Future Directions of Computational Fluid Dynamics

F. D. Witherden* and A. Jameson†
Stanford University, Stanford, CA, 94305

For the past fifteen years computational fluid dynamics (CFD) has been on a plateau. Due to the inability of current-generation approaches to accurately predict the dynamics of turbulent separated flows, reliable use of CFD has been restricted to a small region of the operating design space. In this paper we make the case for large eddy simulations as a means of expanding the envelope of CFD. As part of this we outline several key challenges which must be overcome in order to enable its adoption within industry. Specific issues that we will address include the impact of heterogeneous massively parallel computing hardware, the need for new and novel numerical algorithms, and the increasingly complex nature of methods and their respective implementations.

I. Introduction

Computational fluid dynamics (CFD) is a relatively young discipline, having emerged during the last 50 years. During this period advances in CFD have been paced by advances in the available computational hardware, which have enabled its application to progressively more complex engineering and scientific problems. At this point CFD has revolutionized the design process in the aerospace industry, and its use is pervasive in many other fields of engineering ranging from automobiles to ships to wind energy. It is also a key tool for scientific investigation of the physics of fluid motion, and in other branches of science such as astrophysics. Additionally, throughout its history CFD has been an important incubator for the formulation and development of numerical algorithms which have been seminal to advances in other branches of computational physics.

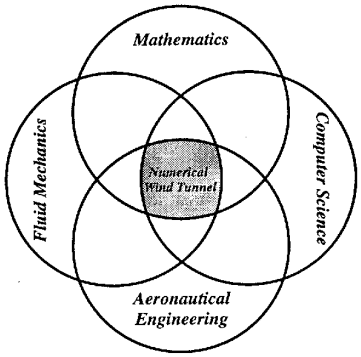

Figure 1: Multidisciplinary nature of CFD.

The objective of this article is to assess the present state of the art in CFD and to identify directions which can lead to further improvements culminating in the solution of hitherto intractable problems in the treatment of unsteady turbulent, separated and vortex dominated flows. Section II provides a brief history of CFD in aerospace to illustrate how CFD has evolved with hardware to reach the current plateau of Reynolds averaged Navier–Stokes (RANS) simulations. Section III reviews the capabilities and limitations of RANS

*Postdoctoral scholar, Department of Aeronautics and Astronautics, Stanford University, Stanford, CA.
†Professor (Research), Department of Aeronautics and Astronautics, Stanford University, Stanford, CA. AIAA fellow.

methods, indicating the need to use large eddy simulations (LES) for many industrial problems of interest, and also examines the computational requirements for LES. Section IV provides a brief overview of how computing hardware has changed over the past twenty years. Section V takes heed of these changes and presents the case for high-order methods as the future of unsteady CFD. Remaining challenges that must be overcome to enable the industrial adoption of scaling resolving CFD simulations are examined in section VI, while section VII outlines future targets beyond aerospace. Finally section VIII suggests some guidelines for future developments and presents our conclusions.

II. Brief history of CFD for aerospace

In this section we present a brief history of the evolution of CFD in the aerospace industry to illuminate the way in which we reached the current state of the art. While there have been pioneering studies of the numerical solution of the equations of gas dynamics dating back to the fifties,[1] the practical applications of CFD have been closely tied to the capabilities of the available computer hardware, both in speed and memory capacity, with an emphasis on steady state calculations.

The first major advance was the introduction of the panel method for the calculation of ideal potential flows governed by Laplace's equation over arbitrary geometries.[2] This was soon extended to lifting flows[3] and enabled the reasonably accurate prediction of low speed flows over complete configurations.

In the seventies the dominant problem was the calculation of transonic flow, because the most efficient cruising speed for commercial aircraft is in the transonic range, at the onset of drag rise due to the formation of shock waves, and also maneuvering combat aircraft fly in the transonic range. The calculation of transonic flows posed problems of great mathematical interest. Transonic flow is inherently nonlinear, and linearized potential flow models which had been successful for low speed subsonic flows and also high speed supersonic flows were not applicable. It was not feasible to solve the full three dimensional Euler equations of gas dynamics because existing computers had insufficient memory, and accordingly it was necessary to resort to nonlinear potential flow models. The corresponding equations are of mixed type, and could not be solved by existing numerical methods for partial differential equations. Following an early breakthrough in the solution of the transonic small disturbance equations,[4] methods of solving the full transonic flow equation were found within a few years.[5] The FLO22 software developed by Jameson and Caughey in 1975 enabled the reasonably accurate prediction of transonic flow over a swept wing, and it was immediately put to industrial use for the design of the wings of the McDonnell Douglas C17 and the Canadair Challenger. These calculations could be performed in about three hours on the Control Data 6600, which represented the state of the art in the early seventies, with a computing speed of about 1 megaflop. The biggest problem was the lack of enough memory. The CDC 6600 had a memory capacity of 131,000 60-bit words (about 1 megabyte) and this was not enough to store the full three dimensional solution, which had to be stored in the disk, while the solution had to be calculated plane by plane.

By the eighties, with the advent of vector computers, it became feasible to calculate solutions of the Euler equations in two and three dimensions. The first vector computer, the Cray 1, achieved computing speeds of about 100 megaflops, but early models had quite limited memory capacity of 32 or 64 megabytes. Drawing on classical contributions by pioneers such as Godunov, Lax and MacCormack, rapid advances were made in numerical shock capturing algorithms, with important contributions by Van Leer, Roe, Harten, Osher, Engquist, and many others. Three dimensional Euler calculations were widely performed using second order accurate finite volume methods, such as the method proposed by Jameson, Schmidt and Turkel.[6] Figure 2 shows an Euler calculation for the Northrup YF23 performed with this method. The year 1986 saw the appearance of Euler solutions for complete aircraft using unstructured meshes.[7] Figures 3 and 4 show representative examples of calculations performed in the late eighties. This period also saw the emergence of commercial CFD software, mostly stemming from industrial simulations of incompressible turbulent viscous flows.

In the nineties further increases in the speed and memory capacity of computers made it feasible to perform simulations using the Reynolds averaged Navier-Stokes (RANS) equations with a variety of turbulence models, generally supplanting the use of inviscid potential flow or Euler solutions with boundary layer corrections, although these methods continue to be used for preliminary design. Aerodynamic shape optimization methods based on control theory also emerged as a useful tool for preliminary design.[9]

The general trajectory of the evolution of CFD towards higher fidelity models and more complex geometries between 1970 and 2000 is illustrated in Figure 5. However, during the last 15 years aerospace

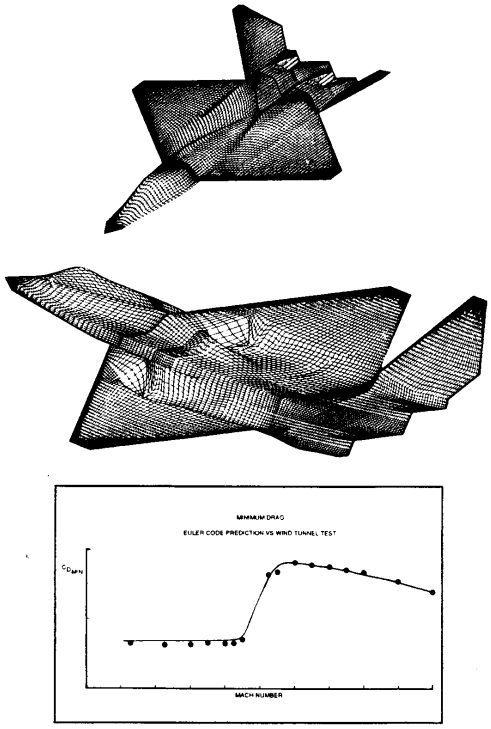

Figure 2: Northrup YF23 calculated using the Jameson-Schmidt-Turkel method (FLO57) by Richard Busch, Jr.[8]

