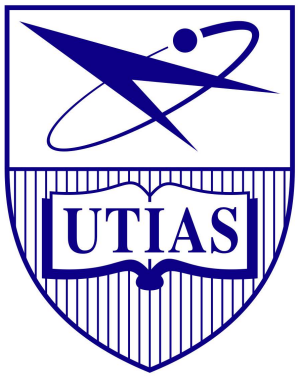


Development of $h-p$ adjoint-based error estimation for LES of reactive flows

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 help reduce time and cost of prototype design for engineering systems. Practical reactive flows almost always involve turbulence, and CFD methods have been developed to help capture such complex phenomena to varying extents of accuracy. Corresponding physical experiments can take a long time and can be very expensive to conduct. Three main approaches exist. Reynolds averaged Navier Stokes (RANS) uses full modeling of turbulent scales. Large eddy simulations (LES) model sub-filter scales (SFS) while resolving larger scales. Direct numerical simulations (DNS) resolve all the turbulent scales. 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 [1], but such calculations are very expensive and limited to simple cases with low Reynolds numbers. More complicated analyses such as those involving combustion simulations use LES to achieve an overall good prediction of results at an accuracy greater than RANS, but at a cost lower than DNS.

To improve accuracy within the CFD models, we need to reduce numerical error within LES simulations. Key sources of error in CFD include numerical and modeling errors. Examples of numerical errors are solution errors, truncation errors and convergence errors. A common alternative for reducing error is the gradient (physics)-based approach, but this technique is actually less efficient, in that it error reduction is not guaranteed for continual levels of mesh (h) refinement. More on gradient-based methods will be discussed in section 4. This work proposes an implementation of an adjoint-based technique for error estimation. The adjoint-based method is a more effective approach to reduce numerical errors at a potentially lower cost would be the use of adjoint-based error estimates for localized error reduction. The adjoint-based technique is frequently used for localized mesh adaptation (h) refinement. However, this work proposes the combination of (h) refinement with (p) refinement. That is, the additional use of high-order discretization schemes (of accuracy $p > 2$) to further improve accuracy of the CFD results, while lowering the cost to attain them. This combination can be expressed as an adjoint-based $\mathcal{O}(h^p)$ error estimation technique and is proposed for error reduction (while reducing computational costs) in LES of reactive flow simulations.

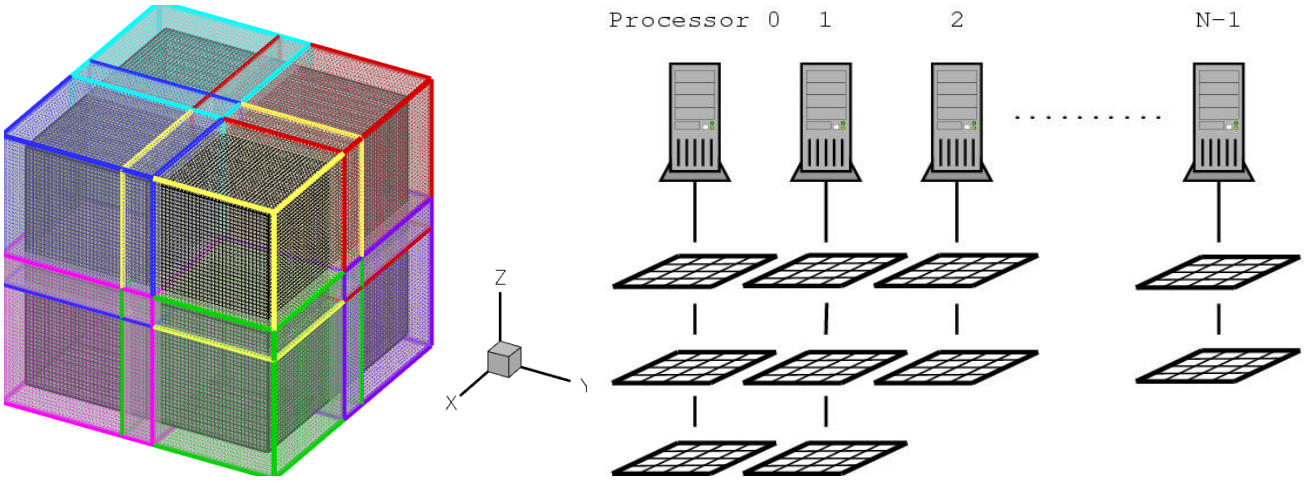
2 Proposed error estimation strategy

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

2.1 Block-based adaptive mesh refinement (AMR)

The model geometry needs to be discretized into unique cells, in a process called meshing. For increase in accuracy, a finer mesh resolution (smaller grid size) is usually be required.

