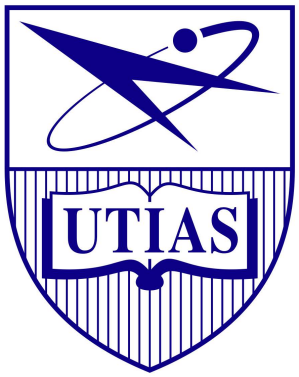


Development of h-p adjoint-based error estimation for LES of reactive flows

Christopher Ngigi

University of Toronto Institute for Aerospace Studies
4925 Dufferin Street, Toronto, Ontario, M3H 5T6, Canada

Doctoral Examination Committee I Report

April 2015

1 Introduction and Motivation

Computational Fluid Dynamics (CFD) has been developed to reduce the time and cost of prototypes in fluid flow experiments. Typical product life-cycles from conception to testing involve numerous design iterations and modifications. From the mid 80s to 90s the stability and reliability of CFD algorithms was improved and CFD's present role has grown to include a vast spectrum of modern day industries. Gordon Moore predicted that computing power would double approximately every two years [1]. Modern day massively-paralleled computer systems have brought computing power to the peta- and exa-scale levels.

Real fluid flows almost always involve turbulence, and CFD methods and models have been developed to capture this phenomenon to varying extents of accuracy. Three main approaches exist: Reynolds averaged Navier Stokes (RANS), which uses full modeling of turbulence phenomenon; large eddy simulations (LES) do direct solutions of Turbulence up to some intermediate Taylor scales while modeling the smaller scales; direct numerical simulation (DNS) performs full resolution and calculation of the turbulence without any modelling. Supercomputers have been used within academic and research lab collaborations for the DNS of turbulence in a box such as that by Kaneda and Ishihara [2], but such calculations are limited to low Reynolds numbers and for simple cases. For more complicated analysis, LES is more practical and achievable, such as those involving combustion simulations.

Such a simulation would need to be as accurate as possible. To improve accuracy, so that CFD results can be closer to those from experiment, we need to reduce error. Two key sources of error within CFD include the discretization of the governing equations (or the general numerical scheme) - which are typically composed in the partial differential equation (PDE) form, and the modeling of the geometry, i.e the mesh resolution. Using high-order discretization schemes and adjoint-based error estimation can prove a more effective way for mesh adaptation since the high-order approach provides an improved accuracy of the solution. Combining this with Adjoint-Based Error Estimation will allow for an optimal technique for mesh refinement. A more common alternative is the gradient (physics) -based approach, but this technique is actually less efficient, in that it requires new sensitivity evaluations for each new parameter of interest. More on gradient-based methods will be discussed in section 4.

2 Scope of Research

This work proposes a two-pronged technique to reduce the numerical error arising from discretization of both the governing equations and the computational domain (mesh) involving implementation of high order finite volume method (FVM) as well as adjoint-based error estimation to compute sensitivity of the output to a change in a given input. The results will be later used for mesh refinement criteria instead of a physics or gradient-based evaluation.

The approach to be used will be to first implement the adjoint solver within the group CFFC Code as modified by Northrup [3], and validate it on already known inviscid (Euler) laminar steady solutions,

and then implement it for unsteady cases, and then introduce viscosity and the temporal variation. Once the Adjoint solver works, the next step will be to advance the work done by Deconinck to test and implement appropriate LES Explicit filters [4] to the high-order FVM which, for simulation of reacting flows, uses the Favre-averaged Navier Stokes (FANS) equations and the central essentially non-oscillatory Scheme (CENO). The application of interest is the improved simulation of turbulent premixed flames while reducing the computational costs. A benefits analysis will then be performed upon conclusion.

The CFFC code already has most of the needed formulations required for this work. Ivan and Groth implemented CENO [[5, 6]], 2-D block-based adaptive mesh refinement (AMR) has been materialized by Zhang and Groth [7], and extended to 3-D by Williamschen and Groth [8], as well as Freret and Groth [9]. Northrup and Groth [3] implemented an iterative GMRES solver for CFFC which relieves the code from the Courant-Friedric-Levy (CFL) condition constraints.

3 Proposed Adaptive Mesh Refinement Strategy

The planned approach to perform the Adjoint-based Error Estimation (covered in Section 4) will utilize the following techniques

3.1 Large Eddy Simulation (LES)

The level of fidelity of LES lies between the fully resolved Direct Numerical Simulation (DNS) and the fully modelled Reynolds-Averaged Navier Stokes (RANS) methods. LES will applies a filter size to the turbulence scales, resolving the larger ones, and modelling the smaller ones (i.e. those smaller than the filter size). Flow variables need to be decomposed into filtered(resolved) and residual (modelled or Sub-Filter Scale (SFS))components via a spatial filtering procedure. Consequently, the Navier Stokes Equations (Section 9.1) need to be Favre-filtered (Section 9.2) before being used in LES.

3.1.1 Explicit Filtering

In LES, a filter is applied to separate the resolved scales from the modelled scales. The governing equations (Compressible form of the Navier Stokes) are filtered via a spatial filtering procedure.

As described by H. Perez[10], two main techniques for Filtering are used: The first is Implicit Filtering, and the second is Explicit Filtering. For Implicit Filtering, filtering occurs indirectly as an effect of the computational mesh size and the discretization operators. The spatial discretization schemes have a dissipation property that depend on the wave-number of the scales. Thus the filter width is inherently related to mesh spacing. That means that for variously refined meshes running the same LES case, then the results may not indicate convergence for a desired output: the finer the mesh, the more the resolved scales, thus making it difficult to distinguish effects of added scales in comparison to that of improved refinement.

For Explicit filtering however, the filter size is fixed for the entire grid, regardless of the levels of mesh refinement that are in effect. Convergence can then be easier to monitor, and results could be compared to DNS results if required. In order to be able to manipulate the Navier-Stokes equations after applying a filter, we require that filters possess the following three properties: Conservation of constants, commutation with addition, and commutation with differentiation values.

3.1.2 Aliasing Errors

Favre filtering of the convective term in the Navier-Stokes equation (see section 9.2) results in the non-linear correlation

$$\frac{\partial}{\partial x_{\mathrm{j}}}(\overline{\rho u_{\mathrm{i}} u_{\mathrm{j}}}) \,, \tag{3.1.1}$$

creating a closure problem. This term is typically treated by computing the product of the closed filtered velocities and modelling the remainder which essentially transfers the closure problem to the right-hand side of the filtered Navier-Stokes equation:

$$\overline{\rho u_{\mathrm{i}} u_{\mathrm{j}}} = \bar{\rho}\tilde{u}_{\mathrm{i}}\tilde{u}_{\mathrm{j}} + \underbrace{(\overline{\rho u_{\mathrm{i}} u_{\mathrm{j}}} - \bar{\rho}\tilde{u}_{\mathrm{i}}\tilde{u}_{\mathrm{j}})}_{\sigma_{\mathrm{ij}}} \,, \tag{3.1.2}$$

where σ_{ij} is the SFS stress tensor. The problem with this decomposition is that the non-linear product $\tilde{u}_{\mathrm{i}}\tilde{u}_{\mathrm{j}}$ generates frequencies beyond the characteristic frequency that defines $\tilde{u}_{\mathrm{i}}$. These frequencies alias back as resolved and act as fictitious stresses [11]. The resulting errors are difficult to eliminate and control if implicit filtering is used. Please refer to the work by H. Perez [10] for additional details.

3.1.3 Commutation Errors

For inhomogeneous turbulent flows, the smallest resolvable eddy sizes differ in regions of the flow. Usually, flow close to solid walls produces smaller eddies due to wall damping effects. This implies that the mesh should be refined to resolve these scales in the near-wall region. To ensure the structure of filtered equations remains unchanged before and after filtering, the filtering operation should commute with the differential operation, and this is described by H. Perez [10] as follows:

$$\overline{\frac{\mathrm{d}\phi}{\mathrm{d}x}} = \frac{\mathrm{d}\bar{\phi}}{dx} \,. \tag{3.1.3}$$

In general, filters do not commute when variable filter width is used. To be acceptable, the errors associated with the commutation properties of the filter should be of the same order as the truncation errors associated with the numerical scheme, i.e.,

$$\left[\frac{\mathrm{d}\phi}{\mathrm{d}x}\right] \equiv \left|\overline{\frac{\mathrm{d}\phi}{\mathrm{d}x}} - \frac{\mathrm{d}\bar{\phi}}{\mathrm{d}x}\right| = \mathcal{O}(\Delta^n) \,, \tag{3.1.4}$$

where n is the order of the filter, which should be at least equal to the order of accuracy of the spatial discretization scheme.