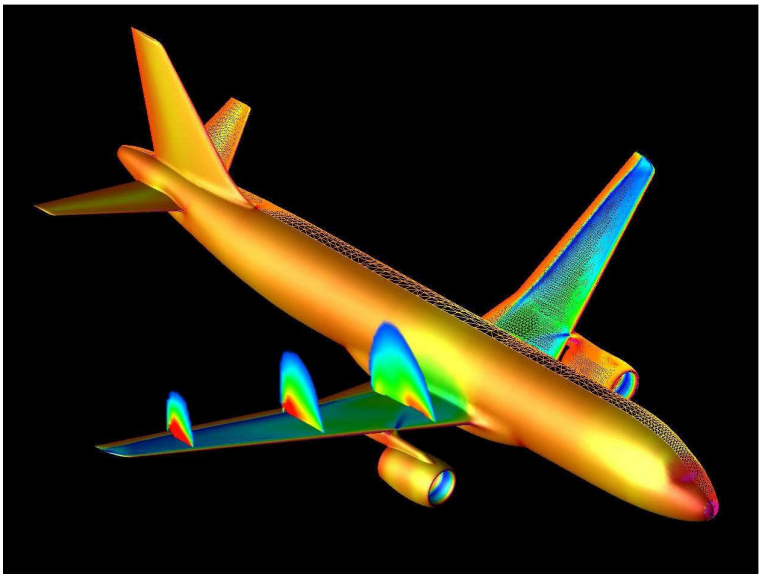

Figure 3: Airbus A320 calculated using the AIRPLANE code.

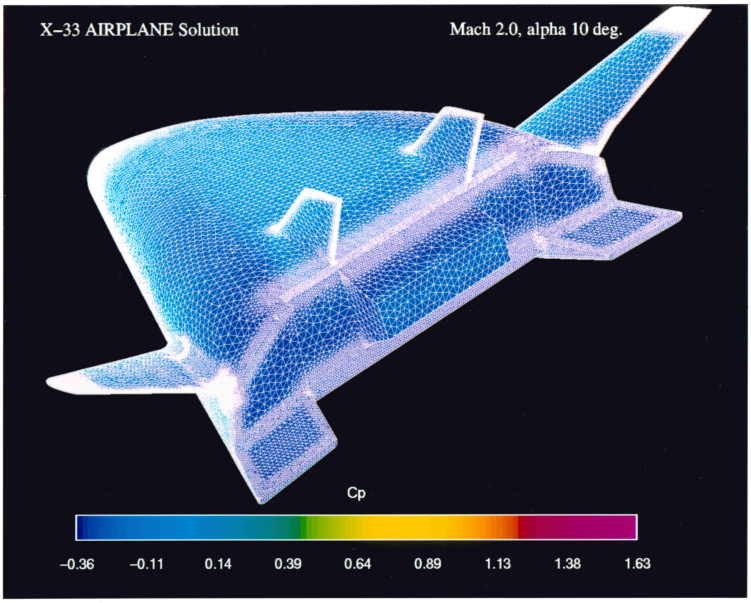

Figure 4: X-33 calculated using the AIRPLANE code.

applications of CFD have remained on a RANS plateau, as discussed in the next section, because a further advance to LES or direct numerical simulations (DNS) still requires many orders of magnitude increases in computer speed and memory.

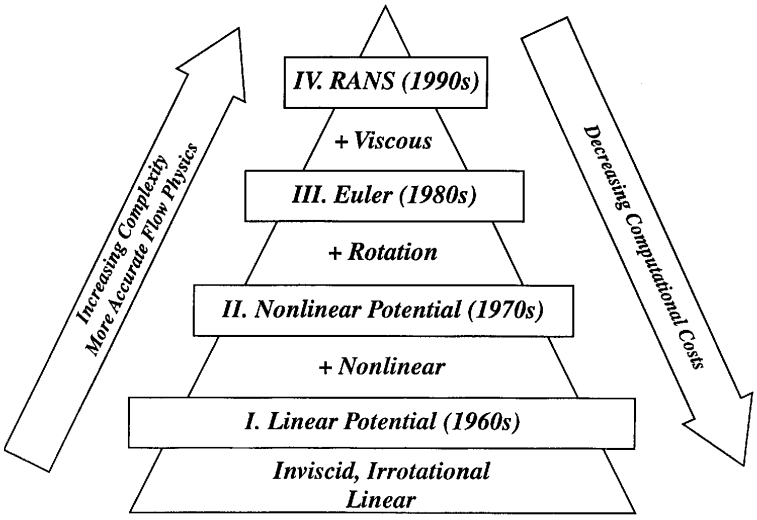

Figure 5: Hierarchy of fluid flow models.

III. The current status of CFD

In the next two sections we examine the current state of the art of applications of CFD in the aerospace industry and also the impact of hardware developments which should eventually enable industrial use of LES. At the present time second order accurate finite volume RANS solvers are widely and successfully used in Aerospace, and also in other applications such as Formula 1 racing cars, using meshes which may have more than 100 million cells. The calculation of steady viscous transonic and supersonic flow is a solved problem. CFD is the main tool for preliminary aerodynamic design, with wind tunnel testing in a supporting

role. CFD applications in the design of the Airbus A380 and Boeing 787 are illustrated in figures 6 and 7. However these applications are essentially limited to steady attached flows. The present situation is summarized by the following quotation from the NASA CFD Vision 2030 Study released in 2014:[10] *" In spite of considerable successes, reliable use of CFD has remained confined to a small but important region of the operating design space due to the inability of current methods to reliably predict turbulent-separated flows."* This situation is further illustrated in Figure 8, which displays the flight envelope of a commerical jet aircraft in terms of equivalent airspeed against load factor, and indicates that RANS predictions are only reliable in a small region centered on the cruise point. Other parts of the envelope may incur separated and unsteady flow phenomena such as buffet that cannot adequately be modeled by RANS methods. Detailed structural design requires knowledge of loads at many thousands of points covering the full flight envelope. Up to the present time it has remained more reliable to obtain this data by wind tunnel testing, and also cheaper because once a model has been built it is possible to obtain a lot of data quickly, and this is less expensive than the cumulative cost of a very large number of RANS calculations.

Figure 6: CFD applications in the design of Airbus A380.

Figure 7: CFD applications in the design of Boeing 787.

Other examples of currently intractable problems include high-lift systems, rotorcraft flows, internal flows in turbomachinery, and airframe and landing gear noise. While progress is ongoing it appears that RANS methods cannot adequately model all the associated flow phenomena, and LES will eventually be needed for more reliable predictions, most likely in conjunction with higher order numerical methods to reduce numerical dissipation, and enable, for example, long term tracking of vortices.

