A component-based toolkit for simulating reacting flows with high order spatial discretisations on structured adaptively refined meshes

Sophia Lefantzi, Jaideep Ray*, Christopher A. Kennedy and Habib N. Najm
Sandia National Laboratories, PO Box 969,
MS 9056, Livermore CA 94551-0969, USA
Fax: 925 294 3638/2595 E-mail: jairay@ca.sandia.gov
E-mail: slefant@ca.sandia.gov E-mail: cakenne@speakeasy.net
E-mail: hnnjm@ca.sandia.gov
*Corresponding author

Abstract: We present an innovative methodology for developing scientific and mathematical codes for computational studies of reacting flow. High-order (>2) spatial discretisations are combined, for the first time, with multi-level block structured adaptively refined meshes (SAMR) to resolve regions of high gradients efficiently. Within the SAMR context, we use 4th order spatial discretisations to achieve the desired numerical accuracy while maintaining a shallow grid hierarchy. We investigate in detail the pairing between the order of the spatial discretisation and the order of the interpolant, and their effect on the overall order of accuracy. These new approaches are implemented in a high performance, component-based architecture (Common Component Architecture) and achieve software re-usability, flexibility and modularity. The high-order approach and the software design are demonstrated and validated on three test cases modelled as reaction-diffusion systems of increasing complexity. We also demonstrate that the 4th order SAMR approach can be computationally more economical compared to second-order approaches.

Keywords: structured adaptive mesh refinement (SAMR); reacting flow; high performance computing; high order spatial interpolations; Common Component Architecture (CCA).

Reference to this paper should be made as follows: Lefantzi, S., Ray, J., Kennedy, C.A. and Najm, H.N. (xxxx) 'A component-based toolkit for simulating reacting flows with high order spatial discretisations on structured adaptively refined meshes', *Progress in Computational Fluid Dynamics*, Vol. x, No. x, pp.xxx–xxx.

Biographical notes: Sophia Lefantzi is a technical staff member at Sandia National Laboratories in Livermore, California. She received her Diploma in Mechanical Engineering from the National Technical University of Athens in Greece, and her MS from Rutgers, The State University of New Jersey. Her research interest is in the area of high performance computational fluid dynamics, with an emphasis on numerical methods and innovative software techniques. Her work can be found in journals and proceedings of international conferences on computational fluid dynamics and parallel computing.

Jaideep Ray is a researcher in Sandia National Laboratories, Livermore. He received his PhD in 1999 from Rutgers, The State University of New Jersey in Mechanical and Aerospace Engineering. His interests lie in computational fluid dynamics, Richtmyer–Meshkov instabilities, and reactive flows, along with various aspects of parallel computing. His papers usually appear in journals and conference proceedings dealing with numerical methods and high performance computing.

Christopher A. Kennedy received his PhD in Applied Mechanics from the University of California, San Diego. He has conducted research on compressible shear layers and jets. To

compliment this, he also worked in high-order, finite difference, spatial discretisations, high-order Runge–Kutta methods for temporal advancement, and physical boundary conditions.

Habib N. Najm a distinguished member of the technical staff at Sandia National Laboratories in Livermore, CA, USA. He received his PhD in Mechanical Engineering from the Massachusetts Institute of Technology, Cambridge, MA, USA. His research interests include the development and utilisation of numerical methods and computational tools for modelling and analysis of reacting flow. He also works on development of numerical methods for uncertainty quantification in chemical systems, and on computational studies of microfluid systems for biodetection applications. He is co-author of over 40 archival journal papers.

1 INTRODUCTION

Reacting flow computations are based on the numerical solution of the mass, momentum (Navier-Stokes), species, and energy conservation equations. These form a time dependent partial differential equation system with requisite initial and boundary conditions. Allowing for both turbulence and detailed chemical kinetics, reacting flow models typically involve a large range of length and time scales, leading to high spatial resolution requirements and severe time integration stiffness. The resulting computations are extremely challenging, even for laboratory-scale flames. As a result, a variety of approaches have been adopted to make flame simulations feasible.

The most obvious way to reduce the computational cost is to use appropriate reduced order models that ameliorate the complexity of the equation system. Approaches like Reynolds averaged Navier-Stokes (RANS) (Wilcox, 1993) and large eddy simulations (LES) (Pope, 2000) are useful for reducing the turbulent flowfield complexity, while the use of simplified chemical mechanisms (Peters and Rogg, 1993) and transport models (Hirschfelder et al., 1954) have proven to be successful in reducing the complexity of the species governing equations. These models are typically validated against experiments and in some simple cases, against direct numerical simulations (DNS). Further, the low Mach number approximation (Williams, 1985), which neglects the acoustic compressibility of the fluid, is very effective in eliminating sonic-speed time scales and their associated time-integrator time-step stability constraints. In the absence of acoustic phenomena (e.g., acoustic instability of flames), high-Mach number flow, and detonations, the low Mach number model has proven very adequate for computations of reacting flow.