3.1.4 Sub-Filter Scale Modelling

Because of the closure problem, we need to model the necessary Sub Filter Scale terms, as described in the Appendix, section 9.2. The unresolved scales need to be modelled, since they have a pronounced effect on the resolved scales. The Energy Cascade, first described by Richardson in 1922, explains how the Turbulent Kinetic Energy is mainly contained in the larger scales, and dissipated by the smallest scales via viscosity.[12].

A number of approaches have been used to model the SFS terms. One of the first was by Smagorinsky [13] which models SFS stress tensors and the Reynolds Stress tensor. This approach also defines a Sub Filter Scale Eddy Viscosity which is in turn used to relate SFS stresses to the filtered Strain rates, based on the assumption that the SFS stress tensor behaves similar to the Viscous stress tensor, itself proportional to the Strain rate tensor. The model goes ahead to express the SFS Eddy Viscosity as proportional to the magnitude of the filtered Strain rate tensor. The constant of proportionality is termed the Smagorinsky Coefficient.

The dynamic Smagorinsky model is a modification of the Smagorinsky model. It observes that the Smagorinsky Coefficient is only valid for an optimized flow regime. Thus a variable coefficient is introduced. This will likely be the approach of choice for this research.

3.2 The Finite Volume Method

Godunov-type Finite Volume Methods (FVM) apply a cell-centered spatial discretization resulting in the solution domain split into control volumes [14]. The integral form of the governing equations is applied. The Navier Stokes and Favre-Averaged Navier Stokes are described in the Appendix, see Section 9.1 and 9.2.

The Favre-filtered form of the conservation equations for mass, momentum, total energy, and species mass fractions, along with the equation of state are used here in the LES of turbulent reactive flows and can be re-expressed in the following general weak conservation form using matrix-vector notation [15]

$$\frac{\partial \overline{\mathbf{U}}}{\partial t} + \vec{\nabla} \cdot \vec{\mathcal{F}} = \frac{\partial \overline{\mathbf{U}}}{\partial t} + \vec{\nabla} \cdot \vec{\mathcal{F}}^{\mathrm{I}} \left(\overline{\mathbf{U}} \right) - \vec{\nabla} \cdot \vec{\mathcal{F}}^{\mathrm{V}} \left(\overline{\mathbf{U}}, \vec{\nabla} \overline{\mathbf{U}} \right) = \overline{\mathbf{S}} \tag{3.2.1}$$

where $\overline{\mathbf{U}}$ is the vector of conserved solution variables and $\vec{\mathcal{F}}$ is the solution flux dyad. The flux dyad can be decomposed into two components and written as $\vec{\mathcal{F}} = \vec{\mathcal{F}}^{\mathrm{I}} - \vec{\mathcal{F}}^{\mathrm{V}}$ where $\vec{\mathcal{F}}^{\mathrm{I}} = \vec{\mathcal{F}}^{\mathrm{I}}(\overline{\mathbf{U}})$ contains the hyperbolic or inviscid components of the solution fluxes $\vec{\mathcal{F}}^{\mathrm{V}} = \vec{\mathcal{F}}^{\mathrm{V}}(\overline{\mathbf{U}}, \vec{\nabla}\overline{\mathbf{U}})$ contains the elliptic or viscous components of the fluxes.

A very important key in FVM is how these fluxes are resolved. The common approach is to evaluate inviscid fluxes within each cell (control volume) via a solution to a Riemann Problem, starting from initial

value left and right states for each cell. The goal is to obtain the final cell-centered, cell-average value.

A semi-discrete form of the governing equations can be derived from the application of the finite-volume method to the integral form of Eq. (3.2.1) for cell (i, j, k) of a three-dimensional multi-block mesh composed of hexahedral volume elements. Using a N_G-point Gaussian quadrature numerical integration procedure to evaluate the solution flux along each of the N_f faces of the cell, the following semi-discrete form is obtained:

$$\frac{\mathrm{d}\overline{\mathbf{U}}_{ijk}}{\mathrm{d}t} = -\frac{1}{V_{ijk}} \sum_{l=1}^{N_f} \sum_{m=1}^{N_{GF}} \left(\omega_m \left(\vec{\mathcal{F}}^{\mathrm{I}} - \vec{\mathcal{F}}^{\mathrm{V}} \right) \cdot \hat{n} A \right)_{ijk,l,m} + \sum_{n=1}^{N_{GV}} \left(\omega_n \mathbf{S} \right)_{i,j,k,n} = \overline{\mathbf{R}}_{ijk} \left(\overline{\mathbf{U}} \right), \tag{3.2.2}$$

where ω_m are the face quadrature weighting coefficients, ω_n are the volumetric quadrature weighting coefficients, A_l denotes the surface area of face l, and $\overline{\mathbf{R}}_{ijk}$ is the residual operator. After the evaluation of $\overline{\mathbf{R}}$, one can advance the solution in time, and therefore iteratively solve the time dependent problem that is described by the equations. Quadrature rules are used to determine the Flux evaluation points on the surface, N_{GF} and those for the volume, N_{GV}. The remaining procedure is:

- Apply Solution Reconstruction (see 3.2.1);
- Evaluate the Inviscid and Viscous Fluxes and apply the weights to respective Gauss-Legendre points;
- Evaluate the Source Vector, which adds effects of turbulence and chemistry in reacting flows;
- Apply an appropriate time-marching scheme, such as a fourth-order Runge-Kutta (RK4) suitable for High-Order methods.

3.2.1 The Centrally Essentially Non-Oscillatory (CENO) Scheme

In solution reconstruction, given the cell average values, a Taylor series expansion polynomial is defined to represent the variation of the solution within the particular cell and the average solution of its neighbors.

The High-Order CENO reconstruction scheme of Ivan and Groth [[5], [6]] uses a k-exact least-squares reconstruction technique, essentially a k^{th} order Taylor series expansion of solution variable U about the cell center. This reconstruction techniques is applied to a monotonicity-preserving limited linear scheme. For Favre-Averaged Navier Stokes (FANS), k corresponds to the spatial accuracy of the scheme. A smoothness indicator is used to switch between reconstruction procedures.

Some of the advantages of CENO are [16]:

- It provides accuracy of other Essentially Non-Oscillatory (ENO) schemes and maintains monotonicity near discontinuities;
- Uses the same central stencil while avoiding complexities of other ENO schemes [5, 6];
- Can be extended to unstructured meshes and multiple dimensions [17];

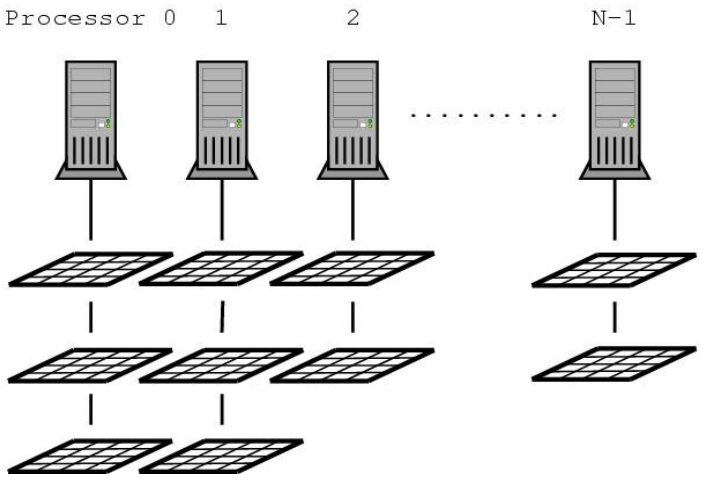


Figure 1: Parallel Implementation of Block-based AMR [3]

- Identifies regions where under-resolution and non-smooth data occurs, and this could prove useful for mesh adaptation techniques.