Over the past thirty years various estimates for the computational cost of LES have been suggested.[11–15] At the time of writing the cost is generally regarded as scaling in accordance with $Re^{1.9\sim2.4}$ where Re is the Reynolds number. For exterior flows it is also accepted that the majority of the resolution is concentrated in and around the boundary layer, with a vanishing small number of degrees of freedom being contributed from

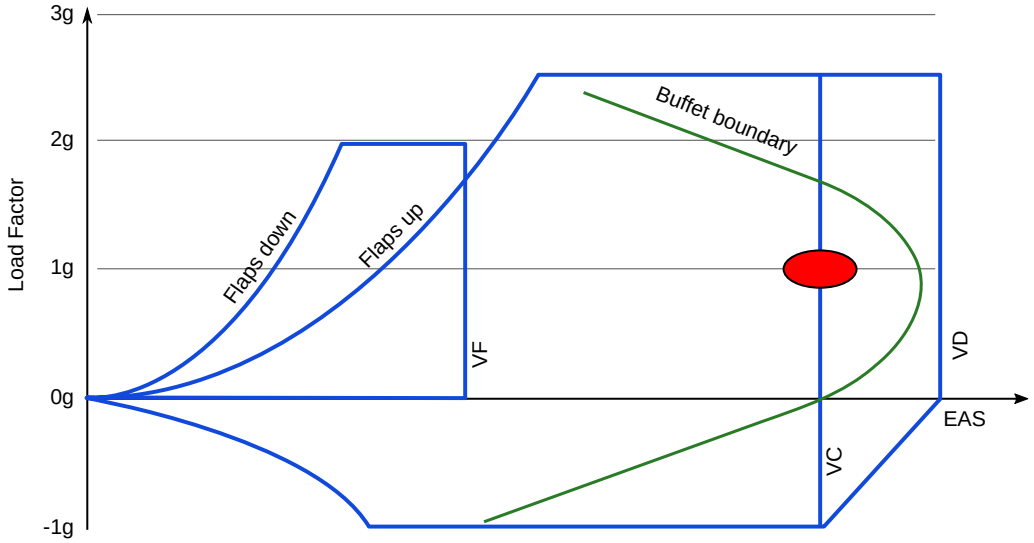

Figure 8: Flight envelope for a commercial jet aircraft in terms of equivalent airspeed versus load. The red oval indicates the region in which current generation RANS schemes can be regarded as reliable. Adapted from a slide originally presented by Murray Cross.

the far-field. Practitioners have therefore long sought to reduce the computational cost of LES by modeling the near-wall flow. Wall modeling has the potential to reduce the cost of LES to $Re^{0.28\sim0.4}$.[13,15] Estimates for the computational requirements of wall modeled LES can be seen in Figure 9. As a point of reference, at the time of writing, the computational capability of a quad-core laptop is of order 100 gigaflops whereas the world's fastest supercomputer, Sunway TaihuLight, is of order 100 petaflops. From the figure it is clear that even the most rudimentary LES are pushing the limits of what can be achieved on current generation hardware. Thus, to understand the future prospects for LES it becomes necessary to consider the impact of current trends in computer hardware on the design of CFD software.

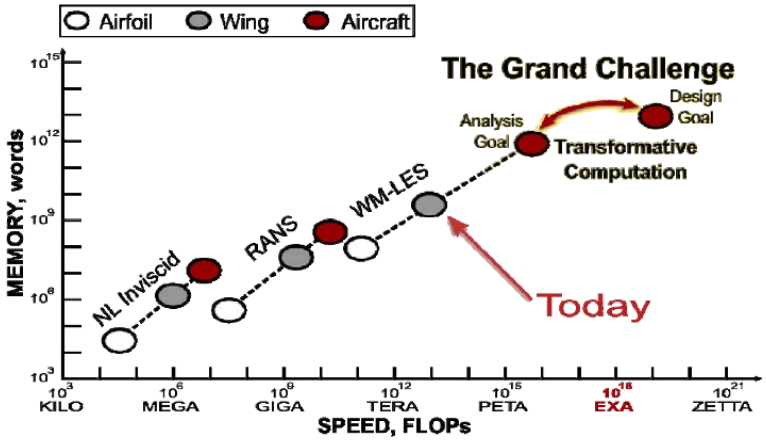

Figure 9: Estimates given by Ashby et al.[16] on the computational cost of wall modeled LES compared with RANS and non-linear inviscid computations.

IV. Impact of hardware

Traditionally, numerical methods have been designed with the goal of minimising the total number of floating point operations required to achieve convergence. The rationale behind this notion stems from the fact that digital computers can perform a finite number of such operations per second; hence, if one can modify an algorithm to reduce the number of floating point operations then a commensurate reduction

in the wall clock time should result. The first shake-up to this paradigm came with the introduction of vector supercomputers in the early 1980s. Vector systems, such as the Cray 1, were capable of performing significantly more floating point operations than their predecessors, but only so long as a large portion the numerical method could be rewritten in vector form. This requirement is one of the first examples of a change in computing hardware forcing a change in numerics. Over the next twenty years or so the field of high-performance computing was characterised by ever-increasing levels of parallelism, with the low core count vector supercomputers slowly giving way to extremely high core count clusters constructed from commodity hardware. Despite these developments the fundamentals remained largely unchanged with a key objective of method design being to minimise the aggregate number of floating point operations.

However, in addition to the trend towards parallelism, since the mid-1990s a new trend has developed: that of an ever-increasing disparity between the arithmetic capabilities of processors, measured in floating point operations per second, and the net bandwidth to/from main memory. This trend can be seen in Figure 10 and Table 1. Here we see that whilst the compute capability has increased exponentially in line with Moore's law, the improvement in memory bandwidth has been much slower. Thus, numerical methods that were computationally balanced in the 1990s, which includes those developed for CFD, are not at all well balanced on today's machines. An analysis by Langguth et al.[17] of a perfect cache-aware tetrahedral finite volume method found that on modern hardware the scheme itself is restricted to just 3% of peak flops.

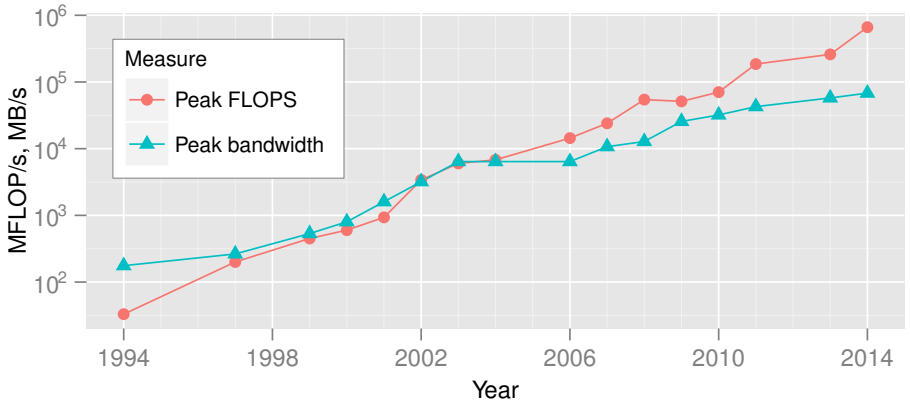

Figure 10: Trends in the peak floating point performance (double precision) and memory bandwidth of sever-class Intel processors from 1994–2014. Figure expanded from that of Witherden et al.[18]

	1994	2014	Ratio
MFLOP/s	33	604,000	18,303
MB/s	176	68,000	386

Table 1: Raw data from Figure 10 showing the impact of twenty years of progress.

On a parallel track, starting in the late 2000s there has also been a trend towards increasing heterogeneity within HPC. Massively parallel accelerators, including GPUs from AMD and NVIDIA, and co-processors from Intel, are now a mainstay in the TOP500 list. This trend is not just limited to the high-end, however. Powerful integrated GPUs are now also commonplace on consumer grade hardware. An example of this can be seen in Figure 11. In this particular configuration more single precision flops are provided by the integrated GPU than all four of the CPU cores combined. If utilised effectively, accelerators can deliver increased throughput relative to many-core CPUs for a given power draw. However, this comes at the cost of increased development costs and ongoing maintenance burdens, especially if one wishes to target multiple devices from a single code base. Moreover, such devices are normally connected to the system through relatively low-bandwidth links, typically PCI express. Thus, unless the majority of the kernels which constitute a numerical scheme can be offloaded onto the device it is not uncommon for any performance improvements to be offset by the costs of moving data on and off of the device. The difficulty with offloading large portions of a scheme are exacerbated for codes which make extensive use of third party software packages; for the effective acceleration is now contingent on the availability of accelerated versions of these packages.