Adaptive Mesh Refinement (AMR) takes its name from the increase in localized grid resolution to allow



(a) 3D Block-Ghost configuration [9]

(b) Parallel domain decomposition [9]

Figure 1: The block-cell structure renders block based AMR easily parallelizable [10]

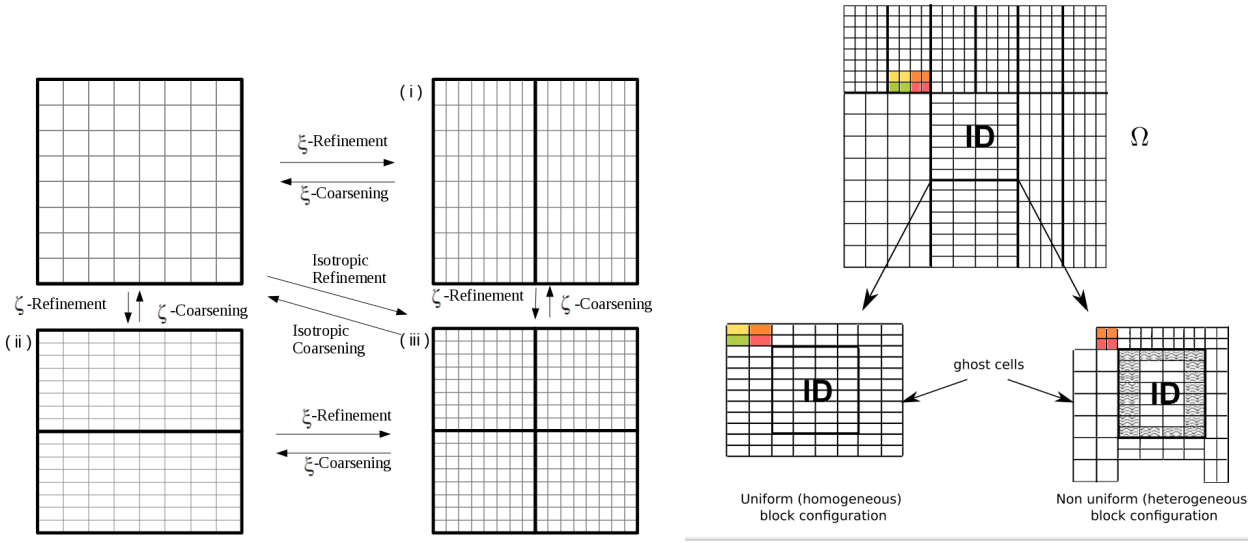
for improved accuracy, and this commonly results in a reduction of cost. Performing a uniform refinement over all the cells would be easier to perform, but often results in a high performance and storage cost to achieve the same effectiveness as s localized approach.

Groth *et al* [2, 3, 4, 5, 6, 7] developed some of the initial work with Block-Based AMR extended it to enable parallel implementation for a variety of complex flows.

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. Message Passing Interface (MPI) could be used for processor inter-communication, whereby information exchange between the different blocks is facilitated by ghost cells that lie in the periphery of the block. Originally, the ghost cells were of the same mesh refinement level as the interior block.

However, Freret and Groth [8] 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. This eliminates the interpolation error in multiple restriction and prolongation cycles within a uniform block-based scheme, while saving the computational cost of otherwise-necessary message passing to ensure flux conservation between the blocks. This message passing was described by Zhang [7] as a temporary remedy for inter-block flux evaluation applied to the ghost-cells.

Northrup and Groth [4] implemented a fully-implicit, parallel Newton-Krylov method, and applies Schwarz preconditioning to take advantage of the Block structure of the computational grid. Within 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:



(a) Isotropic and anisotropic AMR. Zhang, [6]

(b) Non-uniform block structure, Freret and Groth [8]

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

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.

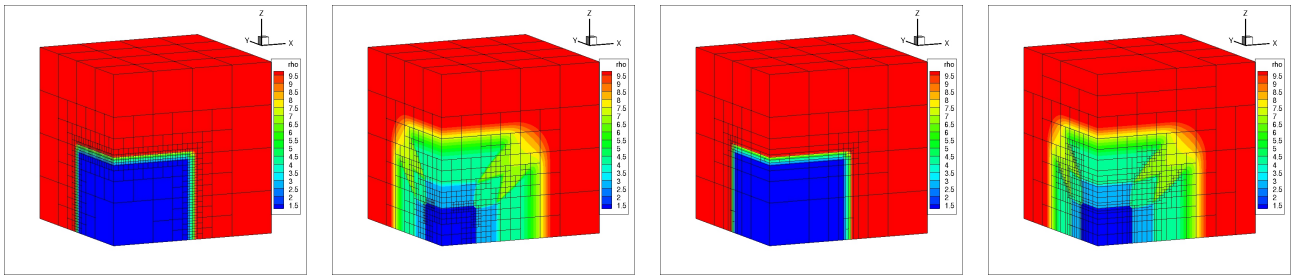
2. **Anisotropic AMR**:

   For flows with directional bulk bias it can be predicted that mesh cells would ideally be stretched along this axis. Anisotropy refers to a preference in mesh refinement axes. Studies by Zhang [6], Zhang and Groth [7], Freret and Groth [8] indicate that anisotropic AMR for given simulations result in lower cell counts (see figure 3), although the refinement procedure is more difficult than that of the Isotropic AMR.

