_3 & c_3 & & & & & & & & \\
 & & b_4 & a_4 & c_4 & & & & & & & \\
 & & & b_5 & a_5 & c_5 & & & & & & \\
 & & & & b_6 & a_6 & & & & & c_6 & \\
 & b_3 & & & & & a_3 & c_3 & & & & \\
 & & & & & & b_4 & a_4 & c_4 & & & \\
 & & & & & & & b_5 & a_5 & c_5 & & \\
 & & & & & & & & b_6 & a_6 & c_6 & \\
 & & & & & & & & & b_7 & a_7 & c_7 \\
 & & & & & & & & & & b_8 & a_8
\end{pmatrix}, \tag{30}$$

resulting in a 12 x 12 system of equations. The solution points a_3, a_4, a_5, and a_6 are implicitly solved (and their neighboring contributions) in both the upper left and lower right blocks (shown in red boxes). The explicit contributions have moved out and are now c_6 and b_3. Overlapping the partitions increases the computational expense of the solution by solving larger set of equations, however, if the gain in stability outweighs the added cost then the method is more efficient. The simple example used above would be very inefficient because a majority of the solution points are double solved with just two overlaps, nearly doubling the size of the original system of equations. However, when solving large systems of equations the overlapped duplicate solves are only a small fraction of the total number of solution points and the method has the potential of being very efficient.

2. SBP 2-4-2 Scheme

The fourth-order spatially accurate scheme (SBP 2-4-2) has a five-point wide stencil and results in a penta-diagonal FD matrix. The domain partitioning and overlapping is performed in a similar manner as with the SBP 1-2-1 scheme. However, the added diagonals in the matrix change the explicit contributions. To show this, a FD matrix with ten solution points, $\mathbf{a} = (a_1, a_2, ..., a_{10})^T$, is shown

$$\begin{pmatrix}
a_1 & c_1 & e_1 & & & & & & & \\
b_2 & a_2 & c_2 & e_2 & & & & & & \\
d_3 & b_3 & a_3 & c_3 & e_3 & & & & & \\
 & d_4 & b_4 & a_4 & c_4 & e_4 & & & & \\
 & & d_5 & b_5 & a_5 & c_5 & e_5 & & & \\
 & & & d_6 & b_6 & a_6 & c_6 & e_6 & & \\
 & & & & d_7 & b_7 & a_7 & c_7 & e_7 & \\
 & & & & & d_8 & b_8 & a_8 & c_8 & e_8 \\
 & & & & & & d_9 & b_9 & a_9 & c_9 \\
 & & & & & & & d_{10} & b_{10} & a_{10}
\end{pmatrix}. \tag{31}$$

The domain is split between solution points a_5 and a_6 and the implicit solves are shown with the red boxes. The wider stencil results in more ghost points that are explicitly solved: e_4, c_5, and e_5 for the upper left block, and d_6, b_6, and d_7 for the lower right block. A similar behavior in the shifted matrix, as compared to the narrow SBP 1-2-1 scheme, is seen when overlapping the partitions. A single overlap results in this matrix structure

$$\begin{pmatrix}
a_1 & c_1 & e_1 & & & & & & & & & \\
b_2 & a_2 & c_2 & e_2 & & & & & & & & \\
d_3 & b_3 & a_3 & c_3 & e_3 & & & & & & & \\
 & d_4 & b_4 & a_4 & c_4 & e_4 & & & & & & \\
 & & d_5 & b_5 & a_5 & c_5 & & & e_5 & & & \\
 & & & d_6 & b_6 & a_6 & & & c_6 & e_6 & & \\
 & & d_5 & b_5 & & & a_5 & c_5 & e_5 & & & \\
 & & & d_6 & & & b_6 & a_6 & c_6 & e_6 & & \\
 & & & & & & d_7 & b_7 & a_7 & c_7 & e_7 & \\
 & & & & & & & d_8 & b_8 & a_8 & c_8 & e_8 \\
 & & & & & & & & d_9 & b_9 & a_9 & c_9 \\
 & & & & & & & & & d_{10} & b_{10} & a_{10}
\end{pmatrix}. \tag{32}$$

Again, the implicit solves are shown inside the red boxes and the explicit contributions (e_5, c_6, e_6 for the upper left block and d_5, b_5, d_6 for the lower right block) have shifted outward. Note that with a single overlap, duplicate explicit contributions are found inside the implicit solve blocks for both the tridiagonal and pentadiagonal matrices. A final example of a pentadiagonal matrix with two overlaps,

$$\begin{pmatrix}
a_1 & c_1 & e_1 & & & & & & & & & & & \\
b_2 & a_2 & c_2 & e_2 & & & & & & & & & & \\
d_3 & b_3 & a_3 & c_3 & e_3 & & & & & & & & & \\
 & d_4 & b_4 & a_4 & c_4 & e_4 & & & & & & & & \\
 & & d_5 & b_5 & a_5 & c_5 & e_5 & & & & & & & \\
 & & & d_6 & b_6 & a_6 & c_6 & & & & & e_6 & & \\
 & & & & d_7 & b_7 & a_7 & & & & & c_7 & e_7 & \\
 & d_4 & b_4 & & & & & a_4 & c_4 & e_4 & & & & \\
 & & d_5 & & & & & b_5 & a_5 & c_5 & e_5 & & & \\
 & & & & & & & d_6 & b_6 & a_6 & c_6 & e_6 & & \\
 & & & & & & & & d_7 & b_7 & a_7 & c_7 & e_7 & \\
 & & & & & & & & & d_8 & b_8 & a_8 & c_8 & e_8 \\
 & & & & & & & & & & d_9 & b_9 & a_9 & c_9 \\
 & & & & & & & & & & & d_{10} & b_{10} & a_{10}
\end{pmatrix}. \tag{33}$$

As expected, the explicit contributions have shifted two more points outward. It is important to remark that the SBP operator matrices do not form perfect tridiagonal and pentadiagonal matrices as shown in the above examples. The upper left and lower right portions of the matrices are modified to handle boundary accuracy (see appendix A). The matrix partitions must be sufficiently away from the boundaries so as to avoid the boundary modified portion of the matrix.

F. 2D Domain Partitioning and Overlapping

The 2D domain partitioning and overlapping, for the OP IMEX method, is presented in this section. A simple example is presented, showing how the domain is partitioned and mapped to the full matrix for the implicit temporal advancement.

The following example demonstrates the domain ordering and mapping of a 2D domain discretized into a 6 x 6 matrix,

$$\begin{pmatrix} 25 & 26 & 27 & 34 & 35 & 36 \\ 22 & 23 & 24 & 31 & 32 & 33 \\ 19 & 20 & 21 & 28 & 29 & 30 \\ 07 & 08 & 09 & 16 & 17 & 18 \\ 04 & 05 & 06 & 13 & 14 & 15 \\ 01 & 02 & 03 & 10 & 11 & 12 \end{pmatrix}. \tag{34}$$

The domain is split into four blocks, each containing a 3 x 3 portion of the domain. The block ordering goes from left-to-right, bottom-to-top, as does the individual block ordering of the solution points. In the example above (equation 34): block 1 is outlined in blue and contains grid points 1-9, block 2 is outlined in yellow and contains grid points 10-18, block 3 is outlined in red and contains grid points 19-27, and block 4 is outlined in green and contains grid points 28-36. The ordering of the domain is such that the grid points in each block are clustered together, which yields smaller implicit block solves once mapped to the full FD matrix (see below).

In order to implicitly solve the 2D problem, a system of equations is formed containing an equation for each grid point in the domain. The example domain (6 x 6) from above is mapped to a full 36 x 36 matrix, which is shown next.

$$\left(\begin{array}{*{36}{c}}
1&2&&4 \\
1&2&3&&5 \\
&2&3&&&6&&&&10 \\
1&&&4&5&&7 \\
&2&&4&5&6&&8 \\
&&3&&5&6&&&9&&&&13 \\
&&&4&&&7&8&&&&&&&&&&&19 \\
&&&&5&&7&8&9&&&&&&&&&&&20 \\
&&&&&6&&8&9&&&&&&&16&&&&&21 \\
&&3&&&&&&&10&11&&13 \\
&&&&&&&&&10&11&12&&14 \\
&&&&&&&&&&11&12&&&15 \\
&&&&&6&&&&10&&&13&14&&16 \\
&&&&&&&&&&11&&13&14&15&&17 \\
&&&&&&&&&&&12&&14&15&&&18 \\
&&&&&&&&9&&&&13&&&16&17&&&&&&&&&&&28 \\
&&&&&&&&&&&&&14&&16&17&18&&&&&&&&&&&29 \\
&&&&&&&&&&&&&&15&&17&18&&&&&&&&&&&&30 \\
&&&&&&7&&&&&&&&&&&&19&20&&22 \\
&&&&&&&8&&&&&&&&&&&19&20&21&&23 \\
&&&&&&&&9&&&&&&&&&&&20&21&&&24&&&&28 \\
&&&&&&&&&&&&&&&&&&19&&&22&23&&25 \\
&&&&&&&&&&&&&&&&&&&20&&22&23&24&&26 \\
&&&&&&&&&&&&&&&&&&&&21&&23&24&&&27&&&&31 \\
&&&&&&&&&&&&&&&&&&&&&22&&&25&26 \\
&&&&&&&&&&&&&&&&&&&&&&23&&25&26&27 \\
&&&&&&&&&&&&&&&&&&&&&&&24&&26&27&&&&&&&34 \\
&&&&&&&&&&&&&&&16&&&&&21&&&&&&&28&29&&31 \\
&&&&&&&&&&&&&&&&17&&&&&&&&&&&28&29&30&&32 \\
&&&&&&&&&&&&&&&&&18&&&&&&&&&&&29&30&&&33 \\
&&&&&&&&&&&&&&&&&&&&&&&24&&&&28&&&31&32&&34 \\
&&&&&&&&&&&&&&&&&&&&&&&&&&&&29&&31&32&33&&35 \\
&&&&&&&&&&&&&&&&&&&&&&&&&&&&&30&&32&33&&&36 \\
&&&&&&&&&&&&&&&&&&&&&&&&&&27&&&&31&&&34&35 \\
&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&32&&34&35&36 \\
&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&33&&35&36
\end{array}\right) \tag{35}$$

The colored boxes in equation 35 correspond to the block number from equation 34 and represent the implicitly solved partitions (block 1 is blue, block 2 is yellow, block 3 is red, and block 4 is green). The terms outside the boxes are the overlapping ghost points which are advanced explicitly. Note that the re-ordering of each block from above yields implicit solve blocks that are tightly clustered together.

G. Time Step Controller

Evaluating the stability of the 2D advection equation was less cut-and-dry as compared to the 1D nonlinear Burger's equation. The stability envelop of the 1D simulation of Burger's equation was analyzed simply by adjusting the time step and running the code. As the time step became too large to maintain numerical stability, the simulation would diverge very quickly due to the nonlinear nature of the equation. The 2D advection equation, on the other hand, did not demonstrate a clear cut stability envelop. As the time step was increased, the error in the solution increased but it did not quickly diverge, making it difficult to quantify the stability of the method. This was most likely due to the linear behavior of the equation. Therefore, an adaptive time step controller was implemented to 'automatically' quantify the stability of the OP IMEX method on this 2D problem.

The explanation from Kanevsky et al.[1] is reproduced here, giving a nice summary on time step controllers. The embedded time-integration scheme, which is one order lower than the main scheme, is used to compute a temporal error. The temporal error is then passed into the adaptive time step controller, such as an I, PI, or PID controller, which automatically and adaptively adjusts the size of the time step.