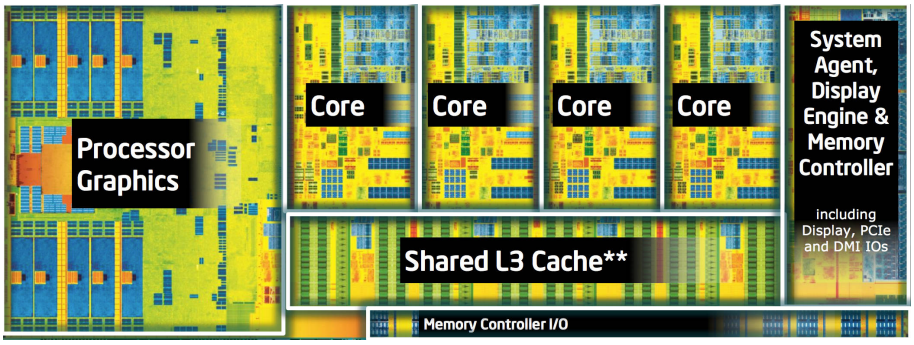

Figure 11: False color image of the die on a four core Intel Haswell CPU. We note here the substantial floorspace occupied by the integrated GPU relative to that of the CPU cores. Image courtesy of Intel Corporation.

The extreme levels of parallelism provided by accelerators, in conjunction with the relatively high costs associated with synchronisation, can also pose problems for algorithms that suffer a loss of numerical efficiency with increasing levels of parallelism. Examples of such schemes include block Gauss–Seidel and block incomplete LU preconditions. These are typically encountered in numerical schemes which involve the solution of large scale linear systems. While such algorithms can be ported over to accelerators, the improved throughput is often offset by the fact that due to the increased level of parallelism more iterations are required. When seeking to accelerate codes, further complications arise from the great disparities in terms of number of host CPU cores to the number of accelerators. Ratios range from as high as 16 cores per accelerator to as low as one core per accelerator. This greatly complicates code development as an offload strategy that is effective for a high-core-to-accelerator-ratio is unlikely to be effective for a low-core-to accelerator-ratio.

Going beyond a single node, another issue that arises is the relatively high latency of inter-node communication; the majority of which can be attributed to the time required for the relevant electrical signals to transverse from one node to another. Numerical schemes whose parallel efficiency is extremely sensitive to the performance of such operations, specifically collective operations, may therefore encounter network-related scaling issues. It is therefore becoming increasingly important to design numerical schemes in such a way as to enable the use of asynchronous (non-blocking) communication. This enables the latency bound communication operations to occur in the background without blocking the main code; thus facilitating the overlapping of communication and computation. Although non-blocking point-to-point operations are commonplace, it is only more recently that it has become possible to issue non-blocking collective operations—of the kind commonly encountered when solving large global systems.

This state of affairs was described by researchers at Sandia National Laboratories in a 2013 report as a "perfect storm".[19] The report further notes that the "halcyon years" of HPC are no more, with the effective utilisation of leadership class machines becoming progressively more difficult. A more recent 2016 report, authored jointly by a committee consisting of representatives from the NSA, DOE, and others, on U.S. leadership in HPC[20] suggests that the trend—at least at the high end—is likely to be towards architectures which are novel and of a specialized nature.

V. High-order methods

This changing hardware landscape places new constraints on the design of numerical schemes; namely that it is no longer sufficient for a method to simply be amenable to parallel computing. Rather, it is now also necessary to consider the arithmetic intensity of the scheme, that is to say how much useful computation it performs for each byte of data transferred. Although more recent techniques such as temporal blocking[21] can ameliorate some of the problems faced by classical finite difference and finite volume schemes, this comes at the cost of a substantial increase in code complexity.

Based on these considerations, and others, we arrive at the following set of requirements for a numerical scheme.

1. The capability to work on unstructured grids. If the scheme supports curved elements then, ideally, there should be no computational penatly associated with the use of such elements.

2. Sufficiently low dispersion and dissipation errors to enable the effective simulation of unsteady vortex dominated flows.

3. Be extremely amenable to parallel computing. Ideally, many of the core building blocks of the scheme should map onto vendor provided *performance primitives*; for example matrix multiplications, convolutions, fast Fourier transforms, etc. These primitives should all have reasonable arithmetic intensities.

4. The numerical performance of the scheme should not degrade with increasing levels of parallelism.

5. Insofar as possible, the scheme should be designed to avoid inter-node communication. Communication should be asynchronous and non-blocking.

6. Finally, the efficient parallel implementation of these schemes should *not* require the regular use of dynamic repartitioning routines.

Within the context of solving the compressible Navier–Stokes equations within the vicinity of complex geometrical configurations, the most promising class of schemes are *high-order discontinuous spectral element methods*. Examples of such schemes include: the venerable discontinuous Galerkin (DG) method, originally proposed by Reed and Hill[22] in 1973 and further developed for conservation laws by Cockburn and Shu in a seminal series of papers,[23–27] spectral difference (SD) methods originally proposed under the moniker 'staggered-gird Chebyshev multidomain methods' by Kopriva and Kolias in 1996[28] and later popularised by Sun et al.,[29] and the flux reconstruction (FR) approach of Huynh.[30] All of these schemes are similar in the sense that, inside of each element, they represent the solution using a high-order polynomial that is *discontinuous* between adjoining elements (cf. the finite element method). Elements are then coupled together via interface flux functions in the exact same manner as in the finite volume method. The use of polynomials inside each element grants these schemes superior resolving power per degree of freedom in comparison with a second order accurate finite volume scheme. This translates into fewer overall degrees of freedom being required for a given level of accuracy. Additionally, since there are now multiple degrees of freedom concentrated in a single element it is relatively easy to achieve a high level of spatial data locality, even on fully unstructured grids. A consequence of this is that, at moderate polynomial orders, the arithmetic intensity of these schemes can be an order of magnitude greater than second order finite volume methods. Moreover, when paired with explicit time-stepping each element only interacts with its direct neighbors and does so only via their common face. Thus, the communication pattern associated with parallel implementations of these schemes is point-to-point and involves halo exchanges that are only one level deep. If an implicit time-stepping scheme is employed, on the other hand, then it is relatively easy to ensure that all of these schemes have a compact stencil.

A good example of a high-order CFD code is PyFR[18] of Witherden et al. Written primarily in the Python programming language and employing the FR discretisation, PyFR was designed from the ground up to run efficiently on modern hardware platforms. As such it is *performance portable* across GPUs from AMD and NVIDIA and CPUs/co-processors from Intel. PyFR has also been shown to be capable of operating on heterogeneous systems with a mix of GPUs and CPUs.[31] Numerical studies with PyFR[32,33] also demonstrate the ability of these schemes to scale up to simulations with hundreds of billions of degrees of freedom on thousands of GPUs. An example of one such simulation can be seen in Figure 12. Moreover due to the highly favorable computational properties of FR schemes PyFR is able to achieve this scaling whilst still maintaining close to 50% of peak flops. Further details regarding the scalability of PyFR can be seen in Table 2. Finally, we note that all of this is accomplished with just $\sim 9,000$ lines of Python code.

