High-Order LES Simulations using Implicit-Explicit Runge-Kutta Schemes

Per-Olof Persson*

University of California, Berkeley, Berkeley, CA 94720-3840, U.S.A.

For many flow problems modeled by Large Eddy Simulation (LES), the computational meshes are such that a large number of the elements would allow for explicit timestepping, but the CFL-condition is highly limited by the smaller stretched elements in the boundary layers. We propose an implicit-explicit time-integration scheme that uses an implicit solver only for the smaller portion of the domain that requires it to avoid severe timestep restrictions, but an efficient explicit solver for the rest of the domain. We use the Runge-Kutta IMEX schemes and consider several schemes of varying number of stages, orders of accuracy, and stability properties, and study the stability and the accuracy of the solver. We also show the application of the technique on a realistic LES-type problem of turbulent flow around an airfoil, where we conclude that the approach can give performance that is superior to both fully explicit and fully implicit methods.

I. Introduction

Time-accurate integration of high-order discretizations of the Navier-Stokes equations poses several difficulties. The CFL conditions imposed by explicit schemes usually dictate timesteps that are several magnitudes too small to be practical. On the other hand, high-order discretizations often produce very large Jacobian matrices, which makes fully implicit schemes prohibitively expensive to use.

There are several sources of stiffness that cause these severe timestep restrictions for explicit schemes, such as the acoustic waves for low-Mach number flows, viscous effects, and large variations in the mesh sizes. For problems involving turbulent flows modeled by Large Eddy Simulation (LES), the most critical reason for employing implicit methods is often the small and stretched mesh elements that are required to resolve the thin boundary layers. For these simulations, the stiffness is caused by the equations corresponding to the degrees of freedom in the stretched region, and the mesh elements away from the bodies might be better handled by explicit time integration. This effect will likely be even more important on the future generation of multi-core and GPU computer architectures, which appear to favor local explicit methods over the more memory-intensive implicit ones.[1]

In an attempt to take advantage of the fact that the problems are nonstiff in most of the computational domain, we propose using the so-called IMEX schemes[2,3] to obtain a combination of the best properties of the implicit and the explicit solvers. These methods are based on a splitting of the residual vector into a stiff and a nonstiff part, and an additive Runge-Kutta method creates a combined method that can be made high-order accurate in time. Many of the original applications of the IMEX schemes used splittings of the actual equations (for example into nonstiff advective terms and stiff diffusive terms), but here we use a splitting based on the size of the elements in the mesh, similar to the geometry-induced stiffness considered in ref. 4. The resulting scheme can be highly efficient, and the Jacobians that have to be computed and used for solving nonlinear equations might only be a fraction of the size of the fully implicit ones. In addition, we re-use both the computed Jacobian matrices as well as the incomplete factorizations, which brings down the cost of the implicit solvers further.

To discretize the compressible Navier-Stokes equations, we use a high-order discontinuous Galerkin method on unstructured meshes of triangles and tetrahedra.[5,6] The implicit parts of the problem are treated with a Newton-Krylov solver with block incomplete-LU preconditioning and approximately optimal

*Assistant Professor, Department of Mathematics, University of California, Berkeley, Berkeley CA 94720-3840. E-mail: persson@berkeley.edu. AIAA Member.

element ordering.[7] We present the schemes, use simple model problems to determine the stability and the accuracy of the schemes, and finally show the application of the method to a realistic LES problem of flow around an airfoil at a Reynolds number of 100,000.

II. Governing Equations and Space Discretization

II.A. The Compressible Navier-Stokes Equations

The compressible Navier-Stokes equations are written as:

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_i}(\rho u_i) = 0, \tag{1}$$

$$\frac{\partial}{\partial t}(\rho u_i) + \frac{\partial}{\partial x_i}(\rho u_i u_j + p) = +\frac{\partial \tau_{ij}}{\partial x_j} \quad \text{for } i = 1,2,3, \tag{2}$$

$$\frac{\partial}{\partial t}(\rho E) + \frac{\partial}{\partial x_i}\left(u_j(\rho E + p)\right) = -\frac{\partial q_j}{\partial x_j} + \frac{\partial}{\partial x_j}(u_j \tau_{ij}), \tag{3}$$

where ρ is the fluid density, u_1, u_2, u_3 are the velocity components, and E is the total energy. The viscous stress tensor and heat flux are given by

$$\tau_{ij} = \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{2}{3}\frac{\partial u_k}{\partial x_j}\delta_{ij}\right) \qquad \text{and} \qquad q_j = -\frac{\mu}{\mathrm{Pr}}\frac{\partial}{\partial x_j}\left(E + \frac{p}{\rho} - \frac{1}{2}u_k u_k\right). \tag{4}$$

Here, μ is the viscosity coefficient and $\mathrm{Pr} = 0.72$ is the Prandtl number which we assume to be constant. For an ideal gas, the pressure p has the form

$$p = (\gamma - 1)\rho\left(E - \frac{1}{2}u_k u_j\right), \tag{5}$$

where γ is the adiabatic gas constant.

For the modeling of the turbulent flows we use Implicit Large Eddy Simulation (ILES). In LES modeling, the large scale flow features are resolved while the small scales are modeled. The rationale behind this is that the small scales are isotropic, carry less of the flow energy and therefore do not have as much influence on the mean flow, and can therefore be approximated or modeled. The effect of these subgrid scales (SGS) is approximated by an eddy viscosity which can be derived from a so-called SGS model or can be taken to be equal to the dissipation in the numerical scheme, which is the principle behind the ILES model.[8] Simulations based on ILES models often give very accurate predictions but are limited to low Reynolds number flows because of the high computational cost of resolving the large scale features of the flow.