The I-based controller derivation is shown next, giving insight into the general theory of adaptive time step controllers. The temporal error, δ, is computed by subtracting the solution based on the embedded scheme of order p from the main scheme of order $p+1$

$$\begin{aligned} \delta &= \mathbf{U} - \hat{\mathbf{U}} \\ &= (\mathbf{U}_{\text{exact}} + \mathcal{O}((\Delta t)^{p+1})) - (\mathbf{U}_{\text{exact}} + \mathcal{O}((\Delta t)^{p})) \\ &= \mathcal{O}((\Delta t)^p) \\ &= C(\Delta t)^p, \end{aligned} \tag{36}$$

where C is some constant. The temporal error between two different time steps $\delta^{(n)}$ and $\delta^{(n+1)}$ is compared

$$\begin{aligned} \frac{\delta^{(n+1)}}{\delta^{(n)}} &= \frac{\mathrm{C}((\Delta t)^{(n+1)})^p}{\mathrm{C}((\Delta t)^n)^p} \\ &= \left(\frac{(\Delta t)^{(n+1)}}{(\Delta t)^n}\right)^p, \end{aligned} \tag{37}$$

where $(\Delta t)^{(n+1)}$ is the time step to be computed based on a temporal error of $\delta^{(n+1)}$. The temporal error is then given as a user-defined input, $\epsilon = \delta^{(n+1)}$, and equation 37 is rearranged to solve for the computed time step,

$$(\Delta t)^{(n+1)} = (\Delta t)^n \left(\frac{\epsilon}{\delta^{(n)}}\right)^{\frac{1}{p}}. \tag{38}$$

Lastly, a factor of safety κ is added and the computed temporal error is written as an L_∞ norm. The I-based controller is

$$(\Delta t)_{\mathrm{I}}^{(n+1)} = \kappa(\Delta t)^n \left(\frac{\epsilon}{\|\delta^{(n)}\|_\infty}\right)^{\frac{1}{p}}. \tag{39}$$

The PID-based controller uses the computed temporal error from two previous time steps, in addition to the current time step, and goes as follows,

$$(\Delta t)_{\mathrm{PID}}^{(n+1)} = \kappa(\Delta t)^n \left(\frac{\epsilon}{\|\delta^{(n)}\|_\infty}\right)^{\alpha} \left(\frac{\|\delta^{n-1}\|_\infty}{\epsilon}\right)^{\beta} \left(\frac{\epsilon}{\|\delta^{(n-2)}\|_\infty}\right)^{\lambda}. \tag{40}$$

where ϵ is the specified tolerance for the temporal error, and p is the order of accuracy of the embedded scheme. The temporal error is computed as

$$\delta^{(n+1)} = \mathbf{U}^{(n+1)} - \hat{\mathbf{U}}^{(n+1)}, \tag{41}$$

where

$$\begin{aligned} \mathbf{U}^{(n+1)} &= \Delta t \sum_{i=1}^{s} b_i\, \mathbf{F}^{ex}(\mathbf{U}^{(i)}) + \Delta t \sum_{i=1}^{s} b_i\, \mathbf{F}^{im}(\mathbf{U}^{(i)}), \\ \hat{\mathbf{U}}^{(n+1)} &= \Delta t \sum_{i=1}^{s} \hat{b}_i\, \mathbf{F}^{ex}(\mathbf{U}^{(i)}) + \Delta t \sum_{i=1}^{s} \hat{b}_i\, \mathbf{F}^{im}(\mathbf{U}^{(i)}). \end{aligned} \tag{42}$$

Simplifying the above equations,

$$\delta^{(n+1)} = \Delta t \sum_{i=1}^{s} (b_i - \hat{b}_i)\, \mathbf{F}^{ex}(\mathbf{U}^{(i)}) + \Delta t \sum_{i=1}^{s} (b_i - \hat{b}_i)\, \mathbf{F}^{im}(\mathbf{U}^{(i)}). \tag{43}$$

The PID controller for this work uses the gains taken from Kennedy and Carpenter,[15]

$$k_I = 0.25, \quad k_P = 0.14, \quad k_D = 0.10, \quad \omega_n = 1, \tag{44}$$

where

$$p\alpha = [k_I + k_P + \left(\frac{2\omega_n}{1+\omega_n}\right) k_D], \quad p\beta = [k_P + 2\omega_n k_D], \quad p\lambda = \left(\frac{2\omega_n^2}{1+\omega_n}\right) k_D. \tag{44}$$

Therefore,

$$\alpha = \frac{.49}{p}, \quad \beta = \frac{.34}{p}, \quad \lambda = \frac{.10}{p}. \tag{44}$$

Also, an L_2 norm of the computed temporal error is used rather than an L_∞ norm of the temporal error. Refer to[23–25] for more information on adaptive time step controllers.

H. Overlapping-Partition IMEX Algorithm

The OP IMEX-RK algorithm is presented in this section to further clarify the implementation of the method. The discrete solution vector, $\mathbf{u}$, is passed into the routine and advanced in time by one time step as shown here. The routine is sequentially repeated to advance the solution to any given final time. The algorithm loops through all the block partitions, advancing the residuals containing the implicit contributions ($\mathbf{F}^{im}$) to the first RK stage. After which, the algorithm loops through all the block partitions again, advancing the residuals containing the explicit contributions ($\mathbf{F}^{ex}$) to the first RK stage. This is repeated and the discrete solution vector is advanced through all six RK stages such that the implicit and explicit residuals are advanced simultaneously. The linear solver used for the implicit solutions of the 1D problem is a freely distributed software called SPARSKIT.[22] Due to the sparse matrices resulting from the method, a direct linear solver tailored for sparse matrices was used (iluk and lusol routines specifically). The 2D problem required additional mapping of the computational domain to the solution vector and required a different sparse matrix solver. PETSc library[26] was used for the implicit linear solutions, specifically the PCLU routine from the KSP solvers, and gave good results. See algorithm 1 below for a pseudo code of the scheme.

Algorithm 1 Overlapping-Partition Implicit-Explicit Runge-Kutta Algorithm

```
for $i = 1 \to s$ do    ! loop over s-stages
    for $e = 1 \to nparts$ do    ! loop through all partitions
        for $j = \to i - 1$ do    ! explicit integration of implicit scheme
            $u = u_n + \Delta t * a_{ij}^{ex} * F^{ex} + \Delta t * A_{ij}^{im} * F^{res}$
        end for
        $u_{hat} = u$    ! store total explicit residual
        while $(newton_{res} < 1.0E - 012)$ do    ! implicit solve (Newton's method)
            Call implicit residual/Jacobian routine (solve for $F^{im}$ and $\frac{dG}{du}$)
            $G_{res} = u_{hat} - u + \Delta t * A^{im} * F^{im}$ ! Newton's residual
            $\frac{dG}{du}\delta u = -G_{res}$ ! Newton's method
            $newton_{res} = \sqrt{(G_{res})^2}$
            Call linear solver (solve for $\delta u$)
            $u = u + \delta u$    ! update $u$
        end while
    end for
    for $e = 1 \to nparts$ do    ! loop through all partitions
        Call explicit residual routine (solve for $F^{ex}$)
    end for
end for
! final integration using RK weights ($b(k)$ vector)
for $k = 1 \to s$ do
    $u = u_n + \Delta t * b(k) * F^{ex} + \Delta t * b(k) * F^{im}$
end for
```

III. 1D Numerical Test: Burger's Equation

The nonlinear viscous Burger's equation written in conservative form is

$$u_t + \left(\frac{u^2}{2}\right)_x = \epsilon\, u_{xx}, \tag{44}$$

where ϵ is the kinematic viscosity, and u is the velocity of the fluid. Burger's equation is a model equation of the NSE due to its unsteady term, nonlinear convective term, and diffusive term, which are much like the terms found in the NSE. Applying new numerical methods to Burger's equation is a common way to gain understanding of how a method performs on nonlinear hyperbolic PDEs, without the added complexities of two- and three-dimensional equations. The newly developed OP IMEX method, presented in section II, is first applied to and tested on Burger's equation. Details of the implementation and the 1D OP IMEX results will follow in this section.

A. Semi-discrete Equation and Boundary Conditions

In order to numerically solve Burger's equation, the convective and diffusive terms are discretized and approximated over the 1D computational domain using SBP operators (see section A). The Dirichlet boundary conditions are imposed using the SAT penalty method (see section B).

The semi-discrete Burger's equation, including SAT boundary conditions and initial condition ($\mathbf{u}_0$), is written as follows:

$$\begin{aligned}\mathbf{u}_t = -\frac{1}{2}\mathcal{D}U\mathbf{u} + \epsilon\mathcal{D}_2\mathbf{u} + \sigma_0\mathcal{P}^{-1}\mathbf{e}_0\left[\tilde{a}_0 u_1 - \epsilon(\mathcal{S}\mathbf{u})_1 - g_0(t)\right] \\ + \sigma_1\mathcal{P}^{-1}\mathbf{e}_1\left[\tilde{a}_1 u_N - \epsilon(\mathcal{S}\mathbf{u})_N - g_1(t)\right],\end{aligned} \tag{44}$$

where

$$\begin{aligned}&\mathbf{x} \in [x_L, x_R],\\ &g_0(t) = \frac{v(x_L,t) + |v(x_L,t)|}{3}v(x_L,t) - \epsilon v_x(x_L,t),\\ &g_1(t) = \frac{v(x_R,t) - |v(x_R,t)|}{3}v(x_R,t) - \epsilon v_x(x_R,t),\\ &U = \mathrm{diag}(\mathbf{u}), \quad \mathbf{e}_0 = (1,0,...)^T, \quad \mathbf{e}_1 = (...,0,1)^T,\\ &\tilde{a}_0 = \frac{u_1 + |u_1|}{3}, \quad \tilde{a}_1 = \frac{u_N - |u_N|}{3},\\ &\mathbf{u}_0 = v(\mathbf{x},0),\end{aligned}$$

and $\mathcal{S}$ is the boundary first derivative operator, $\mathcal{D}$ is the SBP first derivative operator matrix, $\mathcal{D}_2$ is the SBP second derivative operator matrix, and v is the exact analytical solution. Note that the convective SAT penalty contributions are formulated to act as a switch: waves convecting into the domain turn the switch 'on' and the penalty is imposed, and waves convecting out of the domain turn the switch 'off' and the penalty is zero. This is observed in $\tilde{a}_0$ and $\tilde{a}_1$, and in the first terms of $g_0(t)$ and $g_1(t)$. The energy analysis is used to determine that $\sigma_0 = -1$ and $\sigma_1 = 1$, which is derived by Fisher et al.[21] and will not be presented here. The semi-discrete equation is advanced in time at discrete locations, $t \in [0,T]$, using the IMEX-RK time integration schemes.

B. Test Problem

The exact solution to Burger's equation used to test the 1D OP IMEX-RK method is

$$v(x,t) = 1 - \tanh\left(\frac{x - t - x_0}{2\epsilon}\right), \quad x \in [-10, 10], \quad t \in [0,T], \tag{43}$$

where ϵ is the kinematic viscosity, $x_0 = 0$, and the initial condition is $u_0 = v(\mathbf{x},0)$. The problem is integrated up to time, $T = 5$. The wave propagates left-to-right as the solution advances in time (see figure 1).

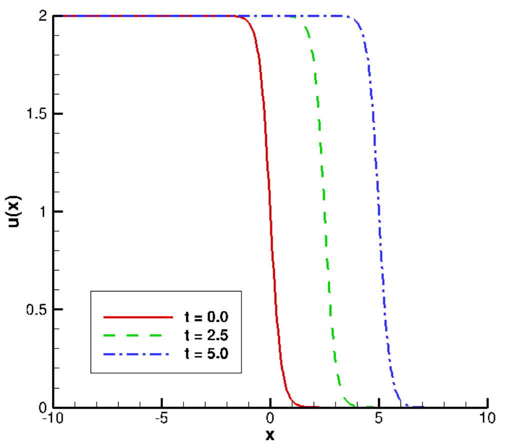


Figure 1. Discrete solution to Burger's equation on a grid with 513 equispaced points, using fourth-order accurate OP IMEX-RK integration, and fourth-order spatially accurate SBP 2-4-2 scheme.

1. *Grid Resolution Study*

The following tables show the effects of overlapping the partitions on coarse, medium, and fine grids. The fully explicit (no partitions) schemes were run with the maximum time step based on numerical stability, which was determined visually, and their corresponding maximum CFL and DFL are presented. Similarly, the OP IMEX schemes were run with varying number of overlapped partitions. The maximum CFL and DFL of the OP IMEX schemes, based on stability, are presented. Results are shown for both the second-order accurate operators (SBP 1-2-1) and the fourth-order accurate operators (SBP 2-4-2) so as to gain an understanding of the method's performance using FD stencils of different widths. The factor of increase quantifies the amount of increase in CFL and DFL that is gained by the OP IMEX scheme with respect to the fully explicit scheme.

Table 1. OP IMEX stability results on coarse grid (258 points) compared to the explicit scheme.

N = 258

	SBP 1-2-1			SBP 2-4-2		
	CFL	DFL	Factor of Increase	CFL	DFL	Factor of Increase
Fully Explicit	0.6	1.0	1.0	0.5	0.7	1.0
IMEX, overlap = 0	3.6	5.8	5.6	2.5	4.0	5.4
IMEX, overlap = 1	12.9	20.6	20.0	10.2	16.4	22.1
IMEX, overlap = 2	13.8	22.1	21.4	10.3	16.5	22.2
IMEX, overlap = 5	13.9	22.3	21.6	10.3	16.5	22.2
IMEX, overlap = 10	13.9	22.3	21.6	10.3	16.5	22.2

N = 513

	SBP 1-2-1			SBP 2-4-2		
	CFL	DFL	Factor of Increase	CFL	DFL	Factor of Increase
Fully Explicit	0.3	1.0	1.0	0.2	0.7	1.0
IMEX, overlap = 0	1.5	4.8	4.8	1.1	3.4	5.3
IMEX, overlap = 1	12.4	39.7	40.3	9.2	29.5	45.0
IMEX, overlap = 2	12.8	41.0	41.7	9.2	29.5	45.0
IMEX, overlap = 5	12.8	40.8	41.5	9.2	29.5	45.0
IMEX, overlap = 10	12.8	41.0	41.7	9.2	29.5	45.0

N = 1014

	SBP 1-2-1			SBP 2-4-2		
	CFL	DFL	Factor of Increase	CFL	DFL	Factor of Increase
Fully Explicit	0.2	1.0	1.0	0.1	0.8	1.0
IMEX, overlap = 0	0.7	4.5	4.4	0.5	3.3	4.3
IMEX, overlap = 1	8.4	53.2	51.9	7.3	46.2	60.0
IMEX, overlap = 2	10.6	67.3	65.6	7.9	50.0	65.0
IMEX, overlap = 5	10.6	67.3	65.6	7.9	50.0	65.0
IMEX, overlap = 10	10.6	67.3	65.6	7.9	50.0	65.0

2. Viscosity Study

A study was performed on the affect of overlapping the partitions while varying the viscosity. Similar to the grid resolution study, the maximum CFL and DFL are presented for the fully explicit scheme and the various overlaps with the IMEX scheme.