High-Order generally refers to $k > 2$. These kind of methods generally have less numerical dissipation, require fewer mesh points to achieve solution accuracy, (as compared to a 2^{nd} Order method, for example)

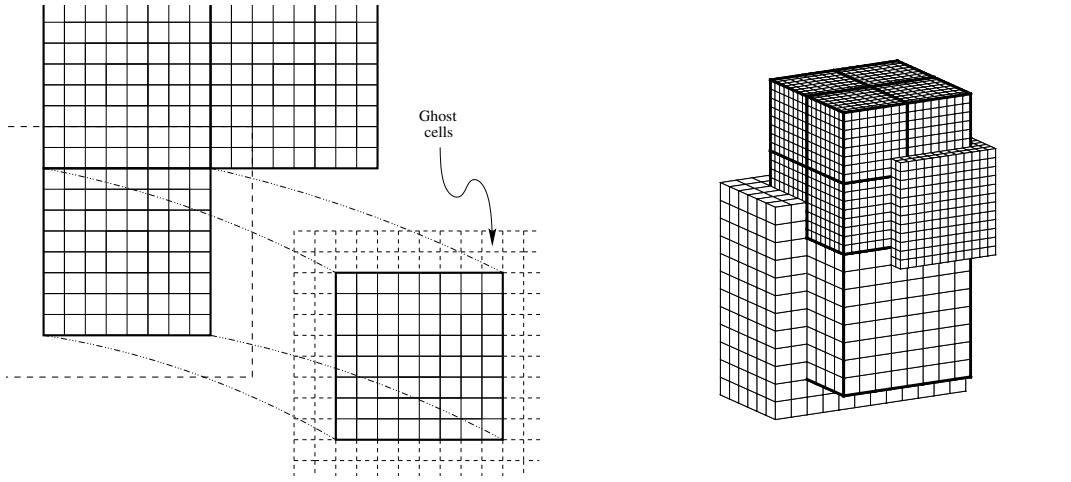
3.3 Block-Based Adaptive Mesh Refinement (AMR)

Having discretized the governing equations, the geometry similarly needs to be discretized into unique cells, in a process called meshing. For increase in accuracy, a finer mesh resolution (smaller grid size) will be required.

Adaptive Mesh Refinement (AMR) takes its name from the variational increase in localized grid resolution to allow for improved accuracy at without carrying out the same at a ubiquitous scale. Global refinement is often cheaper to perform, whereas localized (Adaptive) is not; however, the computational cost for refining many more cells and storing this information makes it more expensive, in spite of the fact that a well designed approach selection will need to be programmed to perform an adaptive refinement.

Block-based AMR groups cells into blocks, making it an optimal configuration for parallel systems, where each processor could have a certain number of Blocks, and protocols such as the Message Passing Interface (MPI) could be used for Block-Block inter-communication. Communication between Blocks occurs via Ghost cells, and in case neighboring blocks have a difference in mesh resolution, then Mutligrid-type restriction and prolongation is used.

Groth *et al* [18] developed some of the initial work with Block-Based AMR for Computational Magnetohydrodynamics, and was extended to enable parallel implementation for a variety of complex flows. Northrup and Groth [19] implemented a fully-implicit, parallel Newton-Krylov method, and applies Schwarz preconditioning to take advantage of the Block structure of the computational grid. Within



(a) 2D Block-Ghost configuration [3] (b) 3D Block-Ghost configuration [3]

Figure 2: Block structure depicting interior cells and ghost cell configurations

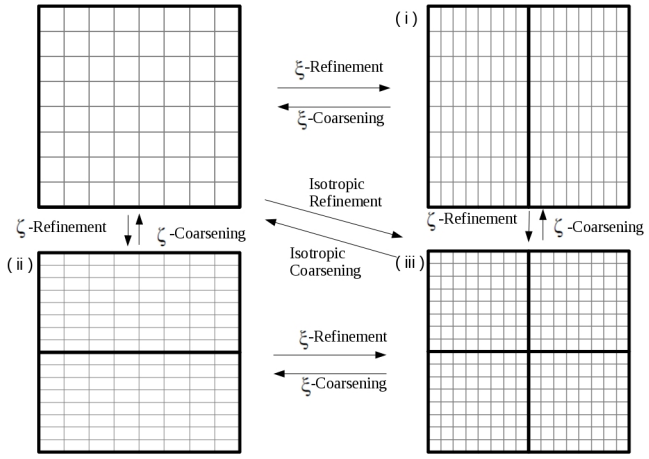


Figure 3: Isotropic and Anisotropic AMR [20]

Block-based AMR, the cell-connectivity is stored in a tree structure, which will typically be an octree in a 3D domain. Block-Based AMR is more optimized than a refinement performed on a cell-by-cell basis, as the information storage process for the latter is computationally expensive: The tree approach is much simpler, faster and cheaper. Blocks flagged for increased resolution can be refined using two approaches:

1. **Isotropic AMR**:

   Isotropic AMR occurs by uniformly splitting the Parent cell into 8 children (in a 3D domain) or into 4 children (in a 2D domain). This considers that the physics is occurring without a preference for a given direction, and usually results in a larger-than-desired cell count.

2. **Anisotropic AMR**:

   For flows having a directional bias, e.g. in the simulation of a jet exiting a nozzle, the dominant

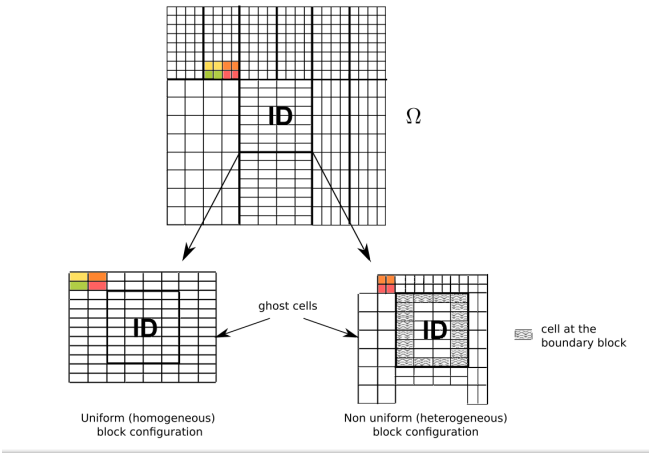


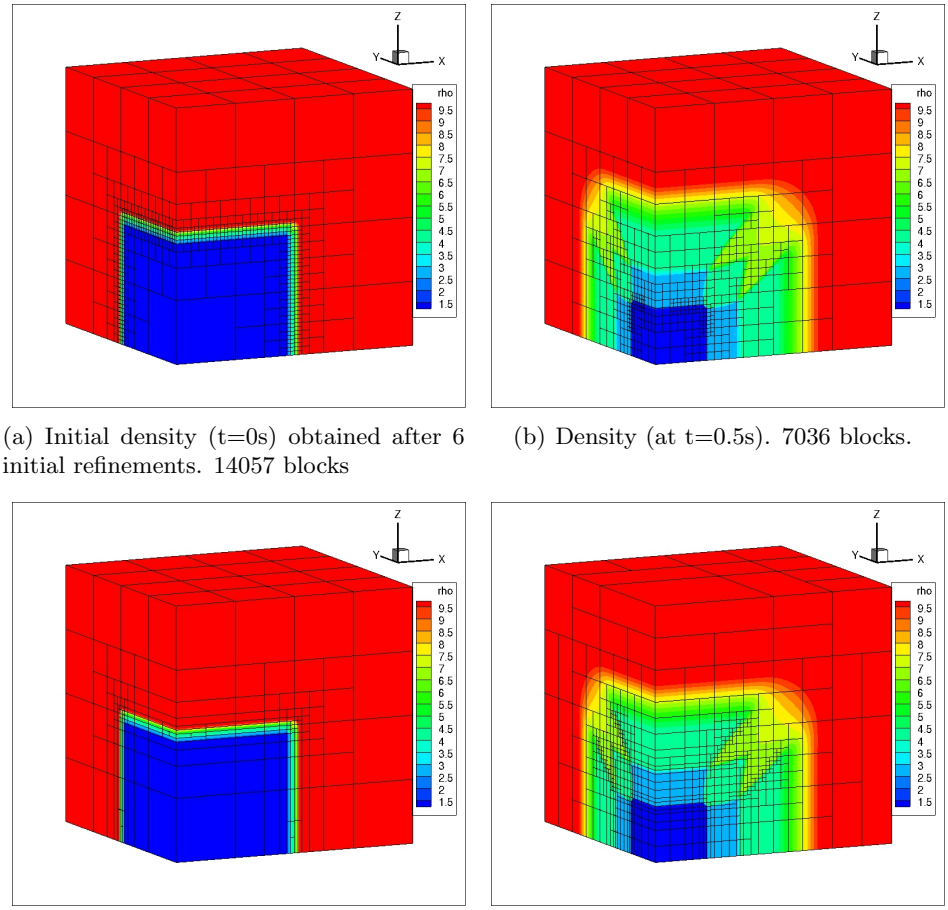
Figure 4: Non-Uniform Ghost Cells on Block [Freret and Groth: 2015] [9]

direction will be along the axis of symmetry of the nozzle. Therefore, a priori, we can tell that the mesh cells would ideally need to be stretched along this same axis. Whenever we have this kind of bias on the cell geometry, we have anisotropy. Mesh refinement occurs in a slightly different manner, as now we switch to a binary tree system, which stores refinement history and cell connectivity. Two, four, or eight refined blocks can be created, resulting in five distinct possible refinements for a single block to better adapt to evolving flow features. Studies by Zhang [20], Zhang and Groth [7], Freret and Groth [9] indicate that anisotropic AMR for given simulations result in lower cell counts (see figure 5), although the refinement procedure is more difficult than that of the Isotropic AMR. The number of cells to reach a convergence criteria is reduced in the case of figure 6 by a factor of approximately 1.5.