Table 2: Weak scaling of PyFR on the Titan supercomputer for a simulation with fourth order solution polynomials. Adapted from the data in.[32]

N_{GPU}	Petaflops	% Peak flops
3,000	1.93	49.2%
6,000	3.88	49.3%
9,000	5.79	49.2%
12,000	7.79	49.3%
15,000	9.65	49.1%

Figure 12: Snapshot of isosurfaces of Q-criterion colored by velocity magnitude for flow over a T106D low pressure turbine cascade. The simulation, performed with PyFR, had 113 billion degrees of freedom and was run across 5,000 NVIDIA K20X GPUs on the Titan supercomputer. Image from.[32]

The ability for high-order methods to be implemented efficiently on modern hardware enables them to deliver on their promise of improved accuracy at a reduced cost compared with lower order finite volume methods.[34]

VI. Challenges

If high-order methods are to be routinely used for the application of LES within the vicinity of complex geometrical configurations then several challenges will need to be overcome. Many of these are not new, having been previously identified by Wang et al. in their 2013 review paper[35] on the status of high-order methods for CFD.

Efficient implicit time-stepping. The memory requirements for implicit time-stepping schemes in a high-order DG type scheme scale as p^6 where p is the polynomial order. (As a point of comparison explicit time integration schemes have memory requirements which scale as p^3.) This makes even modestly sized problems computationally infeasible at all but the lowest polynomial orders. The problem is exacerbated by the fact that on large scale HPC machines the amount of memory provided per core of compute has remained stagnant. Techniques such as line DG[36] have the potential to reduce this requirement to p^4, albeit with the limitation that the grid is restricted to hexahedral elements. Over the past five years there has also been a renewed interest applying hybrid implicit-explicit (IMEX) schemes[37,38] to unsteady flow problems. In an IMEX scheme larger elements are treated explicitly whereas smaller elements are treated implicitly. The effectiveness of these schemes is therefore proportional to the number of elements in the domain which are suitable for explicit time stepping. The presence of both implicit and explicitly time-stepped elements results in a dynamic load-balancing problem. It is therefore necessary for implementations to repartition the grid as the simulation progresses. Further details regarding the application of IMEX schemes to flow problems can be found in the papers of Persson[39] and Vermeire and Nadarajah.[40]

More recent work has sought to adapt hybridized DG (HDG) schemes, which seek to reduce the degree of freedom count for elliptic problems,[41,42] to the hyperbolic compressible Navier–Stokes equations. An example of such an approach is the interior embedded discontinuous Galerkin (IEDG) method of Fernandez et al.[43] Although the method, and closely related variations thereof, is successful in reducing the number of overall degrees of freedom—and hence the memory usage—it is still unclear if they can be implemented efficiently on modern hardware. Further, the temporal scales of LES are often only two or three orders of magnitude greater than the CFL limit. Thus, in contrast to RANS it is not meaningful to run with CFL numbers of $\sim 10^5$. These more stringent temporal accuracy requirements complicate the design of schemes.

Wall and sub-grid scale modeling. It has generally been expected that sub-grid scale (SGS) models will be needed for LES of high Reynolds number flows. Existing models are not easily combined with high-order methods. For Reynolds numbers below $\sim 10^6$ (which are relevant to internal flows such as turbomachinery) a number of investigators have obtained results with no explicit SGS model that

have exhibited excellent agreement with experimental data. Such approaches, termed implicit LES or, alternatively under-resolved DNS, rely on the inherit dissipative properties of the numerical scheme to control high wave number modes beyond the spatial resolution of the grid. Their applicability is the subject of ongoing discussion and research.

High Reynolds number wall bounded flows contain increasingly small scales within the vicinity of the wall; leading to computational requirements which are not going to be available for the next decade. In order to enable industrial applications of LES of flows of this type, such as airplane wings, wall models will be required. Within the context of high-order methods, the development of wall models is also a subject of ongoing research.

High-order grid generation. For a given total number of degrees of freedom, mesh elements must be substantially larger than those employed in finite volume methods. If these coarse elements are to provide a reasonable representation of boundaries it is necessary for the elements themselves to be curved. At the time of writing most workflows are based around first generating a coarse linear mesh (a formidable challenge in and of itself) and then using a separate tool to 'snap' the extra high-order nodes onto the CAD surface. Care is required here so as to ensure that this snapping process does not yield elements which are either invalid due to self-intersection, or of poor quality. Whilst substantial progress has been made over the past five years[44–47] questions surrounding the reliability and robustness of these approaches remain. The need for external tools also has a negative impact on the workflow and thus for high-order methods to gain any sort of traction at an industrial level, fully integrated approaches will be required.

There are also open questions around the degree of curvature required for high-order methods. For instance, if a simulation is run using fourth order solution polynomials, is it necessary for the mesh to also be specified by fourth order polynomials? Additionally, the community is still in the process of adapting meshing guidelines for finite volume methods, such as placing the first cell at $y^+ = 1$, to high-order methods.

Flows with moving and deforming boundaries. Many important industrial applications, such as rotorcraft, turbomachinery, wind turbines, and aeroelasticity, involve flows with moving and deforming boundaries. Numerical techniques for these flows include mesh sliding, overset grids, and mesh reconnection. All of these approaches present challenges around maintaining higher order accuracy and preventing spurious reflections at internal grid boundaries. They must also be carefully implemented in order to avoid excessive computational costs.

Despite these difficulties, the presence of a high-order polynomial inside of each element and the relatively larger sizes of these elements, can result in codes which are more robust and performant than their low order counterparts. A good example of this is in the overset grid approach where, with high order methods, there is no need to reconstruct a donor solution from multiple low order cells. Instead, one is able to obtain a donor solution by simply interrogating the high-order solution polynomial inside of a *single* element.

Error estimation, automated adaptation, and the design of experiments. The engineering design process can only rely on numerical simulations if there errors can be certified or quantified. Ideally, we should have *a posteriori* error bounds which can be calculated from the solution. Such bounds are very hard to obtain in unsteady non-linear simulations. There are, however, several useful methods including those based around Richardson extrapolation,[48] those based around error transport equations,[49–51] and output error estimation by adjoint methods.[52–54] These can all be used to enable automatic adaptation both in mesh size and polynomial order (so called *h*-*p* adaptation).

Since even the most rudimentary forms of LES will require advances in computer hardware to the exaflop range LES will, for the foreseeable future, be used sparingly. These limitations around the number of simulations that can be undertaken raise questions with regards to the optimal design of computer experiments, and how best to integrate the outcomes into lower fidelity models. Further, for the simulations to stand any chance of serving in lieu of a wind tunnel experiment than some form of uncertainty quantification (UQ) will also be required. This necessitates the development of scalable UQ algorithms.

Multiphysics. A wide range of important industrial problems involve multiphysics. Notable examples are reacting flows and fluid structure interaction. The accurate simulation of turbulent combustion is of great relevance to the design of both piston engines and gas turbines. Ionization and chemical reactions also occur in hypersonic flows which are relevant for both launch and re-entry systems for spacecraft.

The importance of multiphysics is also likely to increase over time. There exist classes of problems, for instance acoustic loading in weapons bays, which with RANS are dominated by errors in the flow field. However, if LES were to be employed the leading error term is very likely to be the result of neglecting fluid-structure effects. Thus, in order to realise the potential of LES for these types of flows, it is essential that some level of multiphysics also be incorporated into the simulation.