2.2 The finite volume method

Godunov-type Finite Volume Methods (FVM) will typically apply a cell-centered spatial discretization. This results in the splitting of the solution domain into control volumes [11]. 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



(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 3: 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. For this case, a savings of $\approx 28\%$ in cell count is achieved by the anisotropic method. [Freret and Groth: 2015] [8]

can be re-expressed in the following general weak conservation form using matrix-vector notation [12]

$$\frac{\partial \overline{\mathbf{U}}}{\partial t} + \vec{\nabla} \cdot \vec{\mathcal{F}} = \frac{\partial \overline{\mathbf{U}}}{\partial t} + \vec{\nabla} \cdot \vec{\mathcal{F}}^{\mathrm{I}} \left(\overline{\mathbf{U}} \right) - \vec{\nabla} \cdot \vec{\mathcal{F}}^{\mathrm{V}} \left(\overline{\mathbf{U}}, \vec{\nabla} \overline{\mathbf{U}} \right) = \overline{\mathbf{S}} \tag{2.2.1}$$

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

Inviscid fluxes (the hyperbilic part) within each cell (control volume) are resolved via a solution to a Riemann Problem, given the different left and right states for each cell. The goal is to obtain the final cell-centered, cell-average value. Typically, approximate Riemann solvers are used to this end.

For the purpose of CFD, these governing equations in PDE form need to be discretized. The integral form of Eq. (2.2.1) is applied to a hexahedral reference cell (i, j, k) and an N_G-point Gaussian quadrature numerical integration procedure is used to evaluate the solution flux along each of the N_f faces of the cell, to obtain:

$$\frac{\mathrm{d}\overline{\mathbf{U}}_{ijk}}{\mathrm{d}t} = -\frac{1}{V_{ijk}} \sum_{l=1}^{N_f} \sum_{m=1}^{N_{GF}} \left(\omega_m \left(\vec{\mathcal{F}}^{\mathrm{I}} - \vec{\mathcal{F}}^{\mathrm{V}} \right) \cdot \hat{n} A \right)_{ijk,l,m} + \sum_{n=1}^{N_{GV}} \left(\omega_n \mathbf{S} \right)_{i,j,k,n} = \overline{\mathbf{R}}_{ijk} \left(\overline{\mathbf{U}} \right), \tag{2.2.2}$$

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 2.2.1);
- 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.

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)

2.2.1 The centrally essentially non-oscillatory (CENO) scheme

In solution reconstruction, given 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 [[13], [14]] 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 technique 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 [5]:

- It provides accuracy of other Essentially Non-Oscillatory (ENO) schemes and maintains monotonicity near discontinuities;
- Identifies regions where under-resolution and non-smooth data occurs, and this could prove useful for mesh adaptation techniques.

2.3 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 applies a filter size ($\bar{\Delta}$) to the turbulence scales, resolving the larger ones, and modelling 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. Two main techniques for Filtering are used: implicit and explicit filtering.

In implicit filtering, the filter width is not explicitly defined and is inherently related to grid resolution. The main disadvantage is that it is difficult to compare results of adapted/refined meshes. Hence there is need for explicit filtering, whereby Define $\bar{\Delta}$ is fixed for the entire mesh, and convergence tests are more consistent. [15, 16]

2.3.1 Aliasing errors

As a result of Favre filtering on the governing equations, (see section 9.2) a non-linear correlation will result:

$$\frac{\partial}{\partial x_{\mathrm{j}}}(\overline{\rho u_{\mathrm{i}} u_{\mathrm{j}}}) \,, \tag{2.3.1}$$

This creates a closure problem. To solve this, the product of the closed filtered velocities is evaluated and the remaining term is modeled, essentially transfering the closure problem to the right-hand side of the filtered Navier-Stokes equation:

$$\overline{\rho u_{\mathrm{i}} u_{\mathrm{j}}} = \bar{\rho}\tilde{u}_{\mathrm{i}}\tilde{u}_{\mathrm{j}} + \underbrace{(\overline{\rho u_{\mathrm{i}} u_{\mathrm{j}}} - \bar{\rho}\tilde{u}_{\mathrm{i}}\tilde{u}_{\mathrm{j}})}_{\sigma_{\mathrm{ij}}} \,, \tag{2.3.2}$$

where σ_{ij} is the SFS stress tensor. This decomposition leads to the non-linear product $\tilde{u}_{\mathrm{i}}\tilde{u}_{\mathrm{j}}$ generating frequencies that are beyond the characteristic frequency that defines $\tilde{u}_{\mathrm{i}}$ and act as fictitious stresses [17]. The resulting errors are difficult to eliminate and control if implicit filtering is used. Explicit filtering allows control of these. Please refer to the work of H. Perez [15] and Deconinck [16] for additional details.