II.A.1. High-Order DG Spatial Discretization

Our 3DG flow solver is based on the high-order Discontinuous Galerkin (DG) method with tetrahedral mesh elements and nodal basis functions. We write the governing equations (1)-(3) in a split form as

$$\frac{\partial u}{\partial t} + \nabla \cdot F^{\mathrm{i}}(u) - \nabla \cdot F^{\mathrm{v}}(u, q) = 0 \tag{6}$$

$$q = \nabla u. \tag{7}$$

The equations (6)-(7) are discretized using a discontinuous Galerkin method[5] with the Compact Discontinuous Galerkin (CDG) method[9] for the viscous terms. The spatial domain Ω is discretized into a triangulation $\mathcal{T}_h$, and we seek solutions in the finite element spaces

$$V_h = \{v \in [L^2(\Omega)]^5 \mid v|_K \in [\mathcal{P}_p(K)]^5, \;\; \forall K \in \mathcal{T}_h\}\,, \tag{8}$$

$$\Sigma_h = \{r \in [L^2(\Omega)]^{5\times 3} \mid r|_K \in [\mathcal{P}_p(K)]^{5\times 3}, \;\; \forall K \in \mathcal{T}_h\}\,, \tag{9}$$

where $\mathcal{P}_p(K)$ is the space of polynomial functions of degree at most $p > 0$. Our DG formulation becomes: find $u_h \in V_h$ and $q_h \in \Sigma_h$ such that for all $K \in \mathcal{T}_h$, we have

$$\int_K \frac{\partial u_h}{\partial t} v \, dx - \int_K \left(F^{\mathrm{i}}(u_h) - F^{\mathrm{v}}(u_h, q_h) \right) \cdot \nabla v \, dx + \int_{\partial K} \left(\hat{F}^{\mathrm{i}} - \hat{F}^{\mathrm{v}} \right) v \, ds = 0, \quad \forall v \in [\mathcal{P}_p(K)]^5 \ , \tag{10}$$

$$\int_K q_h \cdot r \, dx = - \int_K u_h \nabla \cdot r \, dx + \int_{\partial K} \hat{u} r \cdot n \, ds, \quad \forall r \in [\mathcal{P}_p(K)]^{5 \times 3} \ . \tag{11}$$

Here, the inviscid numerical fluxes $\hat{F}^{\mathrm{i}}$ are approximated using the method due to Roe.[10] For the viscous fluxes $\hat{F}^{\mathrm{v}}$, we use the compact discontinuous Galerkin (CDG) scheme[9] and choose $\hat{u}$ and $\hat{F}^{\mathrm{v}}$ according to

$$(\hat{F}^{\mathrm{v}})^e = \{\{ F^{\mathrm{v}}(u_h, q_h^e) \cdot \boldsymbol{n} \}\} + C_{11} [\![u_h \boldsymbol{n}]\!] + \boldsymbol{C}_{12} [\![F^{\mathrm{v}}(u_h, q_h^e) \cdot \boldsymbol{n}]\!] \tag{12}$$

$$\hat{u} = \{\{ u_h \}\} - \boldsymbol{C}_{12} \cdot [\![u_h \boldsymbol{n}]\!] \ . \tag{13}$$

Here, $\{\{ \ \}\}$ and $[\![\ ]\!]$ denote the average and jump operators across the interface.[9] We set $C_{11} = 0$ at most internal faces, and $C_{11} = 10/h$ at the Dirichlet boundaries for elements of height h normal to the boundary as well as on internal faces of highly stretched elements, to obtain some additional stabilization. Furthermore, we set $\boldsymbol{C}_{12} = \boldsymbol{n}^*$, where $\boldsymbol{n}^*$ is the unit normal to the interface taken with an arbitrary sign. The "edge" fluxes q_h^e are computed by solving the equation

$$\int_K q_h^e \cdot r \, dx = - \int_K u_h \nabla \cdot r \, dx + \int_{\partial K} \hat{u}^e r \cdot n \, ds, \quad \forall r \in [\mathcal{P}_p(K)]^{5 \times 3} \ , \tag{14}$$

where

$$\hat{u}_h^e = \begin{cases} \hat{u}_h & \text{on edge } e \text{, given by equation (13),} \\ u_h & \text{otherwise.} \end{cases} \tag{15}$$

The boundary conditions are imposed in terms of the fluxes $\hat{u}$ and $\hat{F}^{\mathrm{i}}, \hat{F}^{\mathrm{v}}$. For more details on the scheme we refer to ref. 9.

While our implementation is fully general and capable of three dimensional simulations of arbitrary orders, we limit ourselves here to two space dimensions and polynomial degrees $p = 4$. The discrete finite element spaces V_h, Σ_h are represented by nodal Lagrange basis functions within each tetrahedral element. All volume and face integrals are computed by specialized Gaussian integration rules for simplex elements, up to an order of precision $3p$. We note that due to the choice of $\hat{u}$ in the CDG scheme (13), the variables q_h can be explicitly solved for in (11) and therefore eliminated from (10). This results in a discretized scheme that only depends explicitly on u_h, or the solution vector U, in the form of a nonlinear system of ODEs:

$$M \frac{dU}{dt} = R(U) \tag{16}$$

with mass matrix M and residual vector $R(U)$. This system is integrated in time using explicit or implicit schemes, or as in this work, a combination of both.