Knowledge extraction from big data. Unsteady simulations have the potential to generate huge amounts of data. Whereas a converged RANS calculation gives rise to a single flow field, an unsteady simulation will typically generate tens of thousands of flow fields. This, combined with the far greater number of grid points required for LES, means that it is computationally intractable to retain each such flow field. It will therefore be necessary for practitioners to decide in advance what metrics and statistics are of interest. The net result is that a computational simulation is turned into a *numerical experiment*, with many of the same limitations of the wind tunnel experiments that LES is envisioned to supplant. For LES to deliver insights beyond those of a comparable wind tunnel campaign advances in real-time data analysis, automated feature detection, and scalable approaches for dimensionality reduction, such as dynamic mode decomposition,[55,56] are needed.

Machine learning and data mining are likely to play an important role here. Potential applications include data-driven discovery[57,58] and the use of neural networks to construct application-specific turbulence models.[59,60]

Implementation complexity. Modern CFD packages are extremely complex pieces of software; the popular open source codes OpenFOAM[61] and SU2[62] weigh in at $\sim$700,000 and $\sim$300,000 lines of code, respectively. Without a radical rethink around development methodologies it is highly probable that future codes, employing more elaborate numerical schemes and optimized for a range of hardware platforms, will be substantially more complex. This level of complexity has a negative impact on the time required for new developers—typically graduate students—to get up to speed with the code and begin making contributions. Whilst some complexity is inherent, a large amount stems from the use of overly verbose programming languages and ineffective abstractions.

Instead of writing an entire code in a low level language, such as C++ or Fortran, the current trend appears to be towards using a low level language only for performance sensitive kernels, with the remainder of the code authored in a high level scripting language; for example Julia or Python. If done correctly this approach can almost completely eliminate so-called boiler-plate code along with many common sources of bugs; including memory leaks and out of bounds memory accesses. More importantly, this can be accomplished *without* sacrificing either run-time performance or scalability.

VII. Beyond aerospace

While this article is mainly focused on future directions for CFD in aerospace applications, any progress which is made will spill over to many other fields. Some of these are briefly reviewed in this section. Most immediately, advances in CFD directly impact the automotive, marine, and wind energy industries. These applications involve low Mach number incompressible flows. Many problems are concerned with flows around bluff bodies; something which existing RANS methods struggle to accurately predict. Advances in LES and hybrid RANS/LES methods will be needed to make more accurate predictions of the flows over cars and trucks, with a long-term goal of enabling active flow control to reduce drag. Such developments should also enable the design of improved wind turbines and wind energy farms. Marine applications pose further challenges including the presence of free surfaces, wave breaking, and cavitation. An example of cavitation for a marine propeller can be seen in Figure 13.

Another challenging field involving incompressible flows at low Reynolds numbers is biological fluid dynamics. An example of such a flow can seen in Figure 14. These flows also often involve fluid structure interactions with compliant walls. Advances in CFD are also likely to benefit the closely related field of computational cardiac electrophysiology. LES also has applications in the architectural community in

Figure 13: Controllable-pitch propeller in a large cavitation channel.

the design of energy efficient heating, ventilation, and air conditioning systems. Outside the engineering disciplines, advances in CFD will directly impact geophysics and astrophysics, where the discontinuous high-order methods espoused in the previous sections are starting to gain traction.[63–65] It is therefore likely that CFD will continue to be an incubator for advances in both numerical algorithms and computational implementations seminal to all branches of computational physics.

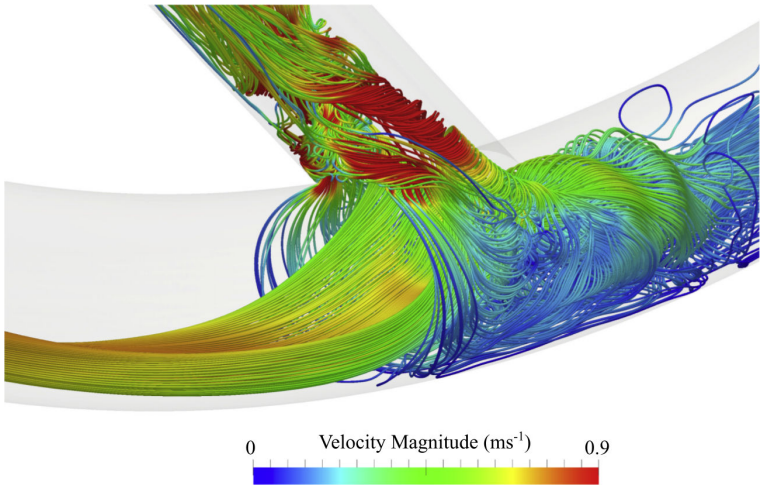

Figure 14: Temporal snapshot of streamlines colored by velocity magnitude for a biological flow simulation inside of an arterio-venous fistulae. Image courtesy of Peter Vincent.[66]

VIII. Conclusions

The early development of CFD in the aerospace industry was primarily driven by the need to calculate steady transonic flows: this problem is quite well solved. CFD has been on a plateau for the last 15 years with second order accurate finite volume methods for the RANS equations almost universally used in both commercial and government codes. However, these methods cannot reliably predict complex separated, unsteady, and vortex dominated flows. Ongoing advances in both numerical algorithms and computer hardware and software should enable an advance to LES for industrial applications within the foreseeable

future. Given the requirements of LES it is the opinion of the authors that research should focus on high-order methods with minimal numerical dissipation for unstructured meshes to enable the treatment of complex configurations.

Current obstacles to the wider adoption of high-order method which call for further research include (i) the high memory requirements associated with implicit time stepping schemes; (ii) difficulties with generating high quality curved unstructured grids around real-world geometries; (iii) a lack of wall models, something which is vital for enabling the affordable simulation of high Reynolds number wall-bounded flows; (iv) the relative complexity of implementing these schemes for high performance hardware whilst still being accessible to graduate students.

Thus, CFD is still a very exciting discipline, both in its own right and as a conduit for basic research in numerical analysis and HPC. By addressing the challenges outlined in this paper it is possible to enable a step-change across several high-tech fields. However, if we as a community are to realize these breakthroughs we need to focus on the development of pipelines that will enable us to rapidly translate numerical advances into application breakthroughs. This idea is illustrated in Figure 15. To maintain focus these pipelines should then be applied to the simulation of hitherto intractable flow problems (so called *grand challenge problems*).

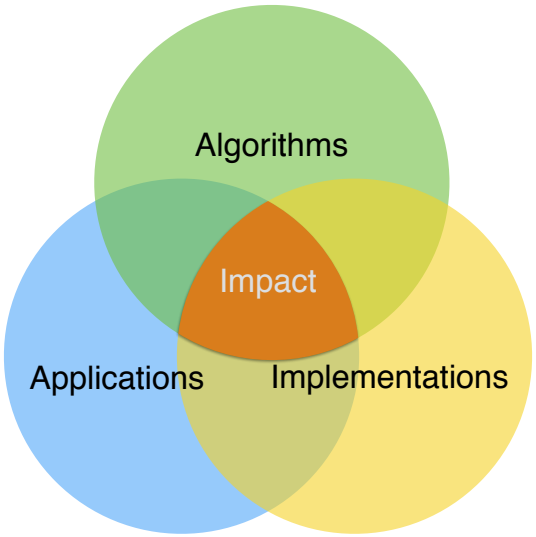

Figure 15: One possible route to impact in CFD.

Acknowledgements

The authors would like to thank the Air Force Office of Scientific Research for their support via grant FA9550-14-1-0186 under the direction of Jean-Luc Cambier.