Table 2. OP IMEX stability results on medium grid (513 points) with a diffusion coefficient of 0.0625

$\epsilon = 0.0625$

	SBP 1-2-1			SBP 2-4-2		
	CFL	DFL	Factor of Increase	CFL	DFL	Factor of Increase
Fully Explicit	1.2	0.9	1.0	1.0	0.7	1.0
IMEX, overlap = 0	9.9	7.9	8.2	6.7	5.4	6.7
IMEX, overlap = 1	14.0	11.2	11.7	11.2	9.0	11.2
IMEX, overlap = 2	13.9	11.1	11.6	11.1	8.9	11.1
IMEX, overlap = 5	13.8	11.0	11.5	10.9	8.7	10.9
IMEX, overlap = 10	14.1	11.3	11.7	11.0	8.8	11.0

$\epsilon = 0.125$

	SBP 1-2-1			SBP 2-4-2		
	CFL	DFL	Factor of Increase	CFL	DFL	Factor of Increase
Fully Explicit	0.6	1.0	1.0	0.5	0.7	1.0
IMEX, overlap = 0	3.5	5.6	5.7	0.5	0.7	1.0
IMEX, overlap = 1	13.2	21.1	21.5	10.2	16.3	22.1
IMEX, overlap = 2	13.8	22.0	22.4	10.2	16.4	22.2
IMEX, overlap = 5	13.9	22.2	22.6	10.2	16.4	22.2
IMEX, overlap = 10	13.9	22.3	22.7	10.2	16.4	22.2

$\epsilon = 0.25$

	SBP 1-2-1			SBP 2-4-2		
	CFL	DFL	Factor of Increase	CFL	DFL	Factor of Increase
Fully Explicit	0.3	1.0	1.0	0.2	0.7	1.0
IMEX, overlap = 0	1.5	4.8	4.8	1.1	3.4	5.3
IMEX, overlap = 1	12.4	39.7	40.3	9.2	29.5	45.0
IMEX, overlap = 2	12.8	40.0	41.7	9.2	29.5	45.0
IMEX, overlap = 5	12.8	40.8	41.5	9.2	29.5	45.0
IMEX, overlap = 10	12.8	41.0	41.7	9.2	29.5	45.0

C. Discussion

It is observed from both studies that the DFL is the limiting condition (DFL ≈1 for the explicit schemes while the CFL varies) in this problem. The OP IMEX scheme shows promising results; a maximum factor of increase of ≈65 on the fine grid, and ≈22 on the coarse grid. As the grid is refined (Table 1) the numerical stability increases. The viscosity study (Table 2) shows that added viscosity also increases the stability. Also, a single overlap in the partitions yields essentially the same increase in stability as multiple overlaps. Referring back to the formulation of the overlapping partitions, section E, a single overlap causes the explicit contributions to be duplicates of terms contained in the implicit blocks. This appears to be key in the added stability of the OP IMEX scheme.

D. Design Order: Burger's equation

Verifying the design order is an important validation of the mathematical correctness of the OP IMEX method. The spatial and temporal convergence tests are presented in Figure 2, showing that the OP IMEX methodology and the implementation are correct for Burger's equation. The design order for the OP IMEX method with no overlaps and with a single overlap, verifying that overlapping the partitions in this way is also correct. All plots are in log-log scale in order to more readily see the order of convergence (as a slope rather than a power). Tables containing the exact design order error values are given in appendix A.

IV. 2D Numerical Test: Advection Equation

Significant increase in numerical stability was demonstrated using the OP IMEX method applied to a 1D problem (Burger's equation). The FD matrix structure for solutions to 2D problems is different than those of 1D problems. Hence, the next part of this work is to develop the OP IMEX method applied to 2D problems, and to test its performance on a simple problem. The 2D advection equation was chosen for this study; it is a linear first-order hyperbolic PDE,

$$u_t + A\,u_x + B\,u_y = 0, \tag{43}$$

where A is the wave speed in the x-direction, and B is the wave speed in the y-direction. Note that it lacks a diffusive term and is purely convective, meaning that only the CFL condition will be studied in this problem (no DFL condition).

A. Semi-discrete Equation and Boundary Conditions

In order to numerically solve the 2D advection equation, the differential equation is transformed into a system of algebraic equations by the following: the domain is discretized, the derivatives are approximated using SBP operators, and the boundary conditions are imposed using the SAT penalty method. The derivative approximations are computed using SBP operators in a line-by-line fashion (i.e., by looping over the grid points in y and performing the derivatives with respect to x and then looping over the grid points in x performing the derivatives with respect to y). This is necessary because the SBP operator matrices approximate the derivative of a vector or line and is not applicable to a full 2D domain by a single matrix operation.

The semi-discrete 2D advection equation, including SAT boundary conditions and initial condition ($\mathbf{u}_0$), is written as follows:

$$\begin{aligned}\mathbf{u}_t = &- A\,\mathcal{D}\,\mathbf{u}(:,y_i) - B\,\mathcal{D}\,\mathbf{u}(x_i,:)\\ &+ \sigma_0\mathcal{P}^{-1}\mathbf{e}_0\left[\tilde{a}_L\,u(x_1,y_i) - g_L(t)\right]\\ &+ \sigma_1\mathcal{P}^{-1}\mathbf{e}_1\left[\tilde{a}_R\,u(x_N,y_i) - g_R(t)\right]\\ &+ \sigma_0\mathcal{P}^{-1}\mathbf{e}_0\left[\tilde{a}_B\,u(x_i,y_1) - g_B(t)\right]\\ &+ \sigma_1\mathcal{P}^{-1}\mathbf{e}_1\left[\tilde{a}_T\,u(x_i,y_N) - g_T(t)\right],\end{aligned} \tag{43}$$

where

$$\begin{aligned}&\mathbf{x} \in [x_L, x_R], \quad \mathbf{y} \in [y_B, y_T],\\ &g_L(t) = \frac{v(x_1,y_i,t) + |v(x_1,y_i,t)|}{2}v(x_1,y_i,t),\\ &g_R(t) = \frac{v(x_N,y_i,t) - |v(x_N,y_i,t)|}{2}v(x_N,y_i,t),\\ &g_B(t) = \frac{v(x_i,y_1,t) + |v(x_i,y_1,t)|}{2}v(x_i,y_1,t),\\ &g_T(t) = \frac{v(x_i,y_N,t) - |v(x_i,y_N,t)|}{2}v(x_i,y_N,t),\\ &\mathbf{e}_0 = (1,0,...)^T, \quad \mathbf{e}_1 = (...,0,1)^T,\\ &\tilde{a}_L = \frac{A+|A|}{2}, \quad \tilde{a}_R = \frac{A-|A|}{2}, \quad \tilde{a}_B = \frac{B+|B|}{2}, \quad \tilde{a}_R = \frac{B-|B|}{2}\\ &\mathbf{u}_0 = v(\mathbf{x},\mathbf{y},0),\end{aligned}$$

and $1 \geq i \geq N$, $\mathcal{D}$ is the SBP first derivative operator matrix, $\sigma_0 = -1$ and $\sigma_1 = 1$ (determined from energy analysis), and v is the exact analytical solution. The SAT penalty boundary conditions are the same as those imposed for Burger's equation without the viscous terms.

B. Test Problem

The exact solution to the 2D advection equation used to test the 2D OP IMEX-RK method is

$$v(x,y,t) = \sin(2\pi(x - A\,t))\sin(2\pi(y - B\,t)), \quad x \in [0, 0.5], \quad y \in [0, 0.5], \quad t \in [0, T], \tag{42}$$

where A is the wave speed in the x-direction, B is the wave speed in the y-direction, and the initial condition is given by $u_0 = v(\mathbf{x},\mathbf{y},0)$. The problem is integrated up to time, $T = 7$. In figure 3 below, the propagation of the 2D wave is shown up to time, $T \approx 2$. The wave repeatedly propagates through the domain as the equation is integrated farther in time. Note that the magnitude is scaled by 0.1 for better visualization.

C. Results

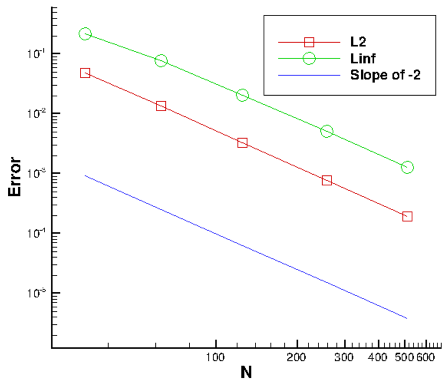
The numerical stability of the OP IMEX method solving the 2D advection equation was assessed on four different grid resolutions: 30 x 30, 60 x 60, 90 x 90, and 120 x 120. These solution domains are mapped to matrices of sizes: 400 x 400, 3600 x 3600, 8100 x 8100, and 14400 x 14400, for the implicit solves, respectively. Recall that the stability of Burger's equation was dependent upon the grid resolution, hence the 2D results are given on several grid sizes. Note that there is no diffusive term in the 2D advection equation as it is purely convective, meaning that there is no DFL stability criteria. The CFL stability will be discussed in this section.

The variable time stepping error tolerance was varied to determine the stability of the OP IMEX method. The linear behavior of the advection equation yielded slowly diverging solutions, unlike the quickly diverging solutions of the nonlinear Burger's equation. This made is necessary to choose a stability parameter, which was chosen to be the magnitude of the advecting wave, C, where the exact solution wave magnitude is $C = 1$. The approximate solution was considered unstable when $C \geq 1.5$. The edge of the explicit stability regime was found by adjusting the error tolerance to be as large as possible while still maintaining a stable explicit solution. This was done for both SBP 1-2-1 and SBP 2-4-2 schemes on all four grid resolutions and is plotted in figure 4. The CFL oscillates until the controller locks into the stable time step after approximately 100 and 150 time steps for the SBP 1-2-1 and SBP 2-4-2 schemes, respectively. The second-order scheme (SBP 1-2-1) is stable at $CFL \approx 2$ and the fourth-order scheme (SBP 2-4-2) is stable at $CFL \approx 1.5$, which is the baseline explicit stability that the OP IMEX scheme will be compared against. The L_2 norms are computed for the explicit solutions for comparison with the OP IMEX solutions and will be presented later.

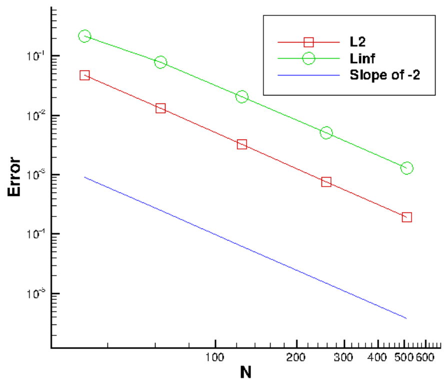
The partitioning of the computational domain into blocks introduces more error into the solution as compared to the full FD domain used for the purely explicit baseline runs. A very small error tolerance in the controller, on the order of 1E-6 or smaller, will drive the OP IMEX solutions to have a smaller CFL than the explicit baseline result because of this added error, even though the scheme is numerically stable at much higher CFL values. In order to use the controller as a stability evaluation tool, two different controller tolerance values, ϵ = 5E-3 and ϵ = 1E-2, were used to test the scheme on the different grid sizes. Testing showed that those values gave good indication of the upper stability limit of the OP IMEX method.

Figures 5 - 8 show how the controller adjusted the CFL over the span of the integration for the explicit (EX), implicit (IM), IMEX with no overlaps (IMEX-0), IMEX with one overlap (IMEX-1), IMEX with five overlaps (IMEX-5), and IMEX with nine overlaps (IMEX-9) schemes while using an error tolerance of ϵ = 5E-3. The CFL of the implicit scheme continues to grow through out the entire integration as it is unconditionally stable. The CFL of explicit solution stays at the baseline values of ≈ 2 and ≈ 1.5 for grids 30 x 30 and 60 x 60, and is unstable for the 90 x 90 and 120 x 120 grids. The CFL of the IMEX scheme with no overlaps and one overlap is unstable for all the grids expect for the 30 x 30, where it increases and then slowly decreases due to instability. The CFL of the IMEX scheme with nine overlaps is stable for all grids and reached a maximum of CFL $\approx$ 8 for the SBP 1-2-1 scheme, and CFL $\approx$ 7 for the SBP 2-4-2, both on the 120 x 120 grid. A general trend is seen of a slight increase in the CFL as the grid resolution is increased for both the 1-2-1 scheme and the wider 2-4-2 scheme.