III. Implicit-Explicit Runge-Kutta Methods

The IMEX schemes are based on a splitting of the residual vector in a system of ODEs of the form

$$\frac{du}{dt} = f(u) + g(u) \tag{17}$$

where $f(u)$ is considered nonstiff terms and $g(u)$ stiff terms. The schemes are of Runge-Kutta type, with one scheme c, A, b for the implicit treatment of $g(u)$ and another scheme $\hat{c}, \hat{A}, \hat{b}$ for the explicit treatment of $f(u)$. These are standard Runge-Kutta schemes by themselves, of the types Diagonally Implicit Runge-Kutta (DIRK)[11] and Explicit Runge-Kutta (ERK). However, the schemes are also designed in such a way that they can be combined for integration of ODEs of the split form (17).

To integrate from step n to step $n+1$ using the timestep Δt, the first stage is always explicit, and the remaining s stages are done pairwise implicit/explicit. The solution at timestep $n+1$ is then a linear combination of the stage derivatives of both schemes, and the method can be written as:

$\hat{k}_1 = f(u_n)$
for $i = 1$ **to** s

Solve for k_i in $k_i = g(u_{n,i})$, where $u_{n,i} = u_n + \Delta t \sum_{j=1}^{i} a_{i,j} k_j + \Delta t \sum_{j=1}^{i} \hat{a}_{i+1,j} \hat{k}_j$

Evaluate $\hat{k}_{i+1} = f(u_{n,i})$

end for

$u_{n+1} = u_n + \Delta t \sum_{i=1}^{s} b_j k_j + \Delta t \sum_{i=1}^{s+1} \hat{b}_j \hat{k}_j$

A number of IMEX schemes of this form have been developed, with various orders of accuracy and stability properties.[2] Here we consider three typical schemes:

IMEX1: 2nd order accurate: 2-stage, 2nd order DIRK + 3-stage, 2nd order ERK

$$\begin{array}{c|c} c & A \\ \hline & b^T \end{array} = \begin{array}{c|cc} \alpha & \alpha & 0 \\ 1 & 1-\alpha & \alpha \\ \hline & 1-\alpha & \alpha \end{array} \qquad \begin{array}{c|c} \hat{c} & \hat{A} \\ \hline & \hat{b}^T \end{array} = \begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ \alpha & \alpha & 0 & 0 \\ 1 & \delta & 1-\delta & 0 \\ \hline & 0 & 1-\alpha & \alpha \end{array}$$

with $\alpha = 1 - \frac{\sqrt{2}}{2}$ and $\delta = -2\sqrt{2}/3$. This DIRK scheme is stiffly accurate, and while the ERK is only second order accurate it has the same stability region as a third order ERK which is appropriate for problems with eigenvalues close to the imaginary axis.

IMEX2: 3rd order accurate: 2-stage, 3rd order DIRK + 3-stage, 3rd order ERK

$$\begin{array}{c|c} c & A \\ \hline & b^T \end{array} = \begin{array}{c|cc} \alpha & \alpha & 0 \\ 1-\alpha & 1-2\alpha & \alpha \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array} \qquad \begin{array}{c|c} \hat{c} & \hat{A} \\ \hline & \hat{b}^T \end{array} = \begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ \alpha & \alpha & 0 & 0 \\ 1-\alpha & \alpha-1 & 2(1-\alpha) & 0 \\ \hline & 0 & \frac{1}{2} & \frac{1}{2} \end{array}$$

with $\alpha = (3+\sqrt{3})/6$. The resulting scheme is third order accurate with the same number of stages as the previous scheme, at the cost of losing the L-stability of the DIRK scheme.

IMEX3: 3rd order accurate: 3-stage, 3rd order DIRK + 4-stage, 3rd order ERK

$$\begin{array}{c|c} c & A \\ \hline & b^T \end{array} = \begin{array}{c|ccc} 0.4358665215 & 0.4358665215 & 0 & 0 \\ 0.7179332608 & 0.2820667392 & 0.4358665215 & 0 \\ 1 & 1.208496649 & -0.644363171 & 0.4358665215 \\ \hline & 1.208496649 & -0.644363171 & 0.4358665215 \end{array}$$

$$\begin{array}{c|c} \hat{c} & \hat{A} \\ \hline & \hat{b}^T \end{array} = \begin{array}{c|cccc} 0 & 0 & 0 & 0 & 0 \\ 0.4358665215 & 0.4358665215 & 0 & 0 & 0 \\ 0.7179332608 & 0.3212788860 & 0.3966543747 & 0 & 0 \\ 1 & -0.105858296 & 0.5529291479 & 0.5529291479 & 0 \\ \hline & 0 & 1.208496649 & -0.644363171 & 0.4358665215 \end{array}$$

This 3-stage DIRK scheme is L-stable and third order accurate, and the 4-stage ERK scheme is third order accurate but has the larger stability region of a fourth order ERK.