References

[1]Godunov, S. K., "A Difference Method for the Numerical Calculation of Discontinuous Solutions of Hydrodynamic Equations," *Mat. Sbornik*, Vol. 47, 1959, pp. 271–306, Translated as JPRS 7225 by U.S. Dept. of Commerce, 1960.

[2]Hess, J. L. and Smith, A. M. O., "Calculation of the non-lifting potential flow about arbitrary three dimensional bodies," *Douglas Aircraft Report, ES*, Vol. 40622, 1962.

[3]Rubbert, P. E. and Saaris, G. R., "A General Three-Dimensional Potential Flow Method Applied to V/STOL Aerodynamics," SAE Paper 680304, 1968.

[4]Murman, E. M. and Cole, J. D., "Calculation of plane steady transonic flows," *AIAA Journal*, Vol. 9, 1971, pp. 114–121.

[5]Jameson, A., "Numerical calculation of the three dimensional transonic flow over a yawed wing," *AIAA Computational Fluid Dynamics Conference*, 1973, p. 3002.

[6]Jameson, A., Schmidt, W., and Turkel, E., "Numerical solution of the Euler equations by finite volume methods using Runge Kutta time stepping schemes," *AIAA, Fluid and Plasma Dynamics Conference, 14th, Palo Alto, CA, June 23-25, 1981.*, Vol. 1, 1981.

[7]Jameson, A., Baker, T. J., and Weatherill, N. P., "Calculation of Inviscid Transonic Flow over a Complete Aircraft," AIAA paper 86-0103, AIAA 24th Aerospace Sciences Meeting, Reno, NV, January 1986.

[8]Busch Jr, R., "Computational fluid dynamics in the design of the Northrop/McDonnell Douglas YF-23 ATF prototype," *AIAA paper 91*, Vol. 1627, 1991.

[9]Jameson, A., "Aerodynamic Design via Control Theory," *J. Sci. Comp.*, Vol. 3, 1988, pp. 233–260.

[10]Slotnick, J., Khodadoust, A., Alonso, J., Darmofal, D., Gropp, W., Lurie, E., and Mavriplis, D., "CFD vision 2030 study: a path to revolutionary computational aerosciences," 2014.

[11]Peterson, V. L., Kim, J., Holst, T. L., Deiwert, G. S., Cooper, D. M., Watson, A. B., and Bailey, F. R., "Supercomputer requirements for selected disciplines important to aerospace," *Proceedings of the IEEE*, Vol. 77, No. 7, 1989, pp. 1038–1055.

[12]Spalart, P., Jou, W., Strelets, M., Allmaras, S., et al., "Comments on the feasibility of LES for wings, and on a hybrid RANS/LES approach," *Advances in DNS/LES*, Vol. 1, 1997, pp. 4–8.

[13]Choi, H. and Moin, P., "Grid-point requirements for large eddy simulation: Chapman's estimates revisited," *Physics of fluids*, Vol. 24, No. 1, 2012, pp. 011702.

[14]Piomelli, U., "Wall-layer models for large-eddy simulations," *Progress in aerospace sciences*, Vol. 44, No. 6, 2008, pp. 437–446.

[15]Chapman, D. R., "Computational aerodynamics development and outlook," *AIAA journal*, Vol. 17, No. 12, 1979, pp. 1293–1313.

[16]Ashby, S., Beckman, P., Chen, J., Colella, P., Collins, B., Crawford, D., Dongarra, J., Kothe, D., Lusk, R., Messina, P., and Others, "The opportunities and challenges of exascale computing," Tech. rep., Advanced Scientific Computing Advisory Committee (ASCAC) subcommittee at the US Department of Energy Office of Science, 2010.

[17]Langguth, J., Wu, N., Chai, J., and Cai, X., "On the GPU performance of cell-centered finite volume method over unstructured tetrahedral meshes," *Proceedings of 3rd Workshop on Irregular Applications Architectures and Algorithms - IA^3 '13*, ACM Press, Denver, Colorado, 2013.

[18]Witherden, F. D., Farrington, A. M., and Vincent, P. E., "PyFR: An open source framework for solving advection–diffusion type problems on streaming architectures using the flux reconstruction approach," *Computer Physics Communications*, Vol. 185, No. 11, 2014, pp. 3028–3040.

[19]Kogge, P. and Resnick, D. R., "Yearly update: exascale projections for 2013," Tech. rep., Sandia National Laboratories (SNL-NM), Albuquerque, NM (United States), 2013.

[20]NSA, DOE, and other workshop attendees, "U.S. Leadership in High Performance Computing," Technical report, 2016.

[21]Muranushi, T. and Makino, J., "Optimal temporal blocking for stencil computation," *Procedia Computer Science*, Vol. 51, 2015, pp. 1303–1312.

[22]Reed, W. H. and Hill, T., "Triangular mesh methods for the neutron transport equation," *Los Alamos Report LA-UR-73-479*, 1973.

[23]Cockburn, B., Hou, S., and Shu, C.-W., "The Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws. IV. The multidimensional case," *Mathematics of Computation*, Vol. 54, No. 190, 1990, pp. 545–581.

[24]Cockburn, B. and Shu, C.-W., "TVB Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws. II. General framework," *Mathematics of computation*, Vol. 52, No. 186, 1989, pp. 411–435.

[25]Cockburn, B., Lin, S.-Y., and Shu, C.-W., "TVB Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws III: one-dimensional systems," *Journal of Computational Physics*, Vol. 84, No. 1, 1989, pp. 90–113.

[26]Cockburn, B. and Shu, C.-W., "The Runge–Kutta local projection P^1-discontinuous-Galerkin finite element method for scalar conservation laws," *RAIRO-Modélisation mathématique et analyse numérique*, Vol. 25, No. 3, 1991, pp. 337–361.

[27]Cockburn, B. and Shu, C.-W., "The Runge–Kutta discontinuous Galerkin method for conservation laws V: multidimensional systems," *Journal of Computational Physics*, Vol. 141, No. 2, 1998, pp. 199–224.

[28]Kopriva, D. A. and Kolias, J. H., "A conservative staggered-grid Chebyshev multidomain method for compressible flows," *Journal of computational physics*, Vol. 125, No. 1, 1996, pp. 244–261.

[29]Sun, Y., Wang, Z. J., and Liu, Y., "High-order multidomain spectral difference method for the Navier–Stokes equations on unstructured hexahedral grids," *Communications in Computational Physics*, Vol. 2, No. 2, 2007, pp. 310–333.

[30]Huynh, H., "A flux reconstruction approach to high-order schemes including discontinuous Galerkin methods," *AIAA paper*, Vol. 4079, 2007, pp. 2007.

[31]Witherden, F. D., Vermeire, B. C., and Vincent, P. E., "Heterogeneous computing on mixed unstructured grids with PyFR," *Computers & Fluids*, Vol. 120, 2015, pp. 173–186.

[32]Vincent, P., Witherden, F., Vermeire, B., Park, J. S., and Iyer, A., "Towards green aviation with python at petascale," *High Performance Computing, Networking, Storage and Analysis, SC16: International Conference for*, IEEE, 2016, pp. 1–11.

[33]Vincent, P., Witherden, F. D., Farrington, A. M., Ntemos, G., Vermeire, B. C., Park, J. S., and Iyer, A. S., "PyFR: Next-Generation High-Order Computational Fluid Dynamics on Many-Core Hardware," *22nd AIAA Computational Fluid Dynamics Conference*, 2015, p. 3050.

[34]Vermeire, B., Witherden, F., and Vincent, P., "On the utility of GPU accelerated high-order methods for unsteady flow simulations: A comparison with industry-standard tools," *Journal of Computational Physics*, 2017.