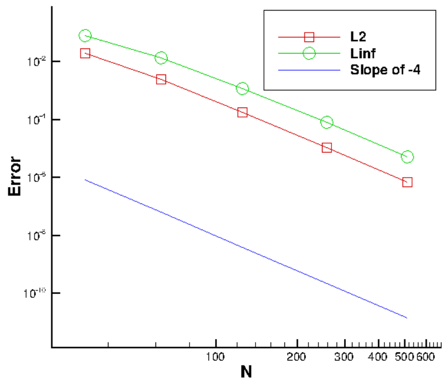
Figures 9 - 12 show the CFL vs. Time Step results with an increased controller error tolerance of ϵ = 1E-2. The results are similar to those presented in figures 5 - 8 for the error tolerance of ϵ = 5E-3, except the increased error tolerance allows the controller to show higher stable CFL values for the IMEX schemes; CFL $\approx$ 10 for the 1-2-1 scheme and CFL $\approx$ 9 for the 2-4-2 scheme, when solved on the fine grid (120 x 120).



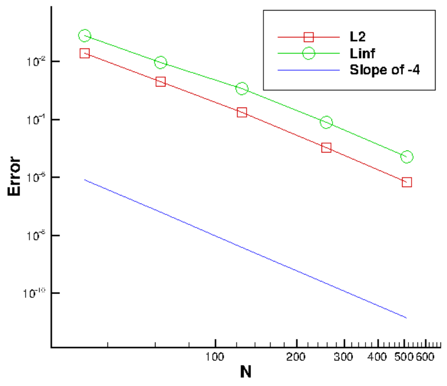
(a) Spatial design order of two: No overlaps



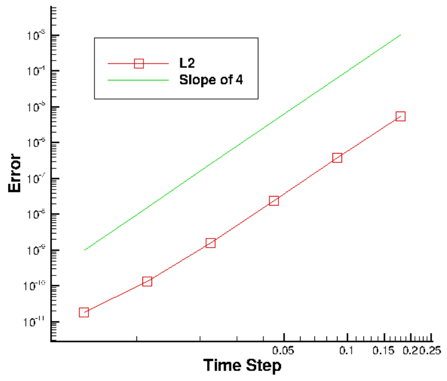
(b) Spatial design order of two: Single overlap



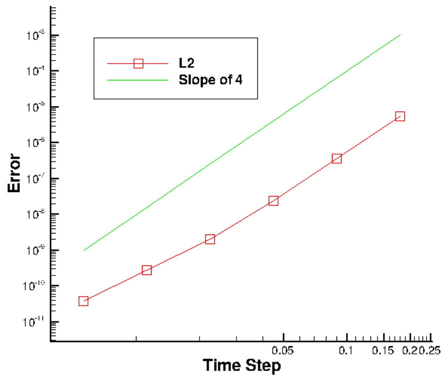
(c) Spatial design order of four: No overlaps



(d) Spatial design order of four: Single overlap

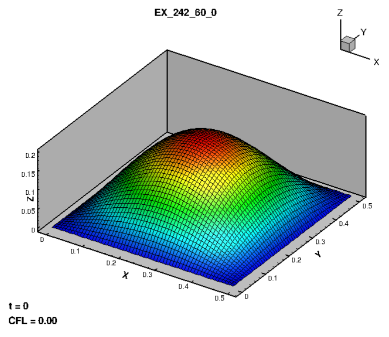


(e) Temporal design order of four: No overlaps

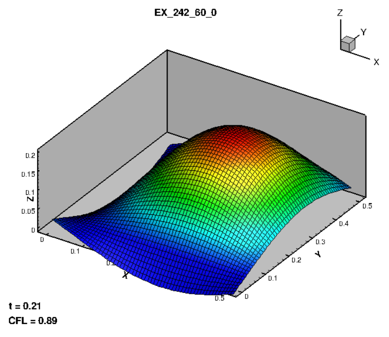


(f) Temporal design order of four: Single overlap

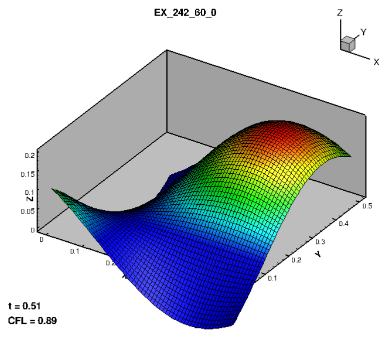
Figure 2. Burger's equation L_2 and L_∞ spatial and temporal design order for OP IMEX scheme.



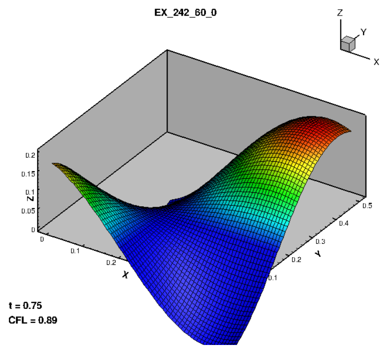
(a) t=0



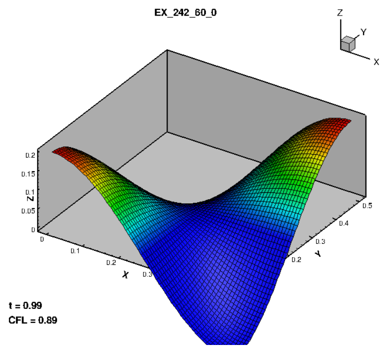
(b) t=0.21



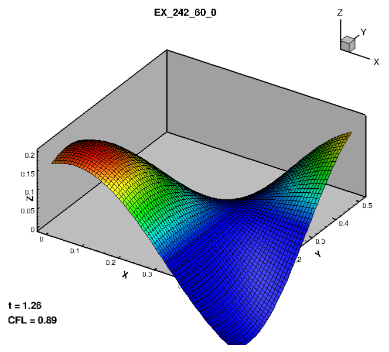
(c) t=0.51



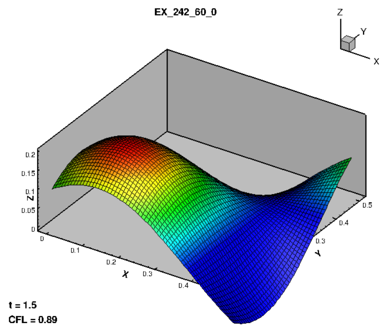
(d) t=0.75



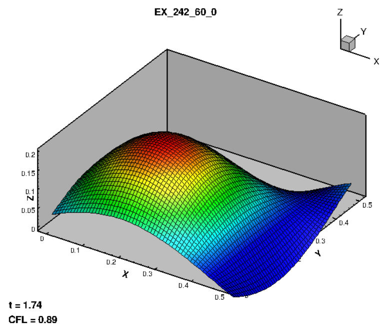
(e) t=0.99



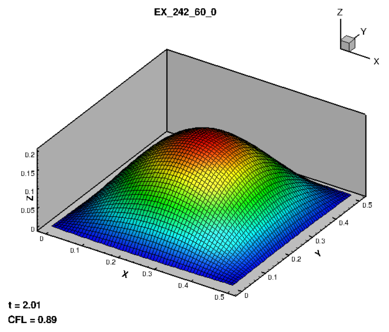
(f) t=1.26



(g) t=1.5

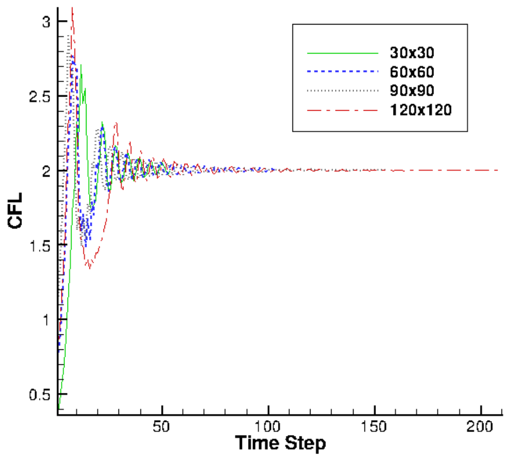


(h) t=1.74

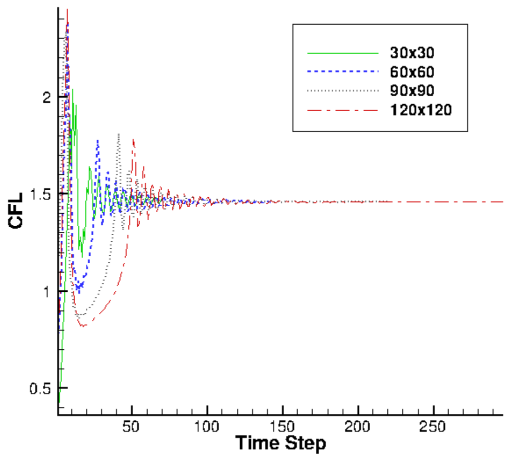


(i) t=2.01

Figure 3. The approximate solution to the 2D advection equation using a 60 x 60 domain, explicit time stepping, and SBP 2-4-2 derivative approximations.

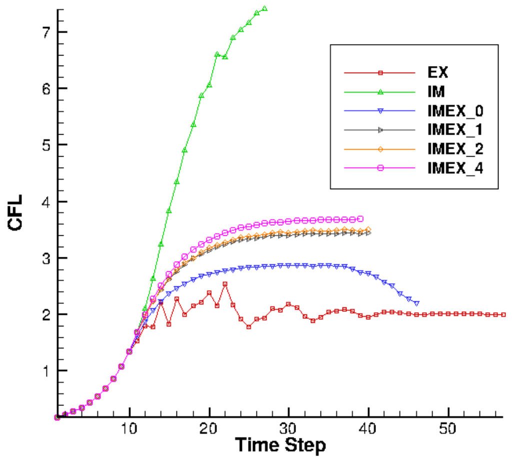


(a) SBP 1-2-1 Scheme

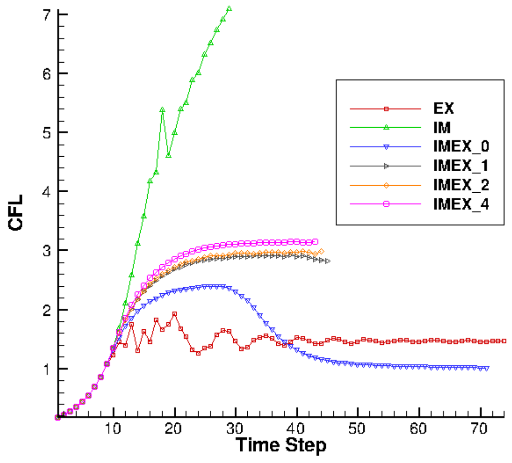


(b) SBP 2-4-2 Scheme

Figure 4. Behavior of CFL vs. Time Step as the adaptive time step controller locks into a stable CFL for the ERK schemes on various grid resolutions.

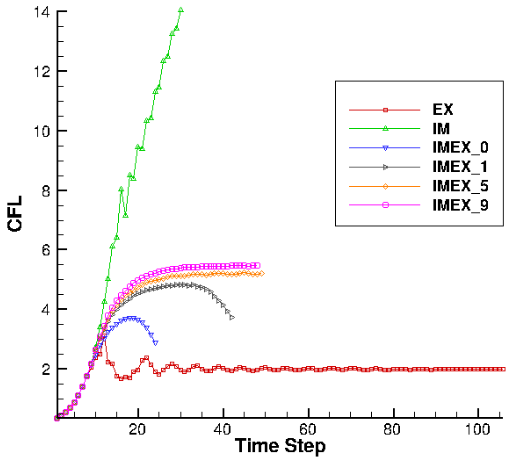


(a) SBP 1-2-1 Scheme

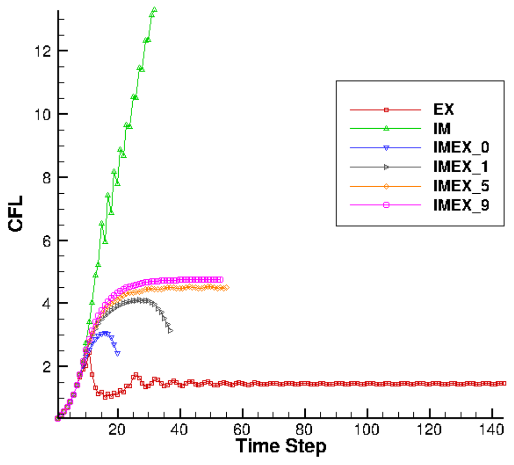


(b) SBP 2-4-2 Scheme

Figure 5. CFL vs. Time step with an error tolerance of ϵ = 5E-3 on the 30 x 30 grid.