III.A. Mesh-Size Based Splitting of Residual

For our system of ODEs (16), we associate each component of the solution vector U with an equation in the residual $R(U)$. Our splitting is based on identifying stiff components U_{im} that are located in mesh elements smaller than a given size, and the remaining nonstiff components U_{ex}. This produces a splitting of the residual vector as

$$R(u) = \begin{bmatrix} R_{\text{im}}(U) \\ R_{\text{ex}}(U) \end{bmatrix} = \begin{bmatrix} 0 \\ R_{\text{ex}}(U) \end{bmatrix} + \begin{bmatrix} R_{\text{im}}(U) \\ 0 \end{bmatrix} = f(U) + g(U) \tag{18}$$

The idea behind this splitting is that the stiffness from the implicit equations should not affect the explicit ones much, and it should be possible to use a timestep limited by the equations in the nonstiff region only. But note that depending on the equations and on the splitting this might not be the case, since the two schemes are coupled between each integration stage and it is unclear how this affects the stability properties of the full scheme. However, as we show in our numerical results below, it is often the case that the IMEX schemes can produce stable results with the larger timestep dictated by the element sizes in the explicit region only.

III.B. Quasi-Newton and Preconditioned Krylov Methods

For the implicit part of the IMEX scheme, nonlinear systems of equations $k_i = g(u_{n,i})$ must be solved. For this, we use a quasi-Newton method with a Jacobian matrix $J = M - \alpha \Delta t dR/dU$, where Δt is the timestep and α is a parameter of order one. This matrix is computed and stored explicitly but re-used between the iterations as well as between the timesteps. This turns out to work extremely well for these types of computations, and with the exception of the first initial transients we essentially never have to recompute the Jacobian matrix.

To solve the linear systems of equations involving J, we use a preconditioned Newton-Krylov technique, consisting of an ILU-preconditioned CGS solver with element ordering by the Minimum Discarded Fill (MDF) algorithm.[7] Since we re-use the Jacobian matrix many times, we can also re-use the incomplete factorization and the total implicit solution time is dominated by the matrix-vector products and the backsolves in the CGS method.

We would also like to point out that since the Jacobian is re-used, one possibility is to employ a direct linear solver and re-use the entire factorization. While normally not considered competitive for three dimensional problems, our implicit problems only involve layers of elements close to the boundary and might therefore behave more like planar problems, which are well-known to scale better in terms of both memory and computational cost.[12]

IV. Results

We demonstrate our method on three test problems. First, we study the stability of the schemes using a model problem of flow over a flat plate in a rectangular domain. Next, we use an Euler vortex model problem to determine the orders of accuracy. Finally, we apply the technique on a more realistic simulation of turbulent flow over an airfoil at a high angle of attack. All simulations are done using our software package 3DG,[13] which is a general-purpose toolkit for discretization of arbitrary systems of conservation laws.[9] It produces fully analytical Jacobian matrices, and it includes efficient parallel Newton-Krylov solvers.[7,14] Due to the highly modular and general design of the 3DG software, it was straight-forward to incorporate the implicit-explicit capabilities and it required only a few dozen lines of additional code.

IV.A. Absolute Stability – Flow Over a Flat Plate

Our first problem is a simple flat plate model problem that we use to to identify the feasibility of the approach, and in particular to determine if the CFL condition for the explicit portion of the domain is affected by the implicit portion. The domain is a square of unit length, with free-stream boundary conditions at left/top, no-slip wall conditions at bottom, and an outflow condition at the right boundary. We set the Mach number to 0.2 and the Reynolds number to 10,000 based on the domain width. A mesh and a steady-state solution is shown in Fig. 1.

A series of meshes of increasing anisotropy is generated in the following way: The initial mesh has 10-by-10 uniformly sized squares (of size 0.1-by-0.1). The bottom row is then split horizontally, and this process is repeated n_{ref} times to generate an anisotropic boundary layer mesh. Finally, we split each quadrilateral into two triangles, since our code is based on simplex elements. We note that the smallest element height is $h_{\min} = 0.1/2^{n_{\text{ref}}}$, and the highest element aspect ratio is $2^{n_{\text{ref}}}$.

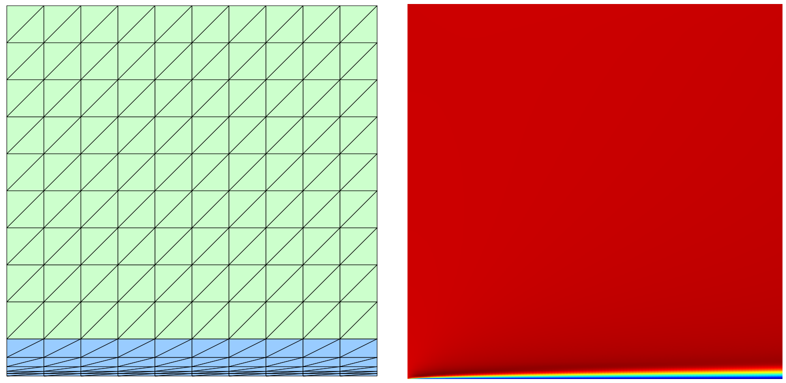


Figure 1. The flow over a flat plate model problem with a thin boundary layer. A typical computational mesh (left) and the steady-state solution (right).