[35]Wang, Z. J., Fidkowski, K., Abgrall, R., Bassi, F., Caraeni, D., Cary, A., Deconinck, H., Hartmann, R., Hillewaert, K., Huynh, H. T., et al., "High-order CFD methods: current status and perspective," *International Journal for Numerical Methods in Fluids*, Vol. 72, No. 8, 2013, pp. 811–845.

[36]Persson, P.-O., "A sparse and high-order accurate line-based discontinuous Galerkin method for unstructured meshes," *Journal of Computational Physics*, Vol. 233, 2013, pp. 414–429.

[37]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. 2-3, 1997, pp. 151–167.

[38]Christopher A, K. and Mark H, C., "Additive Runge–Kutta schemes for convection-diffusion-reaction equations," Tech. rep., NASA Langley Technical Report, 2001.

[39]Persson, P.-O., "High-order LES simulations using implicit-explicit Runge–Kutta schemes," *49th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, p. 684.

[40]Vermeire, B. C. and Nadarajah, S., "Adaptive IMEX schemes for high-order unstructured methods," *Journal of Computational Physics*, Vol. 280, 2015, pp. 261–286.

[41]Cockburn, B. and Gopalakrishnan, J., "A characterization of hybridized mixed methods for second order elliptic problems," *SIAM Journal on Numerical Analysis*, Vol. 42, No. 1, 2004, pp. 283–301.

[42]Cockburn, B., Gopalakrishnan, J., and Lazarov, R., "Unified hybridization of discontinuous Galerkin, mixed, and continuous Galerkin methods for second order elliptic problems," *SIAM Journal on Numerical Analysis*, Vol. 47, No. 2, 2009, pp. 1319–1365.

[43]Fernandez, P., Nguyen, C., Roca, X., and Peraire, J., "Implicit large-eddy simulation of compressible flows using the Interior Embedded Discontinuous Galerkin method," *54th AIAA Aerospace Sciences Meeting*, 2016, p. 1332.

[44]Hindenlang, F., Bolemann, T., and Munz, C.-D., "Mesh curving techniques for high order discontinuous Galerkin simulations," *IDIHOM: Industrialization of High-Order Methods-A Top-Down Approach*, Springer, 2015, pp. 133–152.

[45]Turner, M., Peiró, J., and Moxey, D., "A variational framework for high-order mesh generation," *Procedia Engineering*, Vol. 163, 2016, pp. 340–352.

[46]Karman, S. L., Erwin, J. T., Glasby, R. S., and Stefanski, D., "High-Order Mesh Curving Using WCN Mesh Optimization," *46th AIAA Fluid Dynamics Conference*, 2016, p. 3178.

[47]Ims, J., Duan, Z., and Wang, Z. J., "meshcurve: An automated low-order to high-order mesh generator," *22nd AIAA Computational Fluid Dynamics conference*, 2015, p. 2293.

[48]Berger, M. J. and Jameson, A., "Automatic adaptive grid refinement for the Euler equations," *AIAA journal*, Vol. 23, No. 4, 1985, pp. 561–568.

[49]Qin, Y., Shih, T., Keller, P., Sun, R., Hernandez, E., Perng, C., Trigui, N., Han, Z., Shen, F., and Shieh, T., "Estimating grid-induced errors in CFD by discrete-error-transport equations," *42nd AIAA Aerospace Sciences Meeting and Exhibit*, p. 656.

[50]Shih, T., "Estimating Grid-Induced," *Frontiers of Computational Fluid Dynamics 2006*, 2005, pp. 183.

[51]Banks, J., Hittinger, J., Connors, J., and Woodward, C., "Numerical error estimation for nonlinear hyperbolic PDEs via nonlinear error transport," *Computer Methods in Applied Mechanics and Engineering*, Vol. 213, 2012, pp. 1–15.

[52]Venditti, D. A. and Darmofal, D. L., "Grid adaptation for functional outputs: application to two-dimensional inviscid flows," *Journal of Computational Physics*, Vol. 176, No. 1, 2002, pp. 40–69.

[53]Fidkowski, K. J. and Roe, P. L., "An entropy adjoint approach to mesh refinement," *SIAM Journal on Scientific Computing*, Vol. 32, No. 3, 2010, pp. 1261–1287.

[54]Fidkowski, K. J. and Luo, Y., "Output-based space–time mesh adaptation for the compressible Navier–Stokes equations," *Journal of Computational Physics*, Vol. 230, No. 14, 2011, pp. 5753–5773.

[55]Rowley, C. W., Mezić, I., Bagheri, S., Schlatter, P., and Henningson, D. S., "Spectral analysis of nonlinear flows," *Journal of fluid mechanics*, Vol. 641, 2009, pp. 115–127.

[56]Tu, J. H., Rowley, C. W., Luchtenburg, D. M., Brunton, S. L., and Kutz, J. N., "On dynamic mode decomposition: theory and applications," *arXiv preprint arXiv:1312.0041*, 2013.

[57]Brunton, S. L., Proctor, J. L., and Kutz, J. N., "Discovering governing equations from data by sparse identification of nonlinear dynamical systems," *Proceedings of the National Academy of Sciences*, Vol. 113, No. 15, 2016, pp. 3932–3937.

[58]Rudy, S. H., Brunton, S. L., Proctor, J. L., and Kutz, J. N., "Data-driven discovery of partial differential equations," *Science Advances*, Vol. 3, No. 4, 2017, pp. e1602614.

[59]Ling, J. and Templeton, J., "Evaluation of machine learning algorithms for prediction of regions of high Reynolds averaged Navier Stokes uncertainty," *Physics of Fluids*, Vol. 27, No. 8, 2015, pp. 085103.

[60]Ling, J., Kurzawski, A., and Templeton, J., "Reynolds averaged turbulence modelling using deep neural networks with embedded invariance," *Journal of Fluid Mechanics*, Vol. 807, 2016, pp. 155–166.

[61]Jasak, H., "OpenFOAM: open source CFD in research and industry," *International Journal of Naval Architecture and Ocean Engineering*, Vol. 1, No. 2, 2009, pp. 89–94.

[62]Palacios, F., Alonso, J., Duraisamy, K., Colonno, M., Hicken, J., Aranake, A., Campos, A., Copeland, S., Economon, T., Lonkar, A., et al., "Stanford University Unstructured (SU 2): an open-source integrated computational environment for multi-physics simulation and design," *51st AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, 2013, p. 287.

[63]Käser, M., Castro, C., Hermann, V., and Pelties, C., "Seissol–a software for seismic wave propagation simulations," *High Performance Computing in Science and Engineering, Garching/Munich 2009*, 2010, pp. 281–292.

[64]Mocz, P., Vogelsberger, M., Sijacki, D., Pakmor, R., and Hernquist, L., "A discontinuous Galerkin method for solving the fluid and magnetohydrodynamic equations in astrophysical simulations," *Monthly Notices of the Royal Astronomical Society*, 2013.

[65]Schaal, K., Bauer, A., Chandrashekar, P., Pakmor, R., Klingenberg, C., and Springel, V., "Astrophysical hydrodynamics with a high-order discontinuous Galerkin scheme and adaptive mesh refinement," *Monthly Notices of the Royal Astronomical Society*, Vol. 453, No. 4, 2015, pp. 4278–4300.

[66]Iori, F., Grechy, L., Corbett, R., Gedroyc, W., Duncan, N., Caro, C., and Vincent, P., "The effect of in-plane arterial curvature on blood flow and oxygen transport in arterio-venous fistulae," *Physics of Fluids*, Vol. 27, No. 3, 2015, pp. 031903.