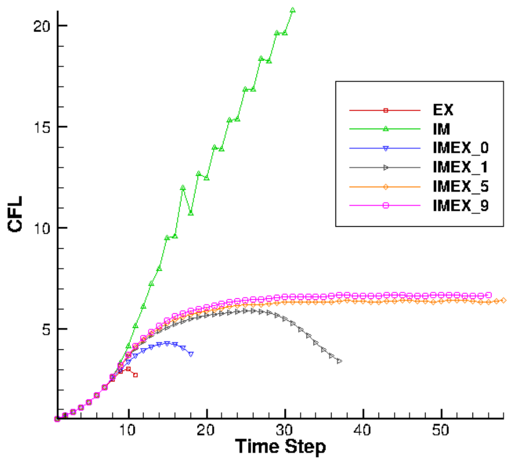


(a) SBP 1-2-1 Scheme

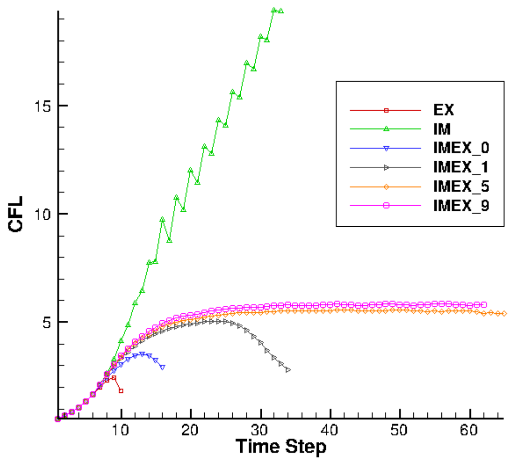


(b) SBP 2-4-2 Scheme

Figure 6. CFL vs. Time step with an error tolerance of ϵ = 5E-3 on the 60 x 60 grid.

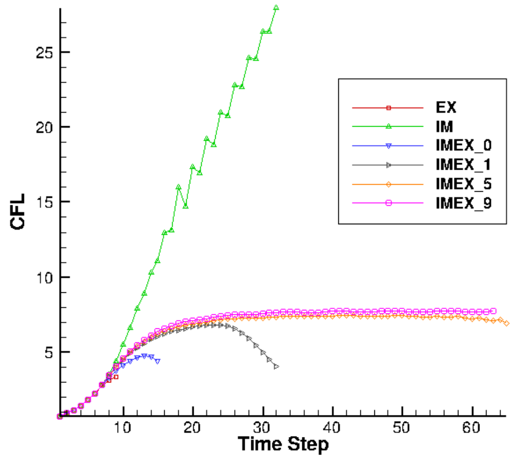


(a) SBP 1-2-1 Scheme

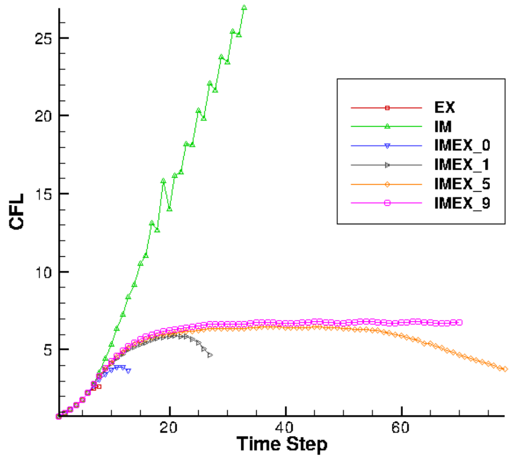


(b) SBP 2-4-2 Scheme

Figure 7. CFL vs. Time step with an error tolerance of ϵ = 5E-3 on the 90 x 90 grid.

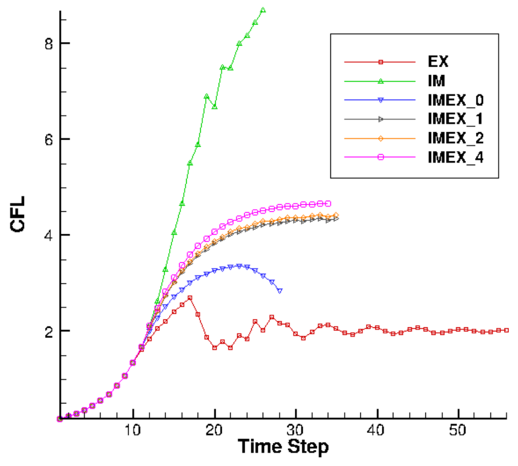


(a) SBP 1-2-1 Scheme

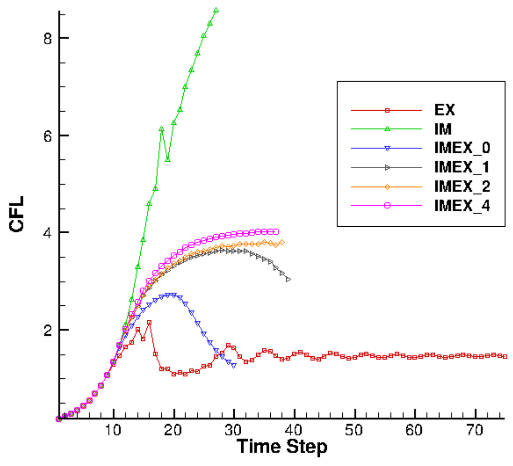


(b) SBP 2-4-2 Scheme

Figure 8. CFL vs. Time step with an error tolerance of ϵ = 5E-3 on the 120 x 120 grid.

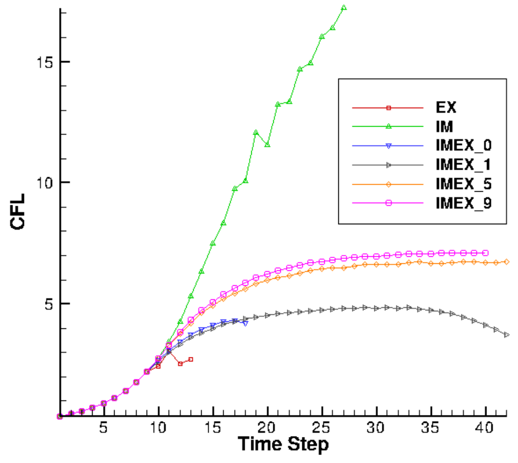


(a) SBP 1-2-1 Scheme

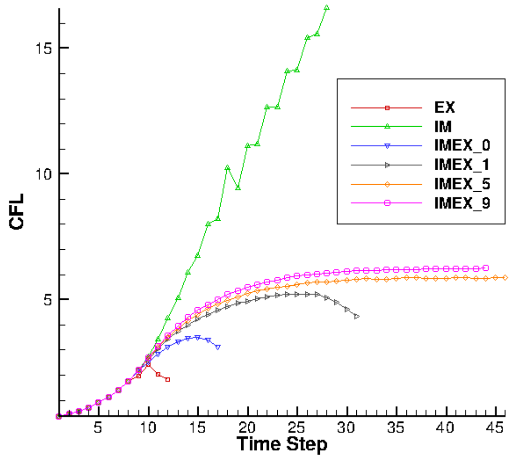


(b) SBP 2-4-2 Scheme

Figure 9. CFL vs. Time step with an error tolerance of ϵ = 1E-2 on the 30 x 30 grid.

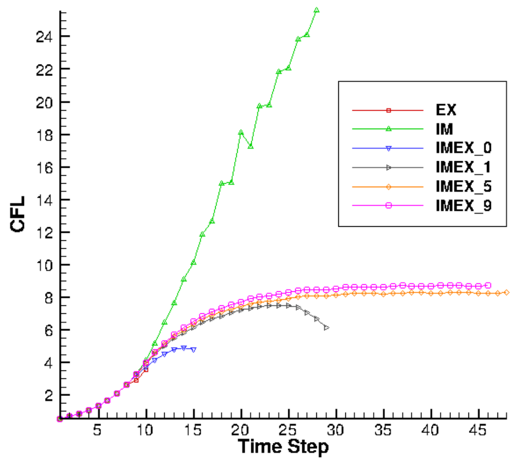


(a) SBP 1-2-1 Scheme

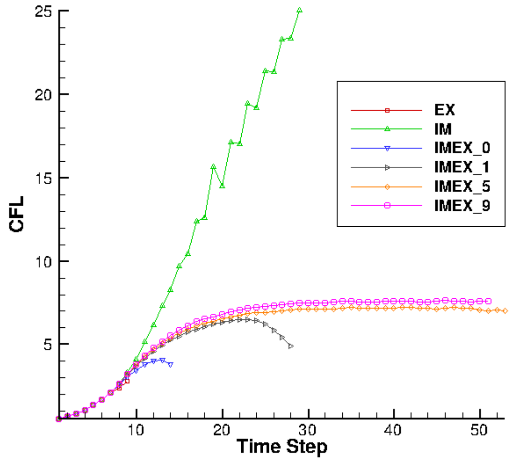


(b) SBP 2-4-2 Scheme

Figure 10. CFL vs. Time step with an error tolerance of ϵ = 1E-2 on the 60 x 60 grid.

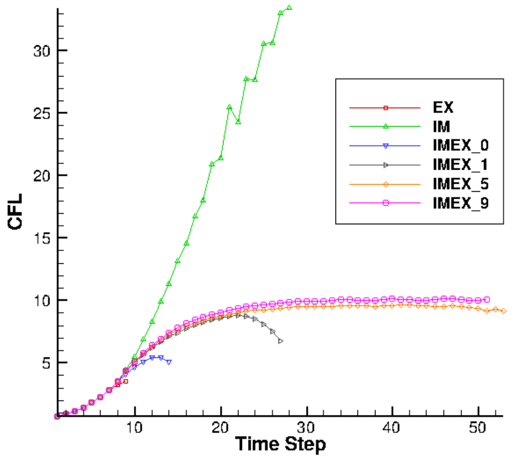


(a) SBP 1-2-1 Scheme

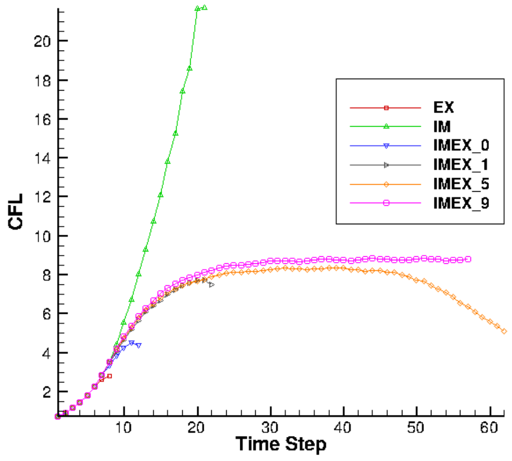


(b) SBP 2-4-2 Scheme

Figure 11. CFL vs. Time step with an error tolerance of ϵ = 1E-2 on the 90 x 90 grid.



(a) SBP 1-2-1 Scheme

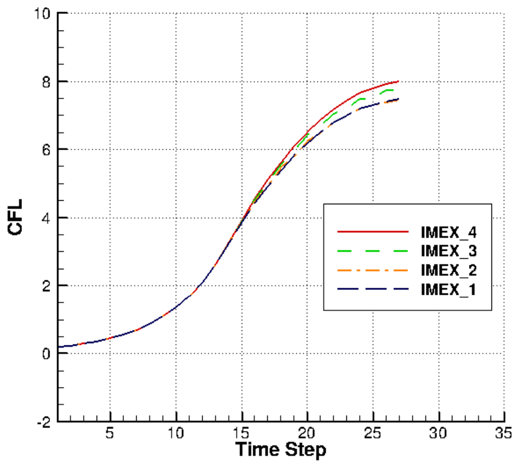


(b) SBP 2-4-2 Scheme

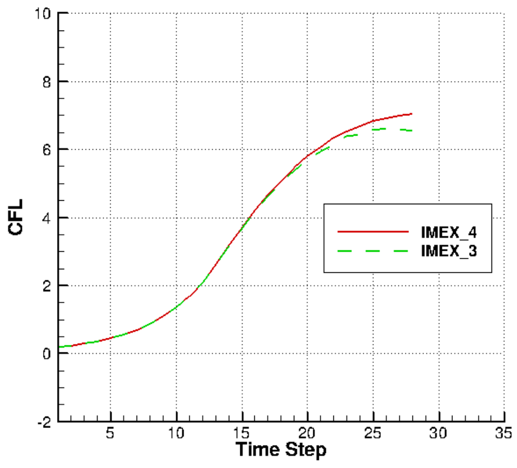
Figure 12. CFL vs. Time step with an error tolerance of ϵ = 1E-2 on the 120 x 120 grid.