Freret and Groth [9] have completed a new formulation for a Non-uniform block-based approach that targets the the ghost cells. In this method, a block will copy over the mesh resolution and solution information of its neighboring block to its own ghost cells, and this eliminates the interpolation error within a Uniform block-based scheme that would otherwise have resulted in necessary solution restriction and prolongation (if the mesh discretization levels were different). Further, this saves the computational cost of flux correction cycles, done via message passing between the blocks, as described by Zhang to be a temporary remedy to ghost-cell evaluation.[7]

3.4 Combustion Modelling via the Presumed Conditional Moment and Flame Propagation for Intrinsic Low Dimensional Manifold (PCM-FPI) Model

When performing LES of turbulent reactive flows, there is need to predict the chemical reaction rates. Implementation of the PCM-FPI model in LES within the CFD Group was carried out by Hernández-Pérez [10] and Shahbazian *et al* [21].



(a) Initial density (t=0s) obtained after 6 initial refinements. 14057 blocks

(b) Density (at t=0.5s). 7036 blocks.

(c) Initial density (t=0s) obtained after 6 initial refinements. 1089 blocks.

(d) Density (at t=0.5s). 5522 blocks

Figure 5: Differences between the isotropic and anisotropic AMR on a shockcube problem. Initial conditions have large density jumps between left and right states: 1 block = 8x8x8 cells. [Freret and Groth: 2015] [9]

The FPI model was proposed to build databases based on detailed simulations of simple flames. The basis of tabulations is the steady-state one-dimensional laminar flame, which are solved using CANTERA software. These flame quanitities are stored in a table that is then read by the CFFC program. The main objective of the FPI tabulation technique is to reduce the cost of performing reactive flow computations with large detailed chemical kinetic mechanisms, but still retain the accuracy of detailed results, by building databases of relevant quantities based on detailed simulations of simple flames. [10]

The PCM technique uses a statistical approach and utilizes Probability Density Functions (PDFs). The approach presumes a PDF-like distribution of a subfilter-scale fluctuating quantity. The PDF subfilter is incorporated into the Favre-filtered reaction rates for species.

Combining PCM with FPI allows an approach to model complex chemistry from tabulated data, is a very general model that can be used for premixed, non-premixed and partially-premixed flames. e.g. In the event that turbulent premixed combustion is the focus of the analysis, look up tables of filtered terms are built up from laminar premixed flamelet data.

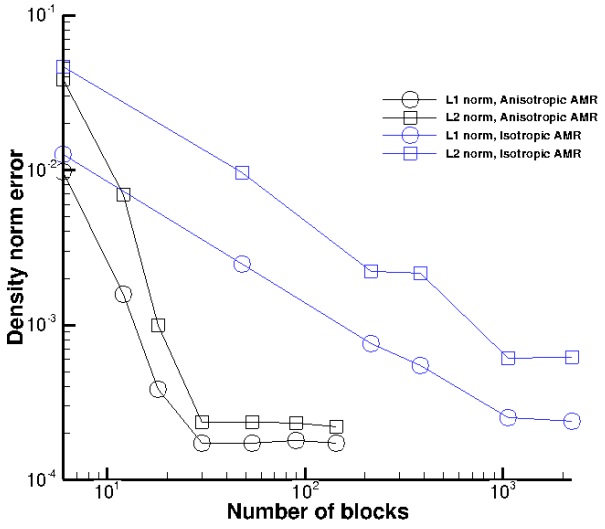


Figure 6: Resulting error in the density norm [Freret and Groth: 2015] [9]

4 Adjoint-Based Method for Error Estimation

4.1 Introduction

In Gradient-based Mesh adaptation techniques, emphasis is placed on the change of values of the Solution variables across cells, and any high rates of change require a mesh refinement suitable enough to capture a smoother transition, e.g. across a shock wave that forms on the upper surface of an airfoil in transonic regime. In this scenario, a quantity such as density or pressure could be monitored.

Adjoint-based Error Estimation focuses on the *sensitivities*, whereby some output of interest (henceforth termed as a *functional*), e.g. Lift on an airfoil, is sensitive to the mesh refinement levels upstream of the airfoil along the chord line. The Adjoint approach is a more efficient, albeit expensive, criteria for mesh refinement: only one calculation for the sensitivity needs to be calculated, whereas for gradient based approaches, each quantity will require a unique calculation.

Both Adjoint and Gradient-Based approaches apply a *posteriori* technique, in that an initial solution, known as the *Primal solution*, needs to first be evaluated - no refinement criteria can be carried out until the primal solution is established. Once this is met, the residual values can be weighted with the values of the adjoint, and this could be used as a mesh refinement criteria. Some of the limitations of the gradient-based approach as discussed by Giles and Pierce [22] include are that the gradient-based approach can only deal with continuous functionals as opposed to discrete optimization functional, as well as its inability to deal with functions that have multiple minima. In this latter case, the gradient-based technique will generally converge to the nearest local minima, whose value may not represent overall system minimum.

Other key research on adjoints applied to aerodynamic flows was performed by Giles and Pierce [22],

Becker and Rannacher [23] and Venditti and Darmofal [24], with a key summary done by Fidkowski and Darmofal [25].

4.2 Derivation

If $\mathbf{R}$ is the set of all residuals for all cells in the domain, then the systems of equations can be written as:

$$\mathbf{R}(\mathbf{U}) = 0 \tag{4.2.1}$$

Considering a scalar output of interest [26], say, J, we can define it such that:

$$J = J(U) \tag{4.2.2}$$

where U is a vector containing the solution variable. We define a discrete Adjoint, $\Psi \in \mathbb{R}^N$ as a vector of sensitivities of the output to the N residuals. Each entry of the adjoint essentially tells us the effect that a perturbation in the corresponding entry within the residual vector would have on output J. Consider the case fo a residual perturbation due to a change to the input parameter, μ, and $\mu \in \mathbb{R}^{N_p}$. A local sensitivity analysis can be applied as follows:

$$\underbrace{\mu}_{\text{inputs} \in \mathbb{R}^{N_\mu}} \rightarrow \underbrace{\mathbf{R}(\mathbf{U}, \mu) = 0}_{N \text{ equations}} \rightarrow \underbrace{\mathbf{U}}_{\text{state} \in \mathbb{R}^N} \rightarrow \underbrace{J(\mathbf{U})}_{\text{output(scalar)}} \tag{4.2.3}$$

To monitor the change of J with μ,

$$\frac{dJ}{d\mu} \in \mathbb{R}^{1 \times N_\mu} = N_\mu \text{ sensitivities} \tag{4.2.4}$$

recalling $\mu \in \mathbb{R}^{N_p}$. If J depended directly on μ then we would have had $\frac{dJ}{d\mu}$, but in the scenario we consider the case that $J = J(U)$ alone. To evaluate the N_μ sensitivities we could use: *finite differencing* where the inputs are perturbed one at a time; *forward linearization* where the sequence of operations in Equation (4.2.3) are linearized; and, lastly, the *adjoint approach* which requires an inexpensive residual perturbation calculation, followed by an adjoint weighting to compute the effect on the output:

$$\frac{dJ}{d\mu} = \Psi^T \frac{\partial R}{\partial \mu} \tag{4.2.5}$$

One of the key ideas behind the adjoint approach is that the forward problem need be solved only once to evaluate a sensitivity. Given a particular input, μ, we could solve the system to find $\mathbf{U}$ such that $\mathbf{R(U,}\mu = 0$. Once we perturb $\mu \rightarrow \mu + \delta\mu$, to find the effect on J, we would otherwise need to re-solve the discretized system, which would be an expensive step. The Adjoint, on the other hand, precomputes the eddect of $\mathbf{R}$ on J. The resulting N sensitiviteis are stored in vector Ψ.