High-fidelity DNS reacting flow computations, where all relevant time and length scales are resolved, are computationally intensive, requiring sophisticated numerical and computational techniques for handling two- and three-dimensional (2D and 3D) laboratory scale flames. Our objective is the development of a toolkit for DNS of reacting flows, with particular emphasis on laboratory scale 2D and 3D flames. Our primary interest is in the testing and implementation of new physical models and numerical schemes and the analysis and modelling of flame physics. Thus, the toolkit needs to be efficient, modular and extensible to accommodate the testing of various physical models with parallel and high performance computing. The most important characteristics are modularity, low maintenance requirements, and a code design that hides the complex data structures from the user. We achieve these characteristics through a judicious mixture of new and old software engineering techniques and numerical methods. In this paper we present a description of both, with an emphasis on numerical methods.

Reacting flows (and especially laboratory-sized hydrocarbon flames at atmospheric conditions) exhibit a wide spectrum of length and time scales. Length scales range from roughly 100 μm (Najm et al., 1998) (the width of atmospheric-pressure flame radical profiles) to the flow geometry scale (O(10 cm) for laboratory-scale flames). Assuming that the finest structures need to be resolved with at least ten grid points, we end up with a grid that has 10^4 grid points on each side. Coupled with the fact that detailed-chemistry reacting flow models have O(100) unknowns (mostly species concentrations), the CPU and memory requirements become significant for adequately resolved simulations.

Except for the cases where reaction zones are distributed in the entire domain (e.g., reacting turbulence), most flames involve localised regions of intense chemical and physical activity while the bulk of the domain is largely sedate. These localised flame front regions are sites of intense concentration gradient, and involve large reaction rates and high diffusional transport fluxes. The total spatial extent of the reaction zones is usually small relative to the flow domain length scales. Therefore, if high grid densities are limited to the active regions, one can achieve significant savings in CPU and memory requirements. Since these flames are unsteady, the grid would also need to be *adaptive*, i.e., these 'reactive' zones need to be automatically identified and the grid needs to be adapted to them. Such a technique, *adaptive mesh refinement* (Chung, 2002), is widely used in hydrodynamics, and we employ one of its subcategories called *structured adaptive mesh refinement* or SAMR in the present work.

Structured adaptively refined meshes overlay grids of increasing refinement until the required accuracy is achieved, thus reducing the total number of mesh points compared to uniform mesh approaches. However, this results in a multi-level mesh, i.e., a point in space admits multiple resolutions and, unless a consistent numerical approach is followed, multiple (erroneous) solutions. The

SAMR approach restricts the sub-domains of higher refinement to regions that can be expressed as a disjoint union of rectangular boxes. Berger and Oliger (1984) first proposed this multi-level approach for inviscid supersonic flows (the compressible Euler equation, hyperbolic PDEs which reduce to ODEs after spatial discretisation) which was simplified (by restricting all rectangles to be aligned to each other) by Berger and Colella (1989). This was extended to incompressible inviscid flows (incompressible Euler equation, which reduces to DAEs after spatial discretisations) by Minion (1996), and variable density incompressible viscous flows (incompressible Navier-Stokes equation) by Almgren et al. (1998). The main contribution in (Minion, 1996; Almgren et al., 1998) was the establishment of a consistent way of enforcing the algebraic constraint of the DAE. Pember et al. (1998) added a chemical reaction to the physics and augmented the fluid dynamical equations with coupled equations for the evolution of chemical species. Day and Bell (2000) further refined the approach and have used SAMR to simulate large 2D and 3D flames (Bell et al., 2003a, 2003b; Sullivan et al., 2002).

Timescales in reacting flow vary between ~1 ns (fast chemical reactions) and ~100 ms (diffusion of 'heavy' hydrocarbon species). Since the differential equations modelling flame chemistry are often 'stiff', an implicit time-integration scheme is indicated. However, implementing an implicit scheme on adaptive meshes in a parallel environment is difficult because of the cross-processor coupling caused by spatial derivative stencils at processor boundaries. We address this issue by adopting an operator-split construction (Strang, 1968), where the reactive terms are integrated separately from the convective and diffusive terms. This allows us to choose separate time-integration algorithms for the stiff reactive system and the diffusive systems. This enables us to avoid the excessively tight timestep constraint that a stiff chemical system may pose if integrated with an explicit time integrator.

Operator splitting, as proposed by Strang (1968) is a commonly used procedure to allow the use of specialised/customised integrators for various terms/operators in an equation. It has been used in atmospheric modelling (Spee, 1995; Verwer et al., 1995; Spee et al., 1998) to decouple different operators. Emphasis has been placed on the stability of different schemes (Sheng, 1989; Wright, 1998) and