The final set of results presented here are plots of the CFL vs. Time Step of the IMEX schemes ran with a larger time step error tolerance, $\epsilon = 5$E-2. At this larger error tolerance, the controller allows the time step to increase to the upper bound of stability. In fact, several overlaps are required to obtain a stable solution with this large error tolerance in some cases.

Figure 13 shows results of the two schemes on the 30 x 30 grid. The 2-4-2 scheme was not stable with 1 or 2 overlaps and reached a CFL$\approx$ 7 with 4 overlaps. The 1-2-1 scheme was stable at a slightly higher CFL, around 8, and was stable with a single overlap.



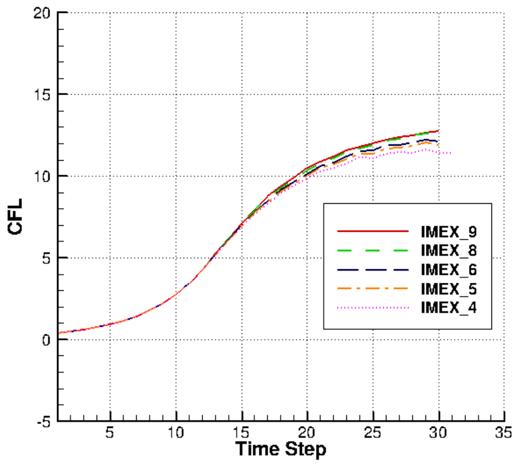
(a) SBP 1-2-1 Scheme



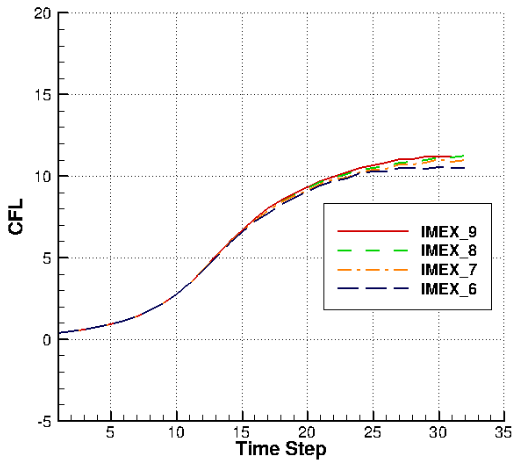
(b) SBP 2-4-2 Scheme

Figure 13. CFL vs. Time step with an error tolerance of ϵ = 5E-2 on the 30 x 30 grid.

The stability on the 60 x 60 grid, figure 14, increased from the 30 x 30 grid but required more overlaps to maintain stability; 4 for the 1-2-1 scheme and 6 for the 2-4-2 scheme. The 1-2-1 scheme with 7 overlaps went unstable just before the end and is not shown in the plot.



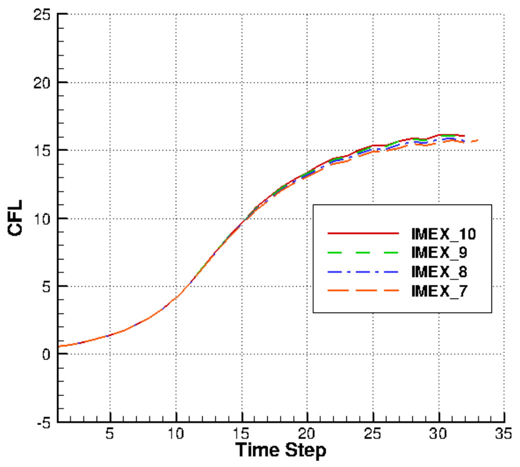
(a) SBP 1-2-1 Scheme



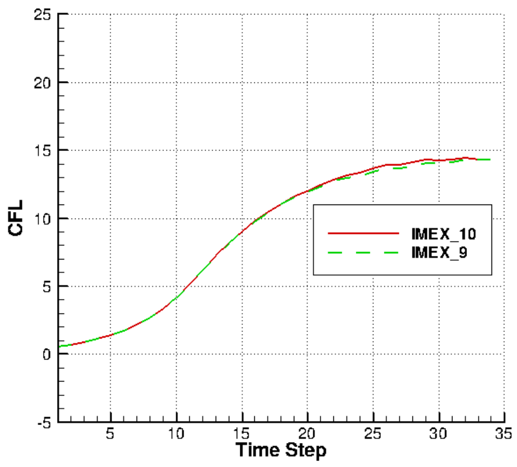
(b) SBP 2-4-2 Scheme

Figure 14. CFL vs. Time step with an error tolerance of ϵ = 5E-2 on the 60 x 60 grid.

In figure 15, the 90 x 90 grid required 7 overlaps for the 1-2-1 scheme, and 9 overlaps for the 2-4-2 scheme in order to maintain stability. The results show the same trends: an increase in stability for both schemes as compared to the coarser grids, and the 1-2-1 scheme has a slightly higher stability than the 2-4-2 scheme.



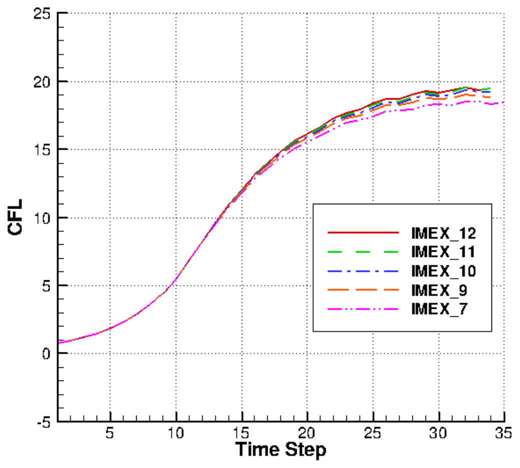
(a) SBP 1-2-1 Scheme



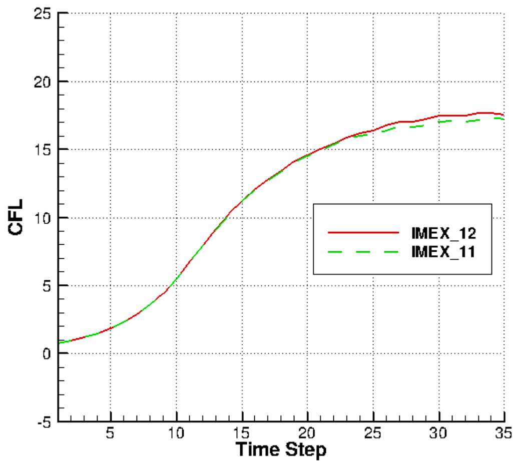
(b) SBP 2-4-2 Scheme

Figure 15. CFL vs. Time step with an error tolerance of ϵ = 5E-2 on the 90 x 90 grid.

The greatest increase in stability again occurred with the finest grid resolution, as seen on the 120 x 120 grid in figure 16. The 1-2-1 scheme reached a CFL≈ 20 with 12 overlaps and the 2-4-2 scheme reached a CFL≈ 18 with 12 overlaps. The 2-4-2 scheme was unstable with less than 11 overlaps, as was the 1-2-1 scheme with less than 7 overlaps.



(a) SBP 1-2-1 Scheme



(b) SBP 2-4-2 Scheme

Figure 16. CFL vs. Time step with an error tolerance of ϵ = 5E-2 on the 120 x 120 grid.

Table 3 shows the factor of increase, which is the amount of increase the OP IMEX scheme achieved as compared to the fully explicit scheme, for both schemes (SBP 1-2-1 and SBP 2-4-2) on the four different grid sizes ran with the largest controller error tolerance, ϵ = 5E-2. The unstable results are marked with a '-'. The trends explained above are seen in the tables below. The tables are provided for easy comparison with the results from Burger's equation. The greatest factor of increase was ≈ 10 for the 1-2-1 scheme and ≈ 12 for the 2-4-2 scheme, both solved with the 120 x 120 grid. Theses results are on the same order of magnitude as the Burger's equation results with the lowest viscosity, which is reasonable because the 2D advection equation is purely convective (no dissipation).

Table 3. Stability comparison of OP IMEX and explicit schemes solved on a grid with 30 x 30 points.

N_x=30, N_y=30

	SBP 1-2-1		SBP 2-4-2	
	CFL	Factor of Increase	CFL	Factor of Increase
Fully Explicit	2.0	1.0	1.5	1.0
IMEX, overlap = 0	-	-	-	-
IMEX, overlap = 1	7.5	3.8	-	-
IMEX, overlap = 2	7.4	3.7	-	-
IMEX, overlap = 3	7.7	3.9	6.6	4.4
IMEX, overlap = 4	8.0	4.0	7.0	4.7

N_x=60, N_y=60

	SBP 1-2-1		SBP 2-4-2	
	CFL	Factor of Increase	CFL	Factor of Increase
Fully Explicit	2.0	1.0	1.5	1.0
IMEX, overlap = 3	-	-	-	-
IMEX, overlap = 4	11.6	5.8	-	-
IMEX, overlap = 5	12.0	6.0	-	-
IMEX, overlap = 6	12.2	6.1	10.6	7.1
IMEX, overlap = 7	-	-	11.0	7.3
IMEX, overlap = 8	12.6	6.3	11.2	7.5
IMEX, overlap = 9	12.7	6.4	11.2	7.5

N_x=90, N_y=90

	SBP 1-2-1		SBP 2-4-2	
	CFL	Factor of Increase	CFL	Factor of Increase
Fully Explicit	2.0	1.0	1.5	1.0
IMEX, overlap = 6	-	-	-	-
IMEX, overlap = 7	15.7	7.9	-	-
IMEX, overlap = 8	15.8	7.9	-	-
IMEX, overlap = 9	16.0	8.0	14.3	9.5
IMEX, overlap = 10	16.1	8.1	14.4	9.6

N_x=120, N_y=120

	SBP 1-2-1		SBP 2-4-2	
	CFL	Factor of Increase	CFL	Factor of Increase
Fully Explicit	2.0	1.0	1.5	1.0
IMEX, overlap = 6	-	-	-	-
IMEX, overlap = 7	18.5	9.3	-	-
IMEX, overlap = 8	-	-	-	-
IMEX, overlap = 9	19.0	9.5	-	-
IMEX, overlap = 10	19.2	9.6	-	-
IMEX, overlap = 11	19.5	9.8	17.2	11.5
IMEX, overlap = 12	19.5	9.8	17.7	11.8

As a sanity check on the solutions obtained with the largest time step controller tolerance, the L_2 error norms are presented in table 4. They are computed using the approximate solution and the exact analytic solution. The L_2 error norm of the explicit solution ran with the highest stable controller tolerance are also presented for comparison.

The majority of the solutions have an error with the same order-of-magnitude as the explicit solution, or one order-of-magnitude larger than the explicit solution. There appears to be no trends with the number of overlaps or SBP scheme and the amount of error. The time step controller shows oscillatory behavior when near the stability envelop which could cause a scatter in errors reported here. When the solution ends, if it is at the upper end of the oscillation the error would be larger, whereas the opposite would be observed if it's trending downward or decreasing the time step when the solution ends. Also, the L_2 error norms are expected to be larger when at the edge of stability as in these cases. This confirms that the results presented in this section are reliable.

Table 4. Comparison of L_2 error norms of fully explicit and OP IMEX schemes on 120 x 120 grid.