Scheme	ERK1		IMEX1	ERK2		IMEX2	ERK3		IMEX3
$\Delta t^0_{\max}$	$3.26 \cdot 10^{-4}$			$3.35 \cdot 10^{-4}$			$2.61 \cdot 10^{-4}$		
n_{ref}	$\frac{\Delta t_{\max}}{\Delta t^0_{\max}}$	Ratio		$\frac{\Delta t_{\max}}{\Delta t^0_{\max}}$	Ratio		$\frac{\Delta t_{\max}}{\Delta t^0_{\max}}$	Ratio	
0	1.0000		Stable	1.0000		Stable	1.0000		Stable
1	0.6612	1.51	Stable	0.6671	1.50	Stable	0.4958	2.02	Stable
2	0.1747	3.79	Stable	0.1762	3.79	Stable	0.1298	3.82	Stable
3	0.0457	3.82	Stable	0.0461	3.82	Stable	0.0337	3.85	Stable
4	0.0118	3.89	Stable	0.0119	3.89	Stable	0.0086	3.92	Stable
5	0.0032	3.62	Stable	0.0033	3.62	Stable	0.0023	3.72	Stable
6	0.0009	3.82	Stable	0.0009	3.85	Stable	0.0006	3.89	Stable

Table 1. The flat plate test problem, using the three ERK schemes and the three IMEX schemes. This confirms that the CFL condition for the IMEX schemes is not affected by the element sizes in the implicit boundary layer region.

For each of the three IMEX schemes, we first determine the largest stable timestep $\Delta t_{\max}$ if the problem was solved using the ERK method only. This is done in an automated way, using a bisection method applied to a function that determines stability numerically. In particular, we define the timestep on the coarse unrefined initial mesh by $\Delta t^0_{\max}$, which is also the timestep we hope to be able to use for our IMEX schemes on any of the stretched meshes.

We then run the test problem using the full IMEX scheme, where all split elements are considered implicit and the remaining (square) elements are considered explicit. To confirm that the stability of this scheme is determined by the mesh size in the explicit portion of the domain, we verify that the method is stable on any of the refined meshes using the timestep $\Delta t^0_{\max}$.

The results are presented in Tab. 1. We make the following observations:

- The timestep on the unrefined mesh $\Delta t^0_{\max}$ is almost equal for ERK1 and ERK2 but about 25% smaller for ERK3. This is unexpected since ERK3 has a larger linear stability region, but for this highly nonlinear problem it appears to be more sensitive than the other two.
- As the boundary layer is refined, the timestep $\Delta t_{\max}$ scales first linearly with $h_{\min}$ (ratio of about 2 between successive values of n_{ref}), and then quadratically (ratio of about 4). This is expected because the inviscid timestep restrictions are dominant for the under-resolved meshes, but eventually the viscous terms determine the timesteps.
- All the IMEX schemes are stable with the timestep $\Delta t^0_{\max}$, independently of the number of refinements n_{ref} and, therefore, of $h_{\min}$.

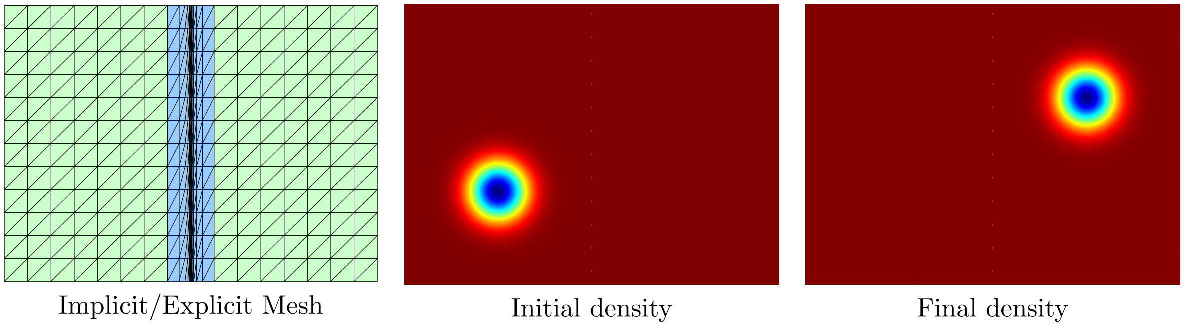


Figure 2. Unsteady Euler vortex problem, computational mesh (left) with implicit elements blue and explicit elements green, and the initial/final solutions (center and right).

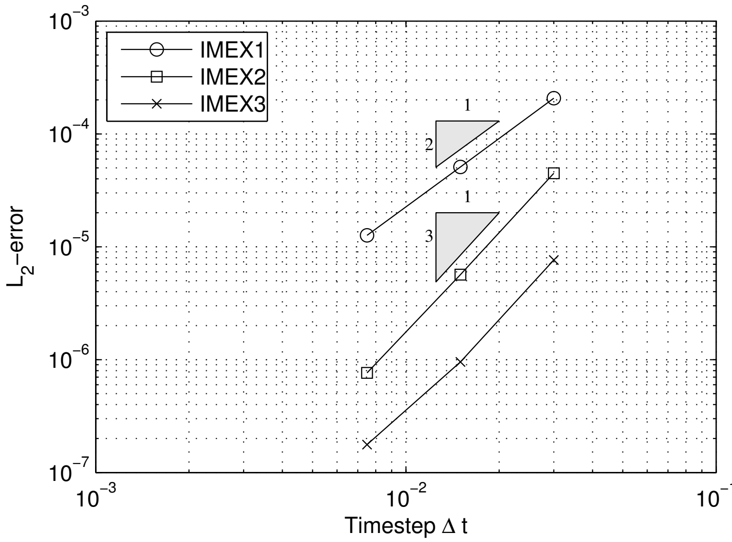


Figure 3. Temporal convergence of the three IMEX schemes for the Euler vortex problem on mesh with stretched elements. The orders of convergence 2, 3, 3, correspond to the expected orders.