Consider the chain of events in computing sensitivities (i.e. the effects of a small perturbance of the

input, $\delta\mu$) via a direct approach:

1. Input: $\mu \rightarrow \mu + \delta\mu$
2. Residual: $\mathbf{R(U,}\ \mu\ +\ \delta\mu) = \delta\mathbf{R} \neq 0 \rightarrow \mathbf{R(U,}\ \mu) + \frac{\partial \mathbf{R}}{\partial \mu}\Big|_{\mathbf{U},\ \mu} \delta\mu = \delta\mathbf{R}$
3. State: $\mathbf{R(U}\ + \delta\mathbf{U,}\ \mu + \delta\mu) = 0 \rightarrow \mathbf{R(U,}\ \mu) + \frac{\partial \mathbf{R}}{\partial \mu}\Big|_{\mathbf{U},\ \mu} \delta\mu + \frac{\partial \mathbf{R}}{\partial \mathbf{U}}\Big|_{\mathbf{U},\ \mu} \delta\mathrm{U} = 0$
4. Output: $J(\mathbf{U} + \delta\mathbf{U}) = J(\mathbf{U}) + \delta J \rightarrow \delta J = \frac{\partial J}{\partial \mathbf{U}} \delta U$

Subtracting step 2 from 3:

$$\begin{aligned} \frac{\partial \mathbf{R}}{\partial \mathbf{U}}\Big|_{\mathbf{U},\ \mu} \delta U &= -\delta R \\ \delta\mathbf{U} &= -\left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right]^{-1} \delta\mathbf{R} \end{aligned} \tag{4.2.6}$$

We combine this to the output linearization in step 4 to give the output perturbation, $\delta\mathbf{U}$ in terms of the residual perturbation, $\delta\mathbf{R}$:

$$\begin{aligned} \delta J &= \frac{\partial J}{\partial U} \delta\mathbf{U} \\ &= \underbrace{\frac{\partial J}{\partial U}\left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right]^{-1}}_{\Psi^T\ \in\ \mathbb{R}^N} \delta\mathbf{R} \end{aligned} \tag{4.2.7}$$

The *Adjoint Equation* is then written as:

$$\left(\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right)\ \Psi^T\ = \frac{\partial J}{\partial \mathbf{U}} \tag{4.2.8}$$

Once we have Ψ, no more solves are required for the system. The calculation of $\frac{\partial \mathbf{R}}{\partial \mu}$ (henceforth called the *Jacobian*) is much cheaper compared to a forward solve. There are four ways to calculate the adjoint equation within CFFC code:

- Complexifying Variables - Calculating deriveatives using complex numbers.
- Finite Differences coupled with Automatic Differentiation Techniques.
- Analytically - by evaluating the exact Jacobian, and this is an expensive, but very accurate, process.
- Approximate Jacobian - This will be the starting approach, and then an evaluation will be made on cost and accuracy. A decision will thereafter be made on whether to use the Analytical approach or not. Northrup [3] implemented the script within the CFFC code that builds part of the structure of the Flux Jacobian matrix.

4.3 Solution of Linear Systems

The Adjoint problem therefore takes the form of a linear system of equations, $Ax = b$, which will be solved utilizing the Trilinos set of packages, written by Sandia National Labs. Trilinos contains very powerful and parallelizable linear algebra solvers. This suite of programs has already been linked to the CFFC code within the SCINET network.

4.4 Use of Solution Error Estimates in Mesh Adaptation

The Adjoint solution will indicate areas of higher sensitivity to given changes in input, and this information could indicate a *posteriori* where the mesh would need to be better refined for higher accuracy.

4.5 Implementing Isotropic Mesh Refinement

Since the Isotropic Adaptive Mesh Refinement (AMR) is easier as an initial implementation, this will be the first step attempted before applying Anisotropic AMR which further decreases the cell counts, for a comparable level of accuracy that can be achieved by Isotropic AMR.

4.5.1 Steady Adjoints

For steady simulations, the solution of the Adjoint is a one time event, and the computational cost is low.

4.5.2 Unsteady Adjoints

The Adjoint solution needs to be calculated every single timestep within an unsteady simulation (e.g. for the goal of the project, which is to simulate Turbulent Premixed Flows). This occcurs via running the simulation forward in time while evaluating all the *Primal* solution values at the different time levels until the final timestep, and then marching backwards in time, and solving the Adjoint at each of those timesteps, and re-meshing as required. An error threshold could be defined prior to the process such that arrival of the solution within a favorable regime could indicate to the automated process a suitable end to the refinement cycle.

The computational cost for this may likely be very high, and perhaps unattainable for practical purposes.

5 Poisson Solver

5.1 Model Problem

Solving a 2D and 3D Poisson equation in C++, using the Trilinos set of libraries.

We seek to solve the Poisson equation

$$-\Delta u = f \text{ in } D, \tag{5.1.1}$$

$$u = 0 \text{ on } \partial D, \tag{5.1.2}$$

- In 2D:
  $D = [0,1]^2$, $f(x,y) = 2(x(1-x) + y(1-y))$ is the source term and $u(x,y)$ is the solution to be computed.

  We consider a finite difference (FD) scheme for solving (5.1.1). The spatial domain D is discretized using a regular grid made of $(N+1)^2$ points $\mathbf{x}_{ij} = (ih, jh)$ with $0 \leq i,j \leq N$, $h = \frac{1}{N}$. We denote by u_{ij} and f_{ij} the approximate solution $u(\mathbf{x}_{ij})$ and $f(\mathbf{x}_{ij})$, respectively.
  Using the centered FD scheme in 2D:

$$-\frac{u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{ij}}{h^2} = f_{ij},$$

  written at each interior node $\mathbf{x}_{ijk}$, $1 \leq i,j,k \leq N-1$, and taking into account the boundary conditions (5.1.2), write down a linear system

$$\mathbf{A}\mathbf{u} = \mathbf{b} \tag{5.1.3}$$

  of size $(N-1)^2 \times (N-1)^2$, where $\mathbf{u}$ is a vector that contains the u_{ij} corresponding to the interior nodes.

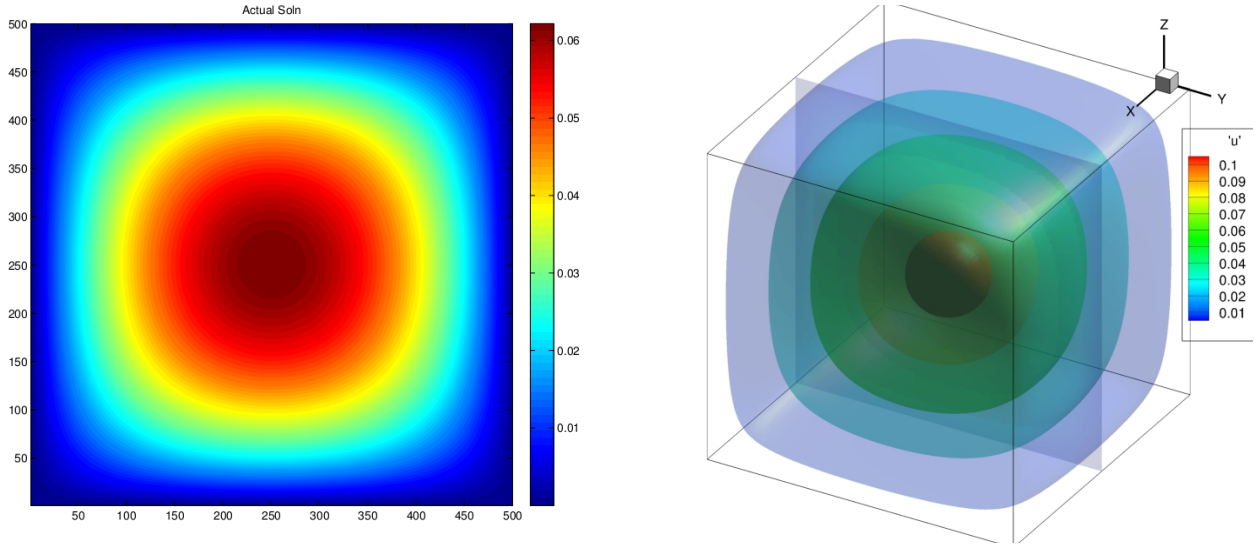
- In 3D:
  $D = [0,1]^3$, $f(x,y,z) = 3(x(1-x) + y(1-y) + z(1-z))$ is the source term and $u(x,y,z)$ is the solution to be computed. We consider a finite difference (FD) scheme for solving (5.1.1). The spatial domain D is discretized using a regular grid made of $(N+1)^2$ points $\mathbf{x}_{ij} = (ih, jh)$ with $0 \leq i,j \leq N$, $h = \frac{1}{N}$. We denote by u_{ij} and f_{ij} the approximate solution $u(\mathbf{x}_{ij})$ and $f(\mathbf{x}_{ij})$, respectively.
  Using the centered FD scheme in 3D:

$$-\frac{u_{i+1,j,k-1}+u_{i-1,j,k-1}+u_{i,j+1,k-1}+u_{i,j-1,k-1}+u_{i+1,j,k+1}+u_{i-1,j,k+1}+u_{i,j+1,k+1}+u_{i,j-1,k+1}-6u_{ijk}}{h^2} = f_{ijk},$$

  written at each interior node $\mathbf{x}_{ijk}$, $1 \leq i,j,k \leq N-1$, and taking into account the boundary conditions (5.1.2), write down a linear system

$$\mathbf{A}\mathbf{u} = \mathbf{b} \tag{5.1.4}$$

  of size $(N-1)^3 \times (N-1)^3$, where $\mathbf{u}$ is a vector that contains the u_{ij} corresponding to the interior nodes.