L_2 Error Norms

N_x=120, N_y=120	SBP 1-2-1	SBP 2-4-2	ϵ
Fully Explicit	0.01081	0.00931	0.002
IMEX, overlap = 7	0.20030	-	0.05
IMEX, overlap = 8	-	-	0.05
IMEX, overlap = 9	0.03186	-	0.05
IMEX, overlap = 10	0.08419	-	0.05
IMEX, overlap = 11	0.13228	0.04079	0.05
IMEX, overlap = 12	0.16849	0.13382	0.05

L_2 Error Norms

N_x=90, N_y=90	SBP 1-2-1	SBP 2-4-2	ϵ
Fully Explicit	0.03661	0.02579	0.0024
IMEX, overlap = 7	0.31106	-	0.05
IMEX, overlap = 8	0.03421	-	0.05
IMEX, overlap = 9	0.08305	0.05441	0.05
IMEX, overlap = 10	0.11561	0.11999	0.05

L_2 Error Norms

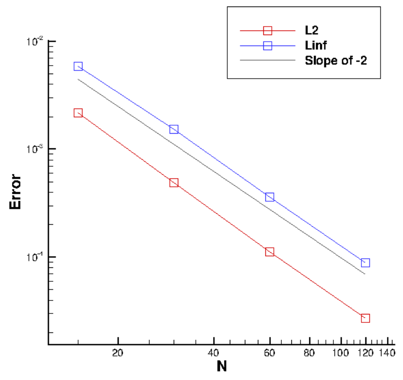
N_x=60, N_y=60	SBP 1-2-1	SBP 2-4-2	ϵ
Fully Explicit	0.01451	0.05119	0.0058
IMEX, overlap = 4	0.34242	-	0.05
IMEX, overlap = 5	0.08720	-	0.05
IMEX, overlap = 6	0.17650	0.17580	0.05
IMEX, overlap = 7	-	0.30452	0.05
IMEX, overlap = 8	0.33656	0.38389	0.05
IMEX, overlap = 9	0.38799	0.06696	0.05

L_2 Error Norms

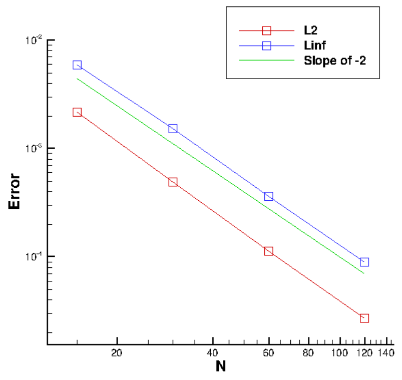
N_x=30, N_y=30	SBP 1-2-1	SBP 2-4-2	ϵ
Fully Explicit	0.13594	0.03471	0.0157
IMEX, overlap = 0	-	-	0.05
IMEX, overlap = 1	0.16943	-	0.05
IMEX, overlap = 2	0.14951	-	0.05
IMEX, overlap = 3	0.34547	0.09775	0.05
IMEX, overlap = 4	0.47630	0.25975	0.05

V. Design Order

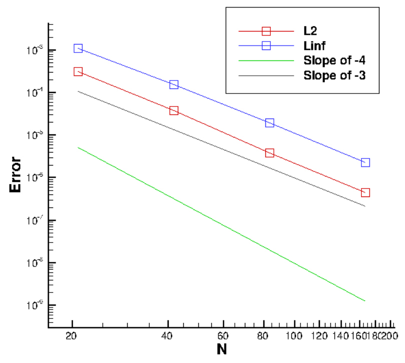
Verifying the design order is an important validation of the mathematical correctness of the 2D OP IMEX method. The spatial and temporal convergence tests are presented in Figure 17, showing that the OP IMEX methodology and the implementation are correct for the 2D advection equation. All plots are in log-log scale in order to more readily see the order of convergence (as a slope rather than a power). Tables containing the exact design order error values are shown in appendix A.



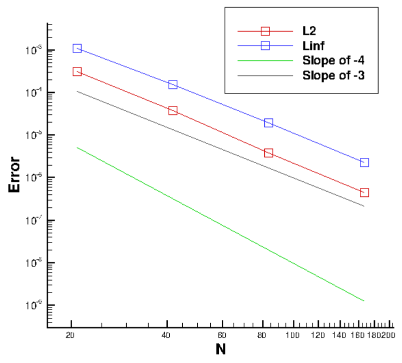
(a) Spatial design order of two: No overlaps



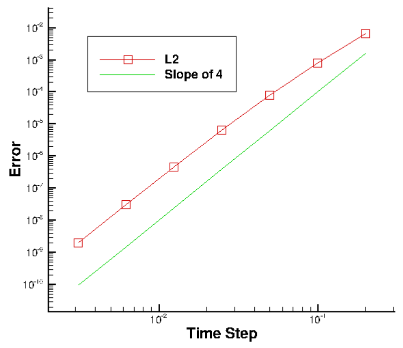
(b) Spatial design order of two: Two overlaps



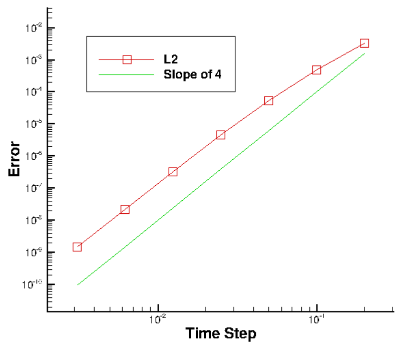
(c) Spatial design order of three: No overlaps



(d) Spatial design order of three: Two overlaps



(e) Temporal design order of four: No overlaps



(f) Temporal design order of four: Two overlaps

Figure 17. L_2 and L_∞ design order for SBP 1-2-1 spatial scheme with IMEX temporal advancement.

VI. Conclusion

A high-order implicit-explicit multi-block time stepping method (OP IMEX) for hyperbolic PDEs was formulated and tested on the 1D nonlinear viscous Burger's equation and the 2D linear advection equation. The first and second derivatives were approximated using both second- and fourth-order accurate SBP operators (FD based), SBP 1-2-1 and SBP 2-4-2, respectively.[16] The semi-discrete system of ODEs is integrated in time using additive 6-stage Runge-Kutta schemes;[15] an explicit scheme (ERK) and an implicit scheme (ESDIRK), which are both fourth-order accurate in time.

The computational domain, 1D or 2D, was split into partitions or blocks. The points inside the blocks were implicitly solved independently of the rest of the domain (at each RK stage), while the terms coupling the blocks together were solved explicitly. The numerical stability of the method was evaluated by comparing the maximum CFL and DFL attainable while maintaining a stable solution against those of the fully explicit scheme. Solving the partitioned domain using the IMEX scheme alone did not increase the stability. However, overlapping portions of the blocks and solving the overlapped portions twice significantly increased the numerical stability.

The IMEX scheme with overlapping partitions significantly reduces the CFL and DFL limitations for numerical stability for both the second-order (SBP 1-2-1) and the fourth-order (SBP 2-4-2) spatial operators. The grid resolution and viscosity studies show that the greatest increase in the stability limitations occurs with more viscous problems. As the viscosity is increased stability limitations decrease. As the number of equispaced nodes is increased, the stability limitations decrease as well.

A. Implicit Solutions Using Direct Solving Methods

Implicit solutions require solving the entire domain simultaneously. The large dimensionality of full 3D CFD simulations, which can be on the order of $1E8$ - $1E9$, force the use of iterative methods simply because direct solving methods for systems of that size are not feasible. However, the OP IMEX method presented in this work may provide a means for using direct solving methods in full 3D CFD.

Nested Dissection is the optimal ordering for a direct solving method on a 3D domain, and scales as N^6 operations for a nearest neighbor connectivity pattern. Assuming a processor can do $1E9$ FLOPS (FLoating-point Operations Per Second) and that the 3D domain is partitioned into blocks of the size N x N x N, then if N $\approx$ 30 the block can be inverted in $\approx$ 1 second by one CPU. If each partition is solved by an individual processor in a parallel fashion, direct solving methods for implicit methods used in full 3D CFD become feasible.

The blocks are overlapped enough to gain the added increase in numerical stability, yet they are kept to a manageable size for inverting the linear matrices using direct solving methods. This newly developed OP IMEX method requires further evaluation, but it appears to have significant new advantages for CFD simulations.

Appendix

The ESDIRK and ERK schemes used in this work are presented below.

Runge-Kutta Schemes

Fourth-Order Explicit Runge-Kutta (ERK) Scheme

$$
\begin{array}{c|cccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 \\
\frac{83}{250} & \frac{13861}{62500} & \frac{6889}{62500} & 0 & 0 & 0 & 0 \\
\frac{31}{50} & -\frac{116923316275}{23936840614681} & -\frac{2731218467317}{15368042101831} & \frac{9408046702089}{11113171139209} & 0 & 0 & 0 \\
\frac{17}{20} & -\frac{451086348788}{29024286899091} & -\frac{2682348792572}{7519795681897} & \frac{12662868775082}{11960479115383} & \frac{3355817975965}{11060851509271} & 0 & 0 \\
1 & \frac{647845179188}{3216320057751} & \frac{73281519250}{8382639484533} & \frac{552539513391}{3454668386233} & \frac{3354512671639}{8306763924573} & \frac{4040}{17871} & 0 \\
\hline
b_i & \frac{82889}{524892} & 0 & \frac{15625}{83664} & \frac{69875}{102672} & -\frac{2260}{8211} & \frac{1}{4} \\
\hline
\hat{b}_i & \frac{4586570599}{29645900160} & 0 & \frac{178811875}{945068544} & \frac{814220225}{1159782912} & -\frac{3700637}{11593932} & \frac{61727}{225920}
\end{array}
$$

Fourth-Order Singly Diagonally Implicit Runge-Kutta (ESDIRK) Scheme

$$
\begin{array}{c|cccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{2} & \frac{1}{4} & \frac{1}{4} & 0 & 0 & 0 & 0 \\
\frac{83}{250} & \frac{8611}{62500} & -\frac{1743}{31250} & \frac{1}{4} & 0 & 0 & 0 \\
\frac{31}{50} & \frac{5012029}{34652500} & -\frac{654441}{2922500} & \frac{174375}{388108} & \frac{1}{4} & 0 & 0 \\
\frac{17}{20} & \frac{15267082809}{155376265600} & -\frac{71443401}{120774400} & \frac{730878875}{902184768} & \frac{2285395}{8070912} & \frac{1}{4} & 0 \\
1 & \frac{82889}{524892} & 0 & \frac{15625}{83664} & \frac{69875}{102672} & -\frac{2260}{8211} & \frac{1}{4} \\
\hline
b_i & \frac{82889}{524892} & 0 & \frac{15625}{83664} & \frac{69875}{102672} & -\frac{2260}{8211} & \frac{1}{4} \\
\hline
\hat{b}_i & \frac{4586570599}{29645900160} & 0 & \frac{178811875}{945068544} & \frac{814220225}{1159782912} & -\frac{3700637}{11593932} & \frac{61727}{225920}
\end{array}
$$

Summation-by-parts Operators

The SBP operators used in this work are presented here. They are obtained from Mattsson,[16] where the interested reader may find more detailed information on their formulation.

SBP 1-2-1 Scheme

The following SBP operators are first order accurate at the boundary and second order accurate in the interior. The discrete norm $\mathcal{P}$ and the discrete SBP operator $\mathcal{D}_1$ approximating $\frac{\mathrm{d}}{\mathrm{d}x}$ are given by

$$\mathcal{P} = \mathrm{d}x \begin{pmatrix} \frac{1}{2} &&&& \\ & 1 &&& \\ && \ddots && \\ &&& 1 & \\ &&&& \frac{1}{2} \end{pmatrix}, \quad \mathcal{D}_1 = \mathcal{P}^{-1}\mathcal{Q} = \frac{1}{\mathrm{d}x} \begin{pmatrix} -1 & 1 &&& \\ -\frac{1}{2} & 0 & \frac{1}{2} && \\ & \ddots & \ddots & \ddots & \\ && -\frac{1}{2} & 0 & \frac{1}{2} \\ &&& -1 & 1 \end{pmatrix}.$$

The discrete SBP operator $\mathcal{D}_2 = \mathcal{P}^{-1}(-\mathcal{M} + \mathcal{BD})$ approximating $\frac{\mathrm{d}^2}{\mathrm{d}x^2}$ and the boundary derivative operator $\mathcal{BS}$ are given by

$$\mathcal{D}_2 = \frac{1}{\mathrm{d}x^2} \begin{pmatrix} 1 & -2 & 1 && \\ 1 & -2 & 1 && \\ & \ddots & \ddots & \ddots & \\ && 1 & -2 & 1 \\ && 1 & -2 & 1 \end{pmatrix}, \quad \mathcal{BD} = \frac{1}{\mathrm{d}x} \begin{pmatrix} \frac{3}{2} & -2 & \frac{1}{2} && \\ & 1 &&& \\ && \ddots && \\ &&& 1 & \\ && \frac{1}{2} & -2 & \frac{3}{2} \end{pmatrix}.$$

SBP 2-4-2 Scheme

The following SBP operators are second order accurate at the boundary and fourth order accurate in the interior. The discrete norm $\mathcal{P}$ is given by

$$\mathcal{P} = \mathrm{d}x \begin{pmatrix} \frac{17}{48} &&&&& \\ & \frac{59}{48} &&&& \\ && \frac{43}{48} &&& \\ &&& \frac{49}{48} && \\ &&&& 1 & \\ &&&&& \ddots \end{pmatrix}.$$

The discrete SBP operator $\mathcal{D}_1$ approximating $\frac{\mathrm{d}}{\mathrm{d}x}$ is given by

$$\mathcal{D}_1 = \mathcal{P}^{-1}\mathcal{Q} = \frac{1}{\mathrm{d}x} \begin{pmatrix} -\frac{24}{17} & \frac{59}{34} & -\frac{4}{17} & -\frac{3}{34} & 0 & 0 & 0 & \\ -\frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & \\ \frac{4}{43} & -\frac{59}{86} & 0 & \frac{59}{86} & -\frac{4}{43} & 0 & 0 & \\ \frac{3}{98} & 0 & -\frac{59}{98} & 0 & \frac{32}{49} & -\frac{4}{49} & 0 & \\ 0 & 0 & \frac{1}{12} & -\frac{2}{3} & 0 & \frac{2}{3} & -\frac{1}{12} & \\ &&& \ddots & \ddots & \ddots & \ddots & \ddots \end{pmatrix}.$$

The discrete SBP operator $\mathcal{D}_2$ approximating $\frac{\mathrm{d}^2}{\mathrm{d}x^2}$ is given by

$$\mathcal{D}_2 = \mathcal{P}^{-1}(-\mathcal{M} + \mathcal{BD}) = \frac{1}{\mathrm{d}x^2} \begin{pmatrix} 2 & -5 & 4 & -1 & 0 & 0 & 0 & \\ 1 & -2 & 1 & 0 & 0 & 0 & 0 & \\ -\frac{4}{43} & \frac{59}{43} & -\frac{110}{43} & \frac{59}{43} & -\frac{4}{43} & 0 & 0 & \\ -\frac{1}{49} & 0 & \frac{59}{49} & -\frac{118}{49} & \frac{64}{49} & -\frac{4}{49} & 0 & \\ 0 & 0 & -\frac{1}{12} & \frac{4}{3} & -\frac{5}{2} & \frac{4}{3} & \frac{1}{12} & \\ &&& \ddots & \ddots & \ddots & \ddots & \ddots \end{pmatrix},$$

and the boundary derivative operator $\mathcal{BD}$ is given by,

$$\mathcal{BD} = \frac{1}{\mathrm{d}x}\begin{pmatrix} \frac{11}{6} & -3 & \frac{3}{2} & -\frac{1}{3} & & & \\ & 0 & & & & & \\ & & & \ddots & & & \\ & & & & & 0 & \\ & & & -\frac{1}{3} & \frac{3}{2} & -3 & \frac{11}{6} \end{pmatrix}.$$

Design Order Tables

The L_2 and L_∞ error norms and convergence rates for Burger's equation and the two-dimensional advection equation are shown here.