IV.A.1. Order of Accuracy, Euler vortex problem

To validate the accuracy of the IMEX schemes, we solve an inviscid model problem consisting of a compressible vortex in a rectangular domain.[15] We use a domain of size 20-by-15, with the vortex initially centered at $(x_0, y_0) = (5, 5)$ with respect to the lower-left corner. The Mach number is $M_\infty = 0.5$ and the free-stream velocity angle $\theta = \arctan(1/2)$. We use periodic boundary conditions and integrate until time $t_0 = \sqrt{10^2 + 5^2}$, when the vortex has moved a relative distance of $(10, 5)$.

Our mesh is again obtained by anisotropic refinement of an initial Cartesian grid. We split vertically through the center of the rectangular domain, a total of $n_{\mathrm{ref}} = 5$ times. The mesh, the initial solution, and the final solution are shown in Fig. 2.

We compute a reference solution using the 4th order accurate RK4 scheme, using a stable timestep based on the smallest element size. The error in the IMEX solutions are then computed in the L_2-norm for the three timestep $\Delta t^0_{\max}$, $\Delta t^0_{\max}/2$, and $\Delta t^0_{\max}/4$, where again $\Delta t^0_{\max}$, is the explicit timestep limit for the unrefined mesh. The resulting convergence plot is shown in Fig. 3, and we can confirm that the order of accuracy for the three IMEX schemes are about 2, 3, and 3, respectively. Note that while this is the expected order of accuracy it was not certain that we would observe it here, since the implicit part of the problem uses large CFL numbers and therefore might not be in the convergent regime.

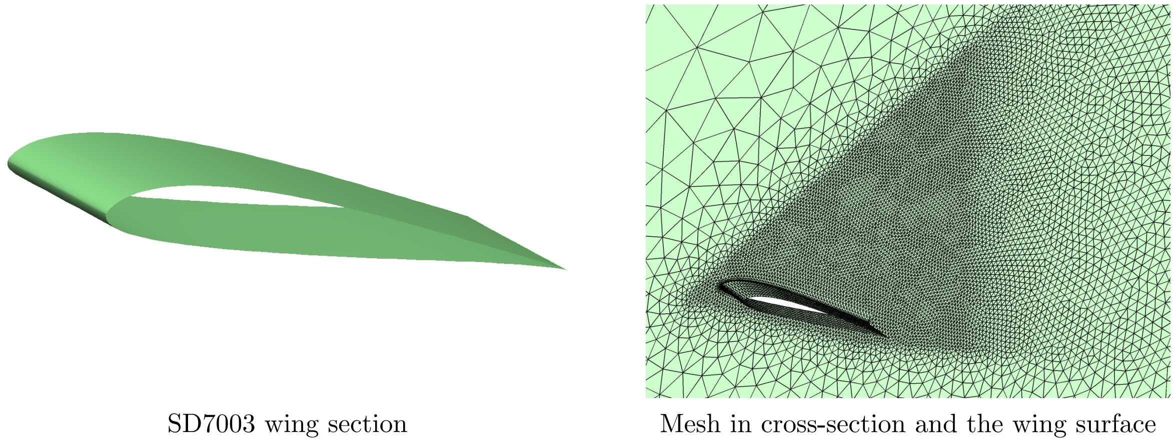


Figure 4. Geometry and the extruded hybrid mesh for the ILES problem, with 312,000 tetrahedral elements.

.

IV.B. Unsteady Large Eddy Simulations of Flow over Airfoil

As an example of a realistic problem where the IMEX schemes can make a significant difference in performance, we study the flow over an SD7003 foil at Reynolds number 100,000 and 30° angle of attack. Our mesh is a typical LES-type mesh – somewhat stretched elements for resolving the boundary layer profile, and an almost uniform mesh in the wake that captures the large scale features of the unsteady flow, see Fig. 4. It is generated using a hybrid approach, where the DistMesh mesh generator[16] is used to create an unstructured mesh for more of the domain, and the boundary points are connected to the airfoil in a structured pattern that allow for a stretching of a factor between 10–50 along the wing surface. The elements are curved to align with the boundaries using a nonlinear elasticity approach.[17] Finally, the triangular mesh is extruded in the span-wise direction to generate six layers of prismatic elements, which are each split into three tetrahedral elements. The total number of elements in the mesh is 312,000, which corresponds to about 31 million degrees of freedom for the Navier-Stokes equations and polynomial orders of $p = 3$.

We split into implicit and explicit equations element-wise based on the smallest edge sizes. In figure 5 we show the two-dimensional cross-section of the mesh and the corresponding element size distribution based on the smallest edge length, since this is what will likely dictate the CFL condition for that element (at least for well-shaped meshes). Less than 9% of the elements are considered boundary layer elements, and by excluding them from the explicit region we bring up the smallest explicit element size by about a factor of 100.

For simplicity we consider only the **IMEX1** scheme, and we obtain the following stability results:

- Using the ERK1 scheme on the explicit portion only, the largest stable timestep is about $\Delta t = 1.2 \cdot 10^{-4}$.
- Using the ERK1 scheme on the entire mesh, the largest stable timestep is about $\Delta t = 1.8 \cdot 10^{-8}$. This large ratio between the two timesteps shows that they are restricted by the viscous effects, which leads to a factor of about $100^2 = 10^4$.
- Using the IMEX1 scheme on the entire mesh with the splitting shown in figure 5, the largest stable timestep is the same as for ERK1 alone on the explicit mesh, that is, about $\Delta t = 1.2 \cdot 10^{-4}$.