(a) 2D case on $N^2 = 200^2$

(b) 3D case on $N^3 = 100^3$

Figure 7: Solution Contours for solution of the Poisson Problem

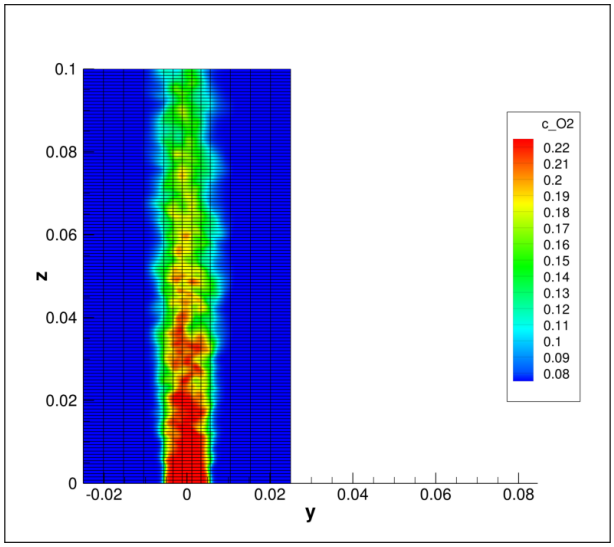


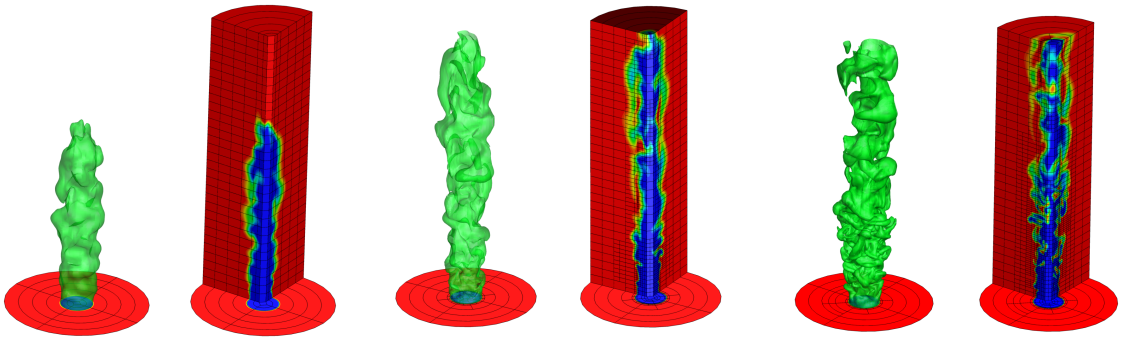
Figure 8: Time averaged O_2 species concentration

6 LES of a Turbulent Premixed Methane Flame

6.1 Model Problem

The CFFC code was used to run a simulation for a Turbulent Premixed Methane flame, using 800 nodes, 3200 blocks and each block having $8^3 = 512$ cells, hence a total of $1,638,400$ cells. Time averaged results for t=6ms, 7ms, 8ms, 9ms, 10ms and 11ms are shown in figure 8, the contours representing the species concentration of Oxygen.

Included here are some pictures from previous CFFC simulations that show the varying mesh cell sizes



(a) At t=2.0ms, 800 (8x8x8) blocks, 410,000 cells (no refinement, 1 mesh level)

(b) At t=4.25ms 5595 (8x8x8) blocks, 2.8 million cells (3 levels of mesh refinement)

(c) At 7.0 ms, 18531 (8x8x8) blocks, 9.5 million cells (3 levels of mesh refinement)

Figure 9: Some figures showing isotropic mesh refinement with 3 levels of refinement for a lean premixed methane air flame in air. LES solutions were obtained with the flame surface density (FSD) model and the refinement was based on temperature gradient.

among the different blocks.

7 Discrete Adjoint Solutions based on a Primal Problem

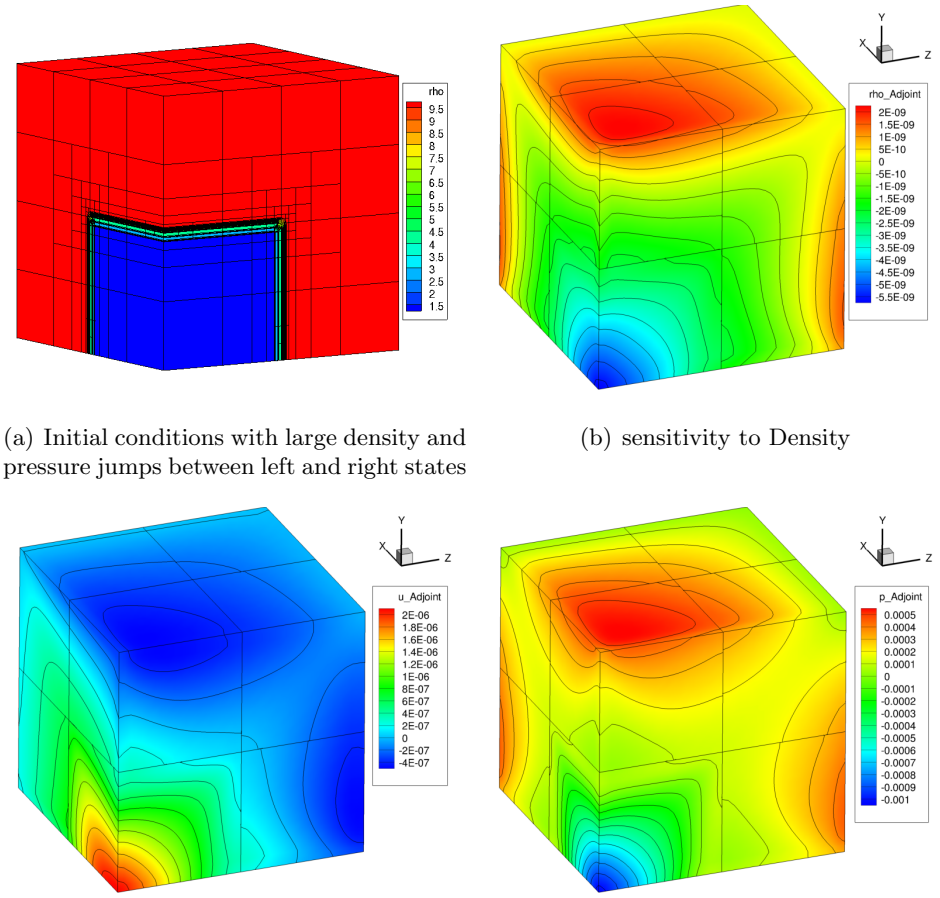
7.1 Background

Following the adjoint formulation explained in section 4 it is seen that the adjoint formulation (also known as the *dual problem*) can be written as a linear system:

$$A^T x = -b^T \tag{7.1.1}$$

Thus we can write the transpose of the Residual Jacobian matrix, $\frac{\partial R}{\partial U}$ as matrix A^T and the right hand side vector b as the derivative of the functional with respect to the state, $\frac{\partial J}{\partial U_i}$

Some current plots obtained for the adjoint sensitivities are shown in figure 10. The initial condition was that of a shock cube problem, and average pressure was selected as the functional, J. This case is still ongoing as the formulation is still being verified.