Burger's Equation

Table 5. Burger's Equation: second-order spatial accuracy with no overlaps.

OP IMEX SBP 1-2-1, No overlaps

N	L_2 Error	L_2 Rate	L_∞ Error	L_∞ Rate
33	4.79E-002	-	2.18E-001	-
63	1.34E-002	1.972369493	7.83E-002	1.580341848
126	3.27E-003	2.032683149	2.06E-002	1.929890762
258	7.72E-004	2.015284878	5.12E-003	1.941126999
513	1.94E-004	2.006000201	1.29E-003	2.003397616

OP IMEX SBP 1-2-1, One overlap

N	L_2 Error	L_2 Rate	L_∞ Error	L_∞ Rate
33	4.79E-002	-	2.18E-001	-
63	1.34E-002	1.972369493	7.83E-002	1.580341848
126	3.27E-003	2.032683149	2.06E-002	1.929890762
258	7.72E-004	2.015284878	5.12E-003	1.941126999
513	1.94E-004	2.006000202	1.29E-003	2.003397617

OP IMEX SBP 2-4-2, No overlaps

N	L_2 Error	L_2 Rate	L_∞ Error	L_∞ Rate
33	2.01E-002	-	8.00E-002	-
63	2.45E-003	3.255233370	1.38E-002	2.713134419
126	1.84E-004	3.735394185	1.22E-003	3.502319399
258	1.10E-005	3.924161809	8.07E-005	3.791414429
513	7.14E-007	3.984038129	5.23E-006	3.982542337

OP IMEX SBP 2-4-2, One overlap

N	L_2 Error	L_2 Rate	L_∞ Error	L_∞ Rate
33	2.01E-002	-	8.00E-002	-
63	2.05E-003	3.530096975	9.26E-003	3.334727521
126	1.84E-004	3.478977829	1.22E-003	2.922444018
258	1.10E-005	3.924161797	8.07E-005	3.79141443
513	7.14E-007	3.984033947	5.23E-006	3.982542623

Table 6. Burger's Equation: fourth-order temporal accuracy.

OP IMEX, No overlaps

dt	L_2 Error	L_2 Rate
0.18	5.50E-006	-
0.09	3.76E-007	3.871331123
0.045	2.42E-008	3.954481746
0.0225	1.59E-009	3.933310419
0.01125	1.37E-010	3.535462943
0.005625	1.84E-011	2.892710861

OP IMEX, One overlap

dt	L_2 Error	L_2 Rate
0.18	5.48E-006	-
0.09	3.75E-007	3.869031493
0.045	2.48E-008	3.916422375
0.0225	2.07E-009	3.584320001
0.01125	2.80E-010	2.88875141
0.005625	3.90E-011	2.841845516

Table 7. 2D Advection Equation: second-order spatial accuracy with no overlaps.

OP IMEX SBP 1-2-1, No overlaps

N_x=N_y	L_2 Error	L_2 Rate	L_∞ Error	L_∞ Rate
15	2.17E-003	-	5.88E-003	-
30	4.94E-004	2.134640187	1.54E-003	1.930163796
60	1.13E-004	2.127598038	3.60E-004	2.098610437
120	2.72E-005	2.054168695	8.92E-005	2.01334724
240	6.68E-006	2.028252476	2.22E-005	2.007627234

OP IMEX SBP 1-2-1, One overlap

N_x=N_y	L_2 Error	L_2 Rate	L_∞ Error	L_∞ Rate
15	2.17E-003	-	5.88E-003	-
30	4.94E-004	2.134641331	1.54E-003	1.928978607
60	1.13E-004	2.127807799	3.60E-004	2.100561594
120	2.72E-005	2.054103722	8.92E-005	2.013569205

OP IMEX SBP 2-4-2, No overlaps

N_x=N_y	L_2 Error	L_2 Rate	L_∞ Error	L_∞ Rate
21	3.12E-004	-	1.08E-003	-
42	3.77E-005	3.051580362	1.53E-004	2.821834935
84	3.85E-006	3.289190695	1.98E-005	2.950534388
168	4.49E-007	3.100397842	2.31E-006	3.10250198

OP IMEX SBP 2-4-2, One overlap

N_x=N_y	L_2 Error	L_2 Rate	L_∞ Error	L_∞ Rate
21	3.12E-004	-	1.08E-003	-
42	3.77E-005	3.05156826	1.53E-004	2.821815868
84	3.85E-006	3.289191155	1.98E-005	2.95049937
168	4.49E-007	3.100216603	2.31E-006	3.103500644

Table 8. 2D Advection Equation: fourth-order temporal accuracy.

OP IMEX, No overlaps

dt	L_2 Error	L_2 Rate
0.200000	6.63E-003	-
0.100000	8.17E-004	3.021287254
0.050000	8.07E-005	3.340077632
0.025000	6.40E-006	3.656917852
0.012500	4.55E-007	3.812723838
0.006250	3.05E-008	3.898016031
0.003125	1.98E-009	3.947159646

OP IMEX, Two overlaps

dt	L_2 Error	L_2 Rate
0.200000	3.36E-003	-
0.100000	4.90E-004	2.776292772
0.050000	5.42E-005	3.177890808
0.025000	4.59E-006	3.560298861
0.012500	3.35E-007	3.775275458
0.006250	2.26E-008	3.888408113
0.003125	1.47E-009	3.945146214

References

[1]Kanevsky, A., Carpenter, M. H., Gottlieb, D., and Hesthaven, J. S., "Application of implicit–explicit high order Runge–Kutta methods to discontinuous-Galerkin schemes," *Journal of Computational Physics*, Vol. 225, No. 2, 2007, pp. 1753–1781.

[2]Chapra, S. C. and Canale, R. P., *Numerical Methods for Engineers*, McGraw-Hill, 1221 Avenue of the Americas, New York, NY 10020, sixth ed., 2010.

[3]Osher, S. and Sanders, R., "Numerical Approximations to Nonlinear Conservation Laws with Locally Varying Time and Space Grids," *Mathematics of Computation*, Vol. 41, No. 164, 1983, pp. pp. 321–336.

[4]Berger, M. J. and Oliger, J., "Adaptive mesh refinement for hyperbolic partial differential equations," *Journal of Computational Physics*, Vol. 53, No. 3, 1984, pp. 484 – 512.

[5]Flaherty, J., Loy, R., Shephard, M., Szymanski, B., Teresco, J., and Ziantz, L., "Adaptive Local Refinement with Octree Load Balancing for the Parallel Solution of Three-Dimensional Conservation Laws," *Journal of Parallel and Distributed Computing*, Vol. 47, No. 2, 1997, pp. 139 – 152.

[6]Dawson, C. and Kirby, R., "High Resolution Schemes for Conservation Laws with Locally Varying Time Steps," *SIAM Journal on Scientific Computing*, Vol. 22, No. 6, 2001, pp. 2256 – 2281.

[7]Tan, Z., Zhang, Z., Huang, Y., and Tang, T., "Moving mesh methods with locally varying time steps," *Journal of Computational Physics*, Vol. 200, No. 1, 2004, pp. 347 – 367.

[8]Bernacki, M., Fezoui, L., Lanteri, S., and Piperno, S., "Parallel discontinuous Galerkin unstructured mesh solvers for the calculation of three-dimensional wave propagation problems," *Applied Mathematical Modelling*, Vol. 30, No. 8, 2006, pp. 744 – 763.

[9]Ascher, U., Ruuth, S., and Wetton, B., "Implicit-Explicit Methods for Time-Dependent Partial Differential Equations," *SIAM Journal on Numerical Analysis*, Vol. 32, No. 3, 1995, pp. 797–823.

[10]Ascher, U. M., Ruuth, S. J., and Spiteri, R. J., "Implicit-explicit Runge-Kutta methods for time-dependent partial differential equations," *Applied Numerical Mathematics*, Vol. 25, No. 23, 1997, pp. 151 – 167.

[11]Calvo, M., de Frutos, J., and Novo, J., "Linearly implicit RungeKutta methods for advectionreactiondiffusion equations," *Applied Numerical Mathematics*, Vol. 37, No. 4, 2001, pp. 535 – 549.

[12]Fritzen, P. and Wittekindt, J., "Numerical solution of viscoplastic constitutive equations with internal state variables Part I: algorithms and implementation," *Mathematical Methods in the Applied Sciences*, Vol. 20, No. 16, 1997, pp. 1411–1425.

[13]Zhong, X., "New high-order semi-implicit Runge-Kutta schemes for computing transient nonequilibrium hypersonic flows," *AIAA Paper 95-2007*, edited by T. Conference, Vol. 30th, San Diego, CA, June 19-22, 1995.

[14]Yoh, J. J.-I. and Zhong, X., "Semi-implicit Runge-Kutta schemes for stiff multi-dimensional reacting flows," *AIAA Paper 97-0803*, edited by A. S. Meeting and Exhibit, Vol. 35th, Reno, NV, January 6-9, 1997.

[15]Kennedy, C. A. and Carpenter, M. H., "Additive RungeKutta schemes for convectiondiffusionreaction equations," *Applied Numerical Mathematics*, Vol. 44, No. 1-2, Jan. 2003, pp. 139–181.

[16]Mattsson, K. and Nordström, J., "Summation by parts operators for finite difference approximations of second derivatives," *Journal of Computational Physics*, Vol. 199, No. 2, 2004, pp. 503–540.

[17]Carpenter, M. H., Gottlieb, D., and Abarbanel, S., "Time-Stable Boundary Conditions for Finite-Difference Schemes Solving Hyperbolic Systems: Methodology and Application to High-Order Compact Schemes," *Journal of Computational Physics*, Vol. 111, No. 2, 1994, pp. 220 – 236.

[18]Kreyszig, E., *Advanced Engineering Mathematics*, John Wiley and Sons Inc., 111 River Street, Hoboken, New Jersey 07030-5774, 2011.

[19]Fisher, T. C., *High-Order L2 Stable Multi-Domain Finite Difference Method for Compressible Flows*, thesis, Purdue University, July 2012.

[20]Butcher, J. C., *Numerical Methods for Ordinary Differential Equations*, Wiley, Chichester, England, 2nd ed., 2003.

[21]Fisher, T. C., Carpenter, M. H., Nordström, J., Yamaleev, N., and Swanson, C. R., "Discretely Conservative Finite-Difference Formulations for Nonlinear Conservation Laws in Split Form: Theory and Boundary Conditions," Technical report tm 2011-217307, NASA, November 2011.

[22]Saad, Y., "SPARSKIT: A basic tool-kit for sparse matrix computations (Version 2)," June 2012.

[23]Gustafsson, K., "Control theoretic techniques for stepsize selection in explicit Runge-Kutta methods," *ACM Trans. Math. Softw.*, Vol. 17, No. 4, Dec. 1991, pp. 533–554.

[24]Gustafsson, K., "Control-theoretic techniques for stepsize selection in implicit Runge-Kutta methods," *ACM Trans. Math. Softw.*, Vol. 20, No. 4, Dec. 1994, pp. 496–517.

[25]Sderlind, G., "Automatic Control and Adaptive Time-Stepping," *Numerical Algorithms*, Vol. 31, No. 1-4, 2002, pp. 281–310.

[26]Balay, S., Brown, J., Buschelman, K., Gropp, W. D., Kaushik, D., Knepley, M. G., McInnes, L. C., Smith, B. F., and Zhang, H., "PETSc Web page," 2012, http://www.mcs.anl.gov/petsc.