2.3.2 Commutation errors

For most inhomogeneous turbulent flows, the smallest eddy sizes that can be resolved differ in various regions of the flow. Usually, flow close to solid walls produces smaller eddies due to 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 [15] as follows:

$$\overline{\frac{\mathrm{d}\phi}{\mathrm{d}x}} = \frac{\mathrm{d}\bar{\phi}}{dx} \,. \tag{2.3.3}$$

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.,

$$\left[\frac{\mathrm{d}\phi}{\mathrm{d}x}\right] \equiv \left|\overline{\frac{\mathrm{d}\phi}{\mathrm{d}x}} - \frac{\mathrm{d}\bar{\phi}}{\mathrm{d}x}\right| = \mathcal{O}(\Delta^n) \,, \tag{2.3.4}$$

where n is the order of the filter, which should be at least equal to the order of accuracy of the spatial discretization scheme. Thus for a p^{th} order numerical scheme, $n = p$.

2.3.3 Sub-filter scale (SFS) modelling

The SFS terms (see section 9.2) need to be modeled in LES, 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.[18].

A number of approaches have been used to model the SFS terms. One of the first was by Smagorinsky [19] which models SFS stress tensors and the Reynolds' stress tensor. This approach also defines a SFS 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. In the case where the Smagorinsky coefficient is valid only for an optimized flow regime, we obtain the dynamic Smagorinsky model. This will likely be the model of choice for this research.

2.4 Combustion modeling 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 [15] and Shahbazian *et al* [20].

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. [15]

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.

3 Error

3.1 Overview on error

We can broadly classify the two key sources of error in CFD as follows:

- Numerical error:
  - Solution error: This is evaluated at the when comparing the CFD simulation prediction value to those of the exact known solution value

  - Truncation error: Formed when the actual governing equations (PDEs) are discretized into an approximate form for the numerical scheme
  - Convergence error: They arise due to nature of the iterative technique used. Whether the residuals actually asymptote to machine-zero levels.

- Modeling error:
  - Sub-filter scale (SFS) turbulence model: Since LES relies on modelling of the scales smaller than the filter size, then any inappropriate model selected would lead to over-/underprediction of the solution values
  - Combustion and chemistry model: In LES we use the PCM-FPI to model to predict chemical reaction rates. If the model has an error, then this will reflect on the chemical kinetics.
  - Aliasing errors occur when decomposed nonlinear terms in the FANS generate feedback of frequencies which are beyond the filter bandwidth, and eventually resolve as fictitious stresses
  - Commutation errors exist between filtering and differential operations.

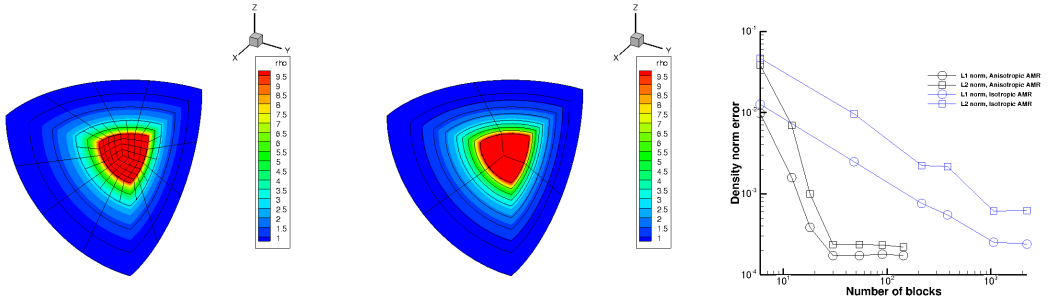
3.2 Error estimation

The basis for selective mesh refinement is the need to refine the mesh where the cells have a very critical effect on the solution, while coarsening the less critical areas to save on computational costs. There are two types of error estimation procedures available:

- a priori error estimators: these predict the long-term behavior of the errors in the discretization. They are not actually designed to approximate the error estimate for a given mesh.
- a posteriori error estimators: these use the simulation results to derive estimates of solution errors. Furthermore, these results are used to guide adaptive schemes:
  - The mesh is locally refined (h-refinement)
  - The degree of polynomial solution representation is raised (p-increment)

Two main a posteriori approaches are the:

- gradient-based: this is the more commonplace, and some useful discussions have been done by Giles and Pierce [21]
- adjoint-based: this will be described in more detail in section 4. Extensive work has been done by Giles and Pierce, [21], Venditti and Darmofal [22, 23, 24] Fidkowski and Darmofal: [25]



(a) With isotropic AMR (b) With anisotropic AMR (c) Error in the density norm

Figure 4: Comparison of anisotropic and isotropic AMR and output error norms

3.3 Gradient-based refinement

In these simulations, the mesh or discretization order is changed based on the rates of change of (physical) solution variables. The locations having the sharpest changes in solution quantities over a few mesh cells is flagged, and here the mesh resolution can be increased (higher mesh refinement), or the scheme order can be increased, effectively using a higher order discretization over these cells.