(a) Initial conditions with large density and pressure jumps between left and right states

(b) sensitivity to Density

(c) sensitivity to x-Momentum

(d) sensitivity to Pressure

Figure 10: Contours of adjoint sensitivities to changes in the conserved state variables

8 Summary of Progress to Date and Future Work

8.1 Progress to Date

Task	Completion Date
Literature Review	September-October 2014
Trelis Meshing Software	November 2014
CFFC Group Code Flux Jacobian Analysis	December 2014
Trilinos Package solution for example Poisson Problem in serial and parallel configurations	December 2014
Running a current-state LES case of a Turbulent Premixed Methane Flame using PCM-FPI to get a threshold estimation of solution run time	January 2015
Implementing the approximate Adjoint Derivative to the Flux Jacobians testing on Euler Equations	April 2015

8.2 Future Work

Task	Completion Date
Extension to Mesh adaptation	May 2015
Application of Adjoint Problem to Navier Stokes	June 2015
Explicit Filters for High Order FVM implementation	October 2015
Conference Paper I draft	November 2015
Coupling of High Order method with Adjoint-based AMR	December 2015
CFD simulation of Cold Flow	January 2016
CFD simulation of Laminar Non-Premixed Flame	February 2016
CFD simulation of Laminar Diffusion Flame	February 2016
Journal Paper I	April 2016
Conference Paper II draft	April 2016
CFD simulation of Turbulent Non-Premixed Flame	May 2016
Journal Paper II	July 2016
Conference Paper I Presentation	July 2016
Conference Paper II draft	October 2016
Journal Paper III	October 2016
CFD simulation of Full Thermo-coupling	October 2016
Conference Paper III draft	November 2016
Thesis write-up	September 2017

9 Appendix

9.1 Navier Stokes Equations

The conservation equations for a thermally perfect reactive mixture of N chemical species evolving in time, t, and space, $\mathbf{x}$, can then be written using tensor notation as [10]

Conservation of Mass:

$$\frac{\partial \rho}{\partial t} + \frac{\partial (\rho u_j)}{\partial x_j} = 0\,, \tag{9.1.1}$$

Conservation of Momentum:

$$\frac{\partial (\rho u_i)}{\partial t} + \frac{\partial (\rho u_i u_j + \delta_{ij} p)}{\partial x_j} - \frac{\partial \tau_{ij}}{\partial x_j} = \rho g_i\,, \tag{9.1.2}$$

Conservation of Energy:

$$\frac{\partial (\rho E)}{\partial t} + \frac{\partial [(\rho E + p) u_j]}{\partial x_j} - \frac{\partial (\tau_{ij} u_i)}{\partial x_j} + \frac{\partial q_j}{\partial x_j} = \rho g_i u_i\,, \tag{9.1.3}$$

Conservation of Species Fraction:

$$\frac{\partial (\rho Y_\alpha)}{\partial t} + \frac{\partial (\rho Y_\alpha u_j)}{\partial x_j} + \frac{\partial \mathcal{J}_{j,\alpha}}{\partial x_j} = \dot{\omega}_\alpha\,, \tag{9.1.4}$$

where

$$\tau_{ij} = \mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) - \frac{2}{3} \mu \delta_{ij} \frac{\partial u_l}{\partial x_l}\,, \tag{9.1.5}$$

$$q_j = -\lambda \frac{\partial T}{\partial x_j} - \rho \sum_{\alpha=1}^{N} h_\alpha \mathcal{D}_\alpha \frac{\partial Y_\alpha}{\partial x_j}\,, \tag{9.1.6}$$

$$\mathcal{J}_{j,\alpha} = -\rho \mathcal{D}_\alpha \frac{\partial Y_\alpha}{\partial x_j}\,, \tag{9.1.7}$$

9.2 Favre-Averaged Navier Stokes Equations

Favre-Filtering, essentially a density-weighted filtering procedure is defined as:

$$\tilde{\phi} = \frac{\overline{\rho \phi}}{\overline{\rho}}, \tag{9.2.1}$$

where $\tilde{\phi}$ represents the Favre-filtered variable.

Performing this on the governing equations, and assuming that the differentiation and filtering operations commute, we obtain the Favre-Filtered Equations as described by H. Perez [10] as follows:

Conservation of Mass:

$$\frac{\partial \bar{\rho}}{\partial t} + \frac{\partial (\bar{\rho} \tilde{u}_j)}{\partial x_j} = 0\,, \tag{9.2.2}$$

Conservation of Momentum:

$$\frac{\partial(\bar{\rho}\tilde{u}_i)}{\partial t} + \frac{\partial(\bar{\rho}\tilde{u}_i\tilde{u}_j + \delta_{ij}\bar{p})}{\partial x_j} - \frac{\partial\check{\tau}_{ij}}{\partial x_j} = \bar{\rho}g_i + \underbrace{\frac{\partial\sigma_{ij}}{\partial x_j}}_{\mathbf{I}} + \underbrace{\frac{\partial(\bar{\tau}_{ij} - \check{\tau}_{ij})}{\partial x_j}}_{\mathbf{II}}, \tag{9.2.3}$$

Conservation of Energy:

$$\begin{aligned}\frac{\partial(\bar{\rho}\tilde{E})}{\partial t} + \frac{\partial[(\bar{\rho}\tilde{E} + \bar{p})\tilde{u}_j]}{\partial x_j} - \frac{\partial(\check{\tau}_{ij}\tilde{u}_i)}{\partial x_j} + \frac{\partial\check{q}_j}{\partial x_j} \quad &= \quad \bar{\rho}\tilde{u}_i g_i - \underbrace{\frac{\partial[\bar{\rho}(\widetilde{h_{\mathrm{s}}u_j} - \check{h}_{\mathrm{s}}\tilde{u}_j)]}{\partial x_j}}_{\mathbf{III}} \\ &\quad + \underbrace{\frac{\partial(\overline{\tau_{ij}u_i} - \check{\tau}_{ij}\tilde{u}_i)}{\partial x_j}}_{\mathbf{IV}} - \underbrace{\frac{\partial(\bar{q}_j - \check{q}_j)}{\partial x_j}}_{\mathbf{V}} \\ &\quad - \underbrace{\frac{1}{2}\frac{\partial[\bar{\rho}(\widetilde{u_ju_iu_i} - \tilde{u}_j\widetilde{u_iu_i})]}{\partial x_j}}_{\mathbf{VI}} \\ &\quad - \underbrace{\frac{\partial[\sum_{\alpha=1}^{N}\Delta h^0_{\mathrm{f}_\alpha}\bar{\rho}(\widetilde{Y_\alpha u_j} - \tilde{Y}_\alpha\tilde{u}_j)]}{\partial x_j}}_{\mathbf{VII}},\end{aligned} \tag{9.2.4}$$

Conservation of Species Fraction:

$$\frac{\partial(\bar{\rho}\tilde{Y}_\alpha)}{\partial t} + \frac{\partial(\bar{\rho}\tilde{Y}_\alpha\tilde{u}_j)}{\partial x_j} + \frac{\partial\check{\mathcal{J}}_{j,\alpha}}{\partial x_j} = \underbrace{-\frac{\partial[\bar{\rho}(\widetilde{Y_\alpha u_j} - \tilde{Y}_\alpha\tilde{u}_j)]}{\partial x_j}}_{\mathbf{VIII}} - \underbrace{\frac{\partial(\bar{\mathcal{J}}_{j,\alpha} - \check{\mathcal{J}}_{j,\alpha})}{\partial x_j}}_{\mathbf{IX}} + \underbrace{\bar{\dot{\omega}}_\alpha}_{\mathbf{X}}, \tag{9.2.5}$$

Where we use the equation of state:

$$\bar{p} = \bar{\rho}\check{R}\tilde{T} + \underbrace{\sum_{\alpha=1}^{N} R_\alpha\bar{\rho}(\widetilde{Y_\alpha T} - \tilde{Y}_\alpha\tilde{T})}_{\mathbf{XI}}, \tag{9.2.6}$$

where

$$\sigma_{ij} = -\bar{\rho}\left(\widetilde{u_iu_j} - \tilde{u}_i\tilde{u}_j\right), \tag{9.2.7}$$

is the SFS stress tensor. The total energy is written as:

$$\tilde{E} = \check{h}_{\mathrm{s}} - \frac{\bar{p}}{\bar{\rho}} + \sum_{\alpha=1}^{N}\Delta h^0_{\mathrm{f}_\alpha}\tilde{Y}_\alpha + \frac{1}{2}\tilde{u}_i\tilde{u}_i + k_\Delta, \tag{9.2.8}$$

where

$$k_\Delta = \frac{1}{2}\left(\widetilde{u_iu_i} - \tilde{u}_i\tilde{u}_i\right), \tag{9.2.9}$$

is the Sub-Filter Scale (SFS) Turbulent Kinetic Energy. The effects of the subfilter scales appear in the