This ratio of about 10,000 comes at the cost of solving non-linear systems of equations. However, these only involve 9% of the unknowns and can be solved efficiently by re-using the Jacobians. After the initial transients have decayed, our solvers uses an average of six Newton iterations per Runge-Kutta stage and the number of Krylov iterations per linear system is less than 10. In our test implementation, this leads to a total cost per stage (implicit and explicit) that is about 3 times higher than a fully explicit stage, which is a performance improvement of about a factor of 3,000.

We did not perform a comparison with a fully implicit method, but we estimate that the IMEX solver would be about a magnitude faster due to the low-cost explicit evaluations for 91% of the degrees of freedom. In addition, the fully implicit scheme would require about 10 times as much memory for storing the full Jacobians.

V. Conclusions

We have shown how to use the Runge-Kutta IMEX schemes to combine the advantages of implicit and explicit time integrators for LES-type flow problems. Two test problems were used to show that the approach produces a scheme which is both accurate and stable without the severe timestep restrictions of a fully explicit method. A realistic larger-scale three dimensional flow problem was used to show that if the meshes are such that a majority of the elements can be integrated explicitly, then the IMEX scheme can be highly efficient compared to fully explicit or implicit schemes. The smaller Jacobian matrices that arise can be re-used between many of the implicit solution steps, which saves additional computational time. Future work includes a more detailed study of the parallel implementation of the scheme, including overlapping of implicit/explicit calculations and using fast multi-core computations for the explicit part.

References

[1] Roca, X., Nguyen, N., and Peraire, J., "GPU-accelerated sparse matrix-vector product for a hybridizable discontinuous Galerkin method," *49th AIAA Aerospace Sciences Meeting and Exhibit, Orlando, Florida*, 2011.

[2] Ascher, U. M., Ruuth, S. J., and Spiteri, R. J., "Implicit-explicit Runge-Kutta methods for time-dependent partial differential equations," *Appl. Numer. Math.*, Vol. 25, No. 2-3, 1997, pp. 151–167, Special issue on time integration (Amsterdam, 1996).

[3] Kennedy, C. A. and Carpenter, M. H., "Additive Runge-Kutta schemes for convection-diffusion-reaction equations," *Appl. Numer. Math.*, Vol. 44, No. 1-2, 2003, pp. 139–181.

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

[5] Cockburn, B. and Shu, C.-W., "Runge-Kutta discontinuous Galerkin methods for convection-dominated problems," *J. Sci. Comput.*, Vol. 16, No. 3, 2001, pp. 173–261.

[6] Peraire, J. and Persson, P.-O., *Adaptive High-Order Methods in Computational Fluid Dynamics*, Vol. 2 of *Advances in Computational Fluid Dynamics*, chap. 5 – High-Order Discontinuous Galerkin Methods for CFD, World Scientific Publishing Co., 2011.

[7] Persson, P.-O. and Peraire, J., "Newton-GMRES preconditioning for discontinuous Galerkin discretizations of the Navier-Stokes equations," *SIAM J. Sci. Comput.*, Vol. 30, No. 6, 2008, pp. 2709–2733.

[8] Boris, J. P., *On Large Eddy Simulation using subgrid turbulence models*, Springer-Verlag, New York, 1990, In J.L. Lumley, editor, Whither Turbulence? Turbulence at the Crossroads.

[9] Peraire, J. and Persson, P.-O., "The compact discontinuous Galerkin (CDG) method for elliptic problems," *SIAM J. Sci. Comput.*, Vol. 30, No. 4, 2008, pp. 1806–1824.

[10] Roe, P. L., "Approximate Riemann solvers, parameter vectors, and difference schemes," *J. Comput. Phys.*, Vol. 43, No. 2, 1981, pp. 357–372.

[11] Alexander, R., "Diagonally implicit Runge-Kutta methods for stiff o.d.e.'s," *SIAM J. Numer. Anal.*, Vol. 14, No. 6, 1977, pp. 1006–1021.

[12] Davis, T. A., *Direct methods for sparse linear systems*, Vol. 2 of *Fundamentals of Algorithms*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2006.

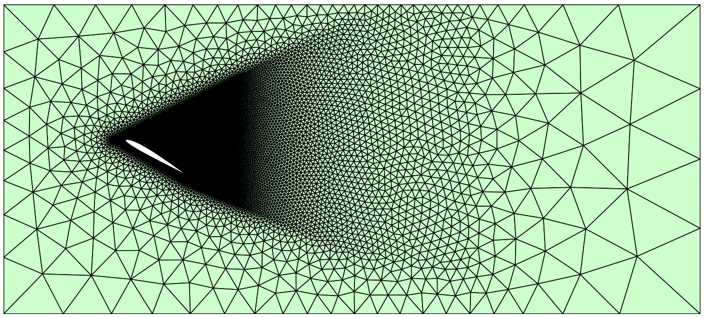
[13] Persson, P.-O., Peraire, J., et al., "The 3DG Project," `http://threedg.mit.edu`.

[14] Persson, P.-O., "Scalable Parallel Newton-Krylov Solvers for Discontinuous Galerkin Discretizations," *47th AIAA Aerospace Sciences Meeting and Exhibit, Orlando, Florida*, 2009, AIAA-2009-606.