- The reasoning behind this is to have enough cells to capture the solution state changes so to represent them as *smoothly* as possible.
- Once refinement is completed, the solution is re-run and the gradients re-evaluated. Changes made as necessary. Error can be compared to a higher discretization (h or p) solution.
- This is the present utility in the anisotropic and isotropic AMR functionality of the CFFC code used by the CFD and Propulsion group.
- As has been indicated in figure 4(c) the gradient-based approach does not guarantee error reduction for higher levels of h refinement.

The number of cells to reach a convergence criteria is reduced in the case of figure 4(c) by a factor of approximately 1.5 Image sources: Freret and Groth, [8] and Williamschen and Groth [3]
A clear disadvantage is that for the gradient based approach, in spite of more mesh refinement, the error norm does not reduce. The graph reveals this asymptotic behavior of the convergence, yet for increased number of cells, there should be continual reduction in the density error norm.

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.

To make error estimation more relevant to engineering applications it is useful to assess the error made in predicting an integral quantity which represents an engineering output. This output is called the functional. For example,the output can be the average pressure on a wall. The adjoint technique is a sensitivity analysis, that measures the rates of change of a design functional to a given change in the input.

Extensive research has been performed by Giles and Pierce [21], Becker and Rannacher [26], Venditti and Darmofal [22, 23] and a review carried out by Fidkowski and Darmofal [25].

The adjoint has two main formulations: the continuous and the discrete.

In the *continuous* approach, an objective function is formed to enforce the flow conditions (i.e. primal nonlinear PDEs). Next, linear perturbations to the primal flow variables are considered, while enforcing the objective function to remain constant with respect to the perturbations. As a result, analytical adjoint equations are obtained, and relevant boundary conditions applied. The formulation is then discretized directly.

For the *discrete* formulation, the nonlinear discrete residual equations from the primal problem are the starting point, and, similar to the continual approach, linear perturbations are applied. If the system is adjoint consistent, i.e. discrete adjoint = continuous adjoint, then there is no need for boundary condition specification. These get automatically incorporated via the primal residual. Finally, a linear system of equations is established with the only necessary evaluation being the linear sensitivities of the functional and the Jacobian matrix associated with the primal residual.

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.

The adjoint-based method applies 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, error estimates can be evaluated from the adjoint, and this could be used as a mesh refinement criteria.

Other key research on adjoints applied to aerodynamic flows was performed by Giles and Pierce [21], Becker and Rannacher [26] 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 [27], 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 **U** such that **R(U,**μ = 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 **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:

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

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:

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

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. Ways to calculate $\frac{\partial \mathbf{J}}{\partial \mathbf{U}}$:

- Complexifying variables by calculating derivatives using complex numbers. Martins et al, [28]
- Finite Differences where the state $\mathbf{U}$ is perturbed to get updated values of $\mathbf{R}(\mathbf{U})$
- Automatic Differentiation Techniques. The ADIFOR tool written by Bischof, [29].
- 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 [10] 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, $\mathbf{Ax} = \mathbf{b}$, 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 time-step within an unsteady simulation (e.g. for the goal of the project, which is to simulate Turbulent Premixed Flows). This occurs via running the simulation forward in time while evaluating all the *Primal* solution values at the different time levels until the final time-step, and then marching backwards in time, and solving the Adjoint at each of those time-steps, 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 Scope of research

- The goal is Reducing numerical error using high order CENO and adjoint based error estimation, $\mathcal{O}(h^p) \rightarrow h$ and p adaptation
- This work proposes a two-pronged technique to reduce the numerical error within the discretized governing equations, as well as in the computational domain (mesh). Implementation of high order finite volume method (FVM) as well as adjoint-based error estimation is believed to be a significant step to achieve this. The adjoint technique can be used for $\mathcal{O}(h^p)$ refinement: evaluating error estimates and then locally switching between h and p refinement, based on the computational cost savings.
- The CFFC code already has most of the needed formulations required for this work.
  - Combustion modeling:

    - PCM-FPI: This model allows detailed chemical kinetics via tabulation of precomputed laminar premixed flames, and has already been implemented within the CFFC code by H. Perez (2011) [30]
  - Favre-filtered Navier Stokes governing equations: The FANS formulation has already been implemented within the group CFFC code.
  - Large Eddy Simulation:
    - Explicit filtering work done by Deconinck, [16]
    - Sub-filter scale (SFS) modeling work done by H. Perez [30]
  - High-order finite volume methods: CENO technique - benefits of higher accuracy on a coarse mesh was implemented by Ivan and Groth [13, 14]
  - AMR
    - Block-based AMR: speed and parallelization [Groth et al 1999]
    - Anisotropic AMR reduces cell count (computational cost): 2-D initial work by Zhang and Groth [7], and extended to 3-D by Williamschen and Groth [3]
    - Freret and Groth, [8] extended the work of Williamschen to include a non-uniform block formulation. This increases code accuracy while saving computational resources.
  - Solution method: Northrup and Groth [10, 31] implemented an implicit GMRES solver that improves the robustness and capability of the CFFC code by relieving it from the Courant-Friedric-Levy (CFL) condition constraints.

- Workflow:
  - Creating a framework for the adjoint based error estimation method. The approach to be used will be to first formulate the adjoint solver within the group CFFC Code.
  - Once the adjoint is formulated, it will be validated on already known inviscid (Euler) laminar steady solutions, and then implemented for unsteady cases. Complexity will be introduced by adding viscosity and then a temporal variation.
  - The next step will be to advance the work done by Deconinck on explicit filters and implement them [16] to the high-order FVM scheme.

- The implementation of the adjoint based error estimation would allow for significant reductions in numerical error, and these will be accounted for in the duration of this research.

6 Error estimation indicators

This is based on the work of Venditti and Darmofal [22] and Fidkowski and Darmofal [25]. The goal is to reduce discretization errors based on the mesh resolution.

- Consider 2 levels of mesh resolution: coarse (H) and fine (h). We calculate the state ($\mathbf{U}_H$) and functional ($\mathbf{J}_H(\mathbf{U}_H)$) on the coarse space. Residual, $\mathbf{R}_H(\mathbf{U}_H) = 0$
- We would like to evaluate the functional on the fine space, $\mathbf{J}_h(\mathbf{U}_h)$ (expensive). Thus we use a prolongation operator ($\mathbf{U}_h^H = I_h^H \mathbf{U}_H$) to inject the fine space state onto the coarse space state.
- The output error, $\delta\mathbf{J} \equiv \mathbf{J}_H(\mathbf{U}_H) - \mathbf{J}_h(\mathbf{U}_h) \neq 0$
- Expect the new residual, $\mathbf{R}_h(\mathbf{U}_h^H) \neq 0$
- The injected coarse state solves: $\mathbf{R}_h(\mathbf{U}'_h) - \mathbf{R}_h(\mathbf{U}_h^H) = 0 \rightarrow (\mathbf{U}'_h) = (\mathbf{U}_h^H)$
- $\delta\mathbf{J} \approx \mathbf{J}_h(\mathbf{U}_h^H) - \mathbf{J}_h(U_h) = \mathbf{\Psi}_h^T \delta\mathbf{R}_h = -\mathbf{\Psi}_h^T \mathbf{R}_h(\mathbf{U}_h^H)$, using the definition of the fine space adjoint, $\mathbf{\Psi}_h$
- Error estimate is the value of $\delta\mathbf{J}$, and does not need evaluation of U_h, primal solution on the fine space. We can use this error estimate as a flag for refinement, given some threshold value

For unsteady adjoints, the adaptation is achieved by marching the solution forward in time while evaluating ψ, and then marching backwards in time. Once all the sensitivities are obtained, they can be used to refine the mesh at those time levels. The tolerance is compared to the set threshold value and repeat until convergence.

One of the main benefits of adjoint vs gradient based methods are that the adjoint technique is a one time calculation for the sensitivity of a single output to several inputs, whereas a gradient based approach requires a separate evaluation for each new sensitivity. [21]

One other use for the evaluation of the error estimate is that it can be used to flag cells for adaptation. This can be done by either refining the mesh (h adaptation), or for this block of cells, a higher order scheme can be used to achieve the higher accuracy. This would essentially materialize the ability for $\mathcal{O}(h^p)$ refinement.

7 Simulations carried out to date

7.1 Poisson model problem

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

We seek to solve the Poisson equation

$$-\Delta u = f \text{ in } D, \tag{7.1.1}$$

$$u = 0 \text{ on } \partial D, \tag{7.1.2}$$

- 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 (7.1.1). 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 (7.1.2), write down a linear system

$$\mathbf{A}\mathbf{u} = \mathbf{b} \tag{7.1.3}$$

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 (7.1.1). 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 (7.1.2), write down a linear system

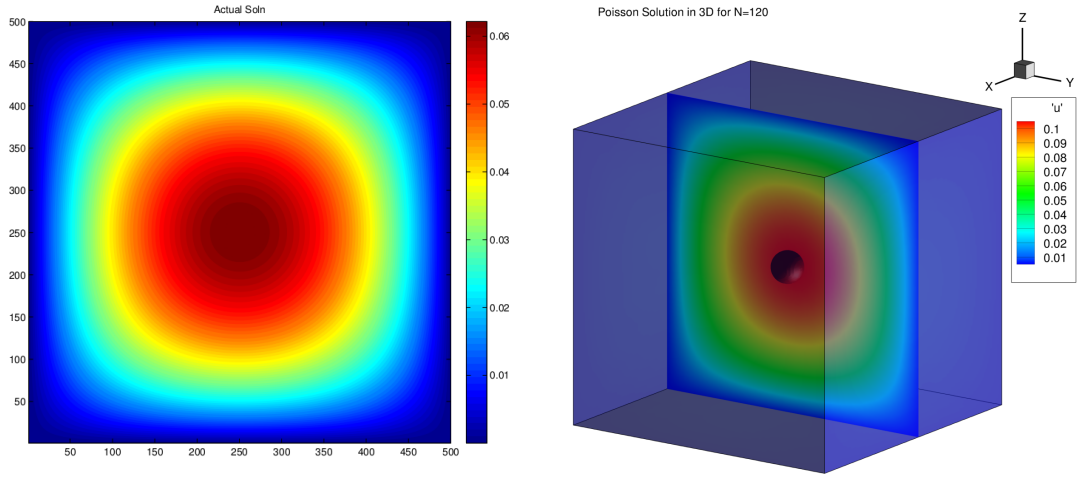
$$\mathbf{A}\mathbf{u} = \mathbf{b} \tag{7.1.4}$$

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.

7.2 LES of a turbulent premixed methane flame

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 **??**, the contours representing the species concentration of Oxygen.

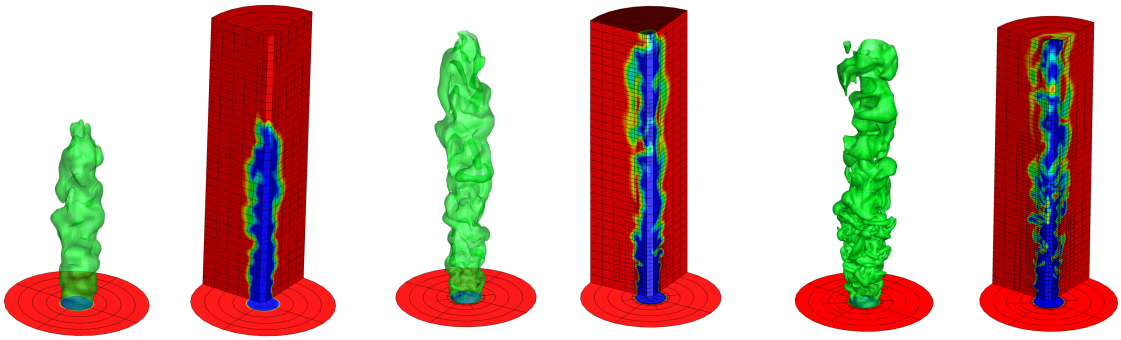
Included here are some pictures from previous CFFC simulations that show the varying mesh cell sizes among the different blocks.



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

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

Figure 5: Solution Contours for solution of the Poisson Problem



(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 6: 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.

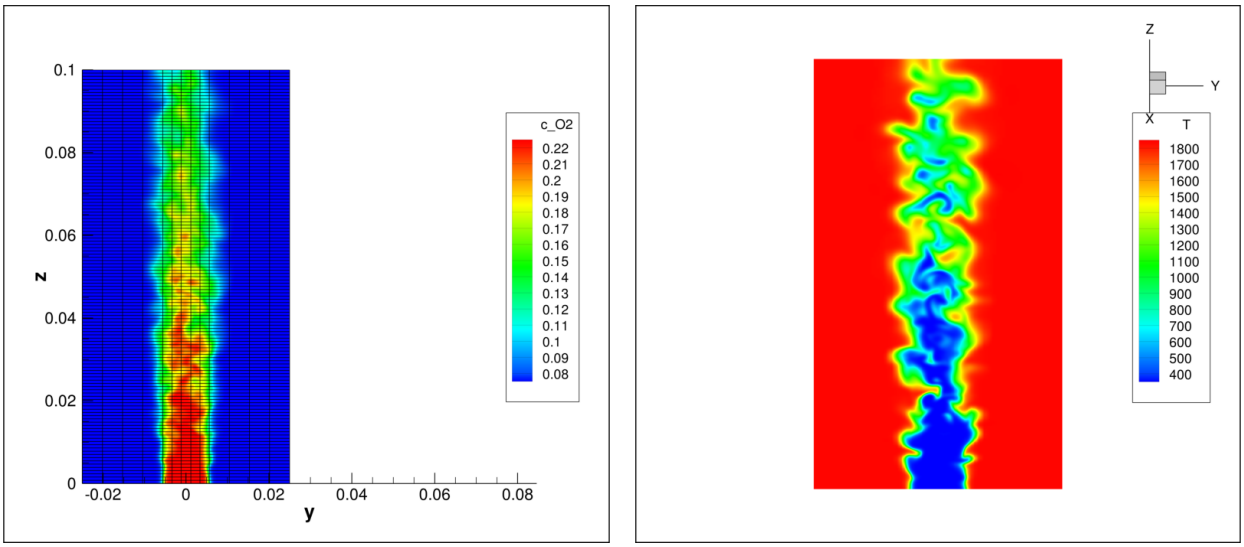
7.3 Discrete adjoint solutions on a shock cube problem

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:

$$\mathbf{A}^T\mathbf{x} = -\mathbf{b}^T \tag{7.3.1}$$

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 $\mathbf{b}$ as the derivative of the functional with respect to the state, $\frac{\partial J}{\partial U_i}$

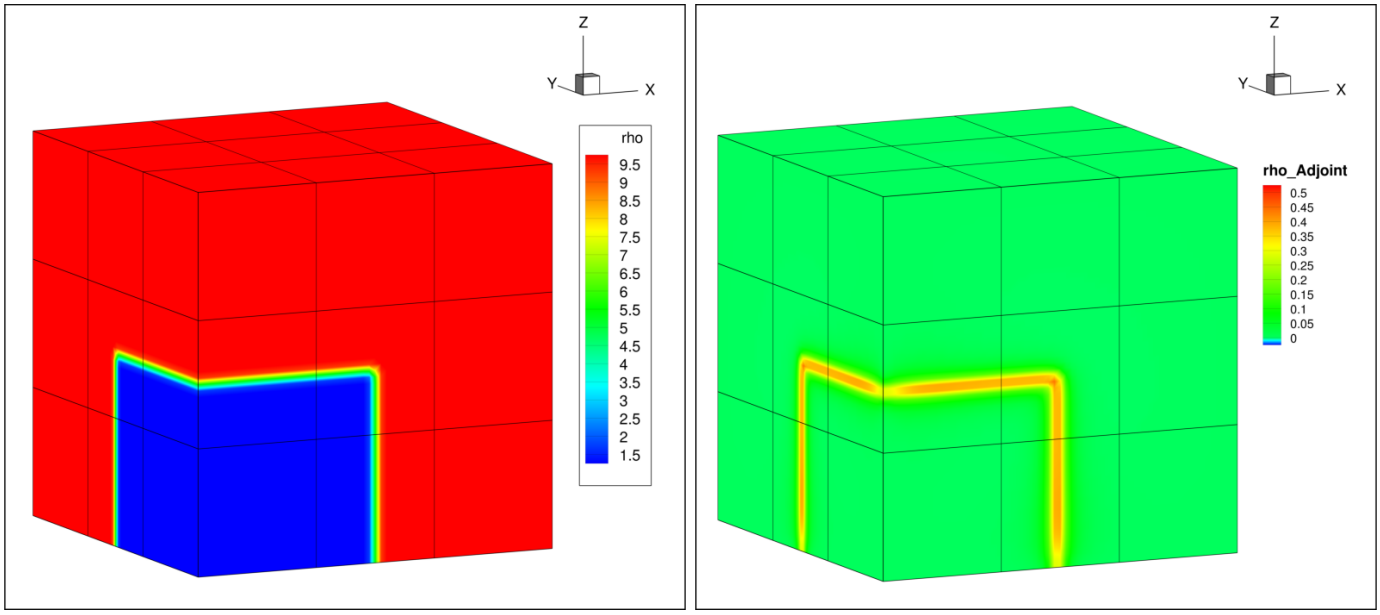
Adjoint sensitivities were evaluated for a shock cube problem governed by the Euler equations (see figure 8). The initial conditions were $\frac{\rho_L}{\rho_R} = 8$, $\frac{P_R}{P_L} = 10$. Average pressure in the entire domain was selected as the functional, J.



(a) Time averaged O_2 species concentration (t = 6-11 ms)

(b) Flame temperature at t= 9.0 ms

Figure 7: Turbulent premixed methane flame having equivalence ratio $\Phi = 0.7$



(a) Initial conditions for density showing large jump between left and right states

(b) sensitivity to Density

Figure 8: Contours of ρ and adjoint ψ_ρ at t = 0 sec. 27 (20x20x20) blocks

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	March 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 a full thermo-coupling problem	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 [15]

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 [15] 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* [32] 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 [33], 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.* [34], 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] Y. Kaneda and T. Ishihara. High-resolution direct numerical simulation of turbulence. *Journal of Turbulence*, 7(20), 2006.

[2] 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.

[3] 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.

[4] 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.

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

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

[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] 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.

[9] R. Rashad. Development of a high-order finite-volume method for the navier-stokes equations in three dimensions. Master's thesis, University of Toronto, 2009.

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

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

[12] 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.

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

[14] 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.

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

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

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

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

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

[20] 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.

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

[22] D. A. Venditti and D. L. Darmofal. Adjoint error estimation and grid adaptation for functional outputs: Application to quasi-one-dimensional flow. *Journal of Computational Physics*, 164:204– 227, 2000.

[23] D. A. Venditti and D. L. Darmofal. Grid adaptation for functional outputs: Application to two-dimensional inviscid flows. *Journal of Computational Physics*, 176:40– 69, 2002.

[24] D. A. Venditti and D. L. Darmofal. Anisotropic grid adaptation for functional outputs: application to two-dimensional viscous 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] R. Becker and R. Rannacher. An optimal control approach to a posteriori error estimation in finite element methods. *Acta Numerica*, 10:1–102, 2001.

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

[28] J. R. R. A. Martins, P. Sturdza, and J. J. Alonso. The complex-step derivative approximation. *ACM Transactions on Mathematical Software*, 29(3):245–262, 2003.

[29] Christian Bischof, Alan Carle, Peyvand Khademi, and Andrew Mauer. The adifor 2.0 system for the automatic differentiation of fortran 77 programs. In *RICE UNIVERSITY*, pages 18–32, 1994.

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

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

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

[33] 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.

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