filtered total energy, $\tilde{E}$, the filtered equation of state and the right-hand-sides of the governing continuity, momentum, energy and species mass fraction equations (i.e., terms **I**,...,**XI**). The symbol ($\check{}$) is used to indicate the evaluation of expressions in terms of filtered variables, i.e., $\check{R}=R(\tilde{Y}_\alpha)$, $\check{h}_\mathrm{s}=h_\mathrm{s}(\tilde{Y}_\alpha,\tilde{T})$, and so on. The fluxes $\check{\tau}_{ij}$, $\check{q}_j$, and $\check{\mathcal{J}}_{j,\alpha}$ are expressed as

$$\check{\tau}_{ij} = 2\check{\mu}\left(\check{S}_{ij}-\frac{1}{3}\delta_{ij}\check{S}_{ll}\right), \tag{9.2.10}$$

$$\check{q}_j = -\check{\lambda}\frac{\partial\tilde{T}}{\partial x_j}-\bar{\rho}\sum_{\alpha=1}^{N}\check{h}_\alpha\check{D}_\alpha\frac{\partial\tilde{Y}_\alpha}{\partial x_j}, \tag{9.2.11}$$

$$\check{\mathcal{J}}_{j,\alpha} = -\bar{\rho}\check{D}_\alpha\frac{\partial\tilde{Y}_\alpha}{\partial x_j}, \tag{9.2.12}$$

where $\check{S}_{ij}=\frac{1}{2}\left(\partial\tilde{u}_i/\partial x_j+\partial\tilde{u}_j/\partial x_i\right)$, is the strain rate tensor calculated with the Favre-filtered velocity. The temperature used for the molecular transport coefficients $\check{\mu}$, $\check{\lambda}$, and $\check{D}_\alpha$ calculations is $\tilde{T}$.

Modelling of the SFS terms is required to close the above system of equations. Term **II** is neglected under the assumption that the filtered viscous stresses, $\bar{\tau}_{ij}$, can be approximated to the viscous stresses evaluated in terms of the Favre-filtered velocity, $\check{\tau}_{ij}$. Following similar assumptions for the total heat and species mass diffusion fluxes, terms **V** and **IX** may be neglected. Vreman *et al* [27] performed a priori LES of a mixing layer at different Mach numbers and concluded that neglecting the non-linearities of the diffusion terms in the momentum and energy equations is acceptable. Regarding term **IV** (SFS viscous diffusion), it is generally much smaller than the other terms that require a model [28], and so is neglected. As for term **XI** (SFS temperature-species correlation), it is assumed to be small and generally neglected. Following the work of Knight *et al.* [29], term **VI** (the SFS turbulent diffusion) can be modelled in terms of the SFS stresses and the resolved velocity as

$$-\frac{\bar{\rho}\left(\widetilde{u_j u_i u_i}-\tilde{u}_j\widetilde{u_i u_i}\right)}{2}=\sigma_{ij}\tilde{u}_i\,. \tag{9.2.13}$$

Term **VII** involves the SFS species fluxes (term **VIII**) and is closed with the modelled SFS species fluxes.

References

[1] Excerpts from a conversation with gordon moore: Moore's law. Video Transcript 0305/TSM/LAI/XX/PDF 306959-001US, Intel Corporation, September 2005.

[2] Y. Kaneda and T. Ishihara. High-resolution direct numerical simulation of turbulence. *Journal of Turbulence*, 7(20), 2006.

[3] S. A. Northrup. *A Parallel Implicit Adaptive Mesh Refinement Algorithm for Predicting Unsteady Fully-Compressible Reactive Flows*. PhD thesis, University of Toronto, 2013.

[4] W. Deconinck. Design and application of discrete explicit filters for large eddy simulation of compressible turbulent flows. Master's thesis, University of Toronto, 2008.

[5] L. Ivan and C. P. T. Groth. High-order central ENO finite-volume scheme with adaptive mesh refinement. Paper 2007-4324, AIAA, June 2007.

[6] L. Ivan and C. P. T. Groth. High-order solution-adaptive central essentially non-oscillatory (CENO) method for viscous flows. *Journal of Computational Physics*, 257:830–862, 2013.

[7] Z. J. Zhang and C. P. T. Groth. Parallel high-order anisotropic block-based adaptive mesh refinement finite-volume scheme. Paper 2011-3695, AIAA, June 2011.

[8] M. J. Williamschen and C. P. T. Groth. Parallel anisotropic block-based adaptive mesh refinement algorithm for three-dimensional flows. Technical report, American Institute of Aeronautics and Astronautics, June 2013. 21st AIAA Computational Fluid Dynamics Conference.

[9] L. Freret and C. P. T. Groth. Anisotropic non-uniform block-based adaptive mesh refinement for three-dimensional inviscid and viscous flows. 22nd aiaa computational fluid dynamics conference, American Institute of Aeronautics and Astronautics, June 2015. Accepted.

[10] F. E. Hernández-Pérez. *Subfilter Scale Modelling for Large Eddy Simulation of Lean Hydrogen-Enriched Turbulent Premixed Combustion*. PhD thesis, University of Toronto, 2011.

[11] T Lund. The use of explicit filters in large eddy simulation. *Computers and Mathematics with Applications*, Jan 2003.

[12] Stephen B. Pope. *Turbulent Flows*, chapter 6, page 183. Cambridge University Press, 4th edition, 2005.

[13] J Smagorinsky. General circulation experiments with the primitive equations. *Monthly Weather Review*, 91(3):99–164, 1963.

[14] E. F. Toro. *Riemann solvers and numerical methods for fluid dynamics : a practical introduction*. Springer, Berlin, New York, 1997.

[15] L. Tobaldlini Neto and C. P. T. Groth. A high-order finite-volume scheme for large-eddy simulation of turbulent premixed flames. Technical report, American Institute of Aeronautics and Astronautics, January 2014. 52nd Aerospace Sciences Meeting.

[16] C. P. T. Groth. Control and reduction of numerical errors in reactive flow les. Technical report, UTRC Workshop, September 2013.

[17] M. R. J. Charest and C. P. T. Groth. A high-order central eno finite-volume scheme for three-dimensional turbulent flows on unstructured mesh. Technical report, American Institute of Aeronautics and Astronautics, June 2013. 21st AIAA Computational Fluid Dynamics Conference.

[18] C. P. T. Groth, D. L. De Zeeuw, K. G. Powell, T. I. Gombosi, and Q. F. Stout. A parallel solution-adaptive scheme for ideal magnetohydrodynamics. Technical report, American Institute of Aeronautics and Astronautics, June 1999. 14th Computational Fluid Dynamics Conference.

[19] C. P. T. Groth and S. A. Northrup. Parallel implicit adaptive mesh refinement scheme for body-fitted multi-block mesh. Technical report, American Institute of Aeronautics and Astronautics, June 2005. 17th AIAA Computational Fluid Dynamics Conference.

[20] Z. J. Zhang. Parallel anisotropic block-based adaptive mesh refinement finite-volume scheme. Master's thesis, University of Toronto, 2011.

[21] N. Shahbazian, C. P. T. Groth, and Ö. L. Gülder. Assessment of presumed pdf models for large eddy simulation of turbulent premixed flames. Paper 2011-0781, AIAA, January 2011.

[22] M. B. Giles and N. A. Pierce. An introduction to the adjoint approach to design. *Flow, Turbulence and Combustion*, 65:393–415, 2000.

[23] R. Becker and R. Rannacher. An optimal control approach to a posteriori error estimation in finite element methods. *Acta Numerica*, 10:1–102, 2001.

[24] D. A. Venditti and D. L. Darmofal. Anisotropic grid adaptation for functional outputs: application to two-dimensional flows. *Journal of Computational Physics*, 187:22–46, 2003.

[25] K. J. Fidkowski and D. L. Darmofal. Review of output-based error estimation and mesh adaptation in computational fluid dynamics. *AIAA Journal*, 49(4), 2011.

[26] K. J. Fidkowski. High-order output-based adaptive methods for steady and unsteady aerodynamics. Invited lecture, Von Karman Institute, December 2013.

[27] B. Vreman, B. Geurts, and H. Kuerten. Subgrid-modeling in LES of compressible flow. 54:191–203, 1995.

[28] M. P. Martín, U. Piomelli, and G. V. Candler. Subgrid-scale models for compressible large-eddy simulations. *Theoretical and Computational Fluid Dynamics*, 13:361–376, 2000.

[29] D. Knight, G. Zhou, N. Okong'o, and V. Shukla. Compressible large eddy simulation using unstructured grids. Paper 98-0535, AIAA, January 1998.