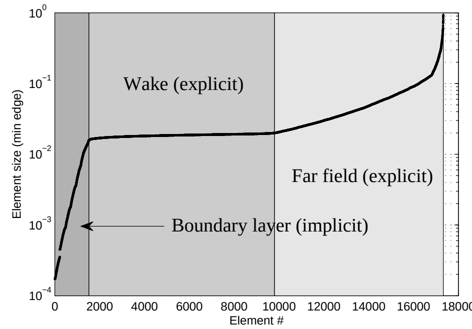
[15] Persson, P.-O., Bonet, J., and Peraire, J., "Discontinuous Galerkin Solution of the Navier-Stokes Equations on Deformable Domains," *Comput. Methods Appl. Mech. Engrg.*, Vol. 198, No. 17-20, 2009, pp. 1585–1595.

[16] Persson, P.-O. and Strang, G., "A simple mesh generator in Matlab," *SIAM Rev.*, Vol. 46, No. 2, 2004, pp. 329–345.

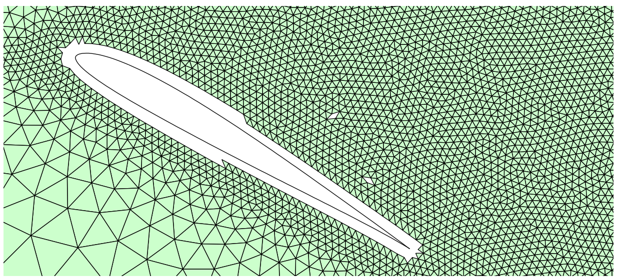
[17] Persson, P.-O. and Peraire, J., "Curved Mesh Generation and Mesh Refinement using Lagrangian Solid Mechanics," *47th AIAA Aerospace Sciences Meeting and Exhibit, Orlando, Florida*, 2009, AIAA-2009-949.



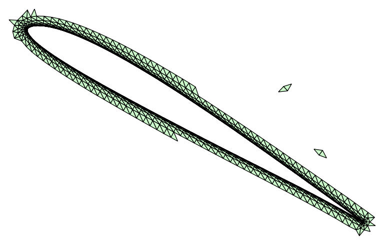
Mesh Cross-Section



Element size distribution in cross-section



Explicit elements



Implicit elements

Figure 5. Splitting of the mesh around an SD7003 airfoil into implicit and explicit elements. Less than 9% of the elements are in the boundary layer and integrated with the implicit scheme.

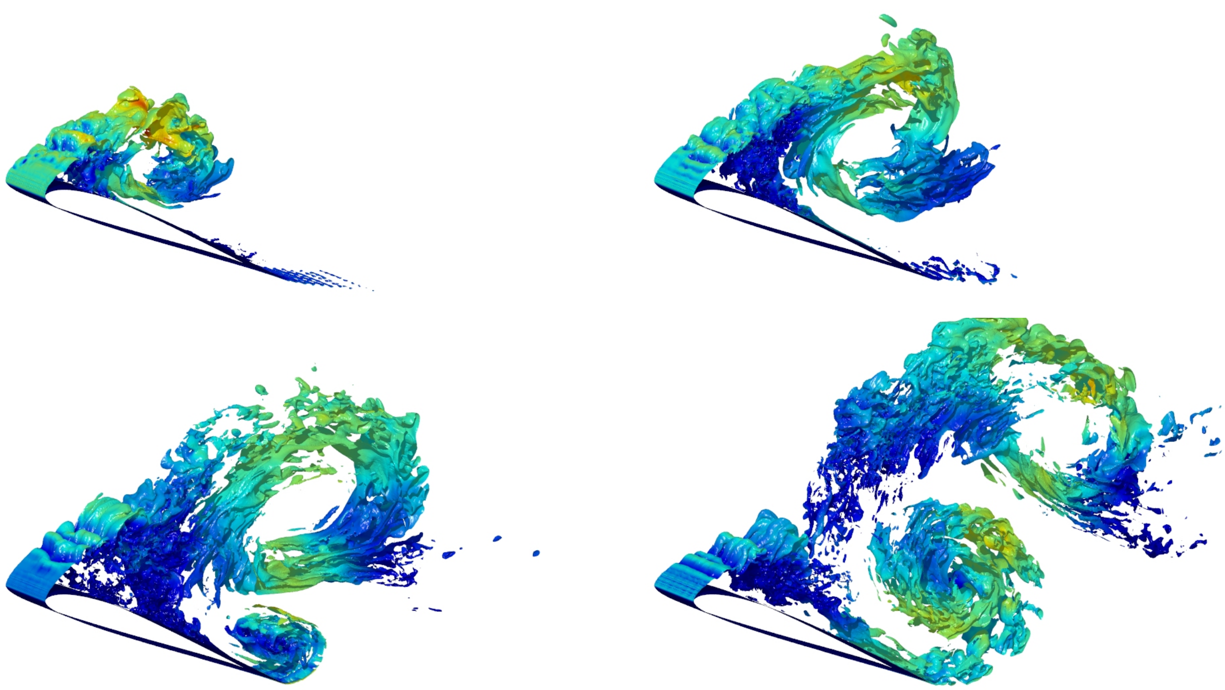


Figure 6. Four instantaneous solution to the flow over an SD7003 airfoil, shown by the Mach number as color on an isosurface of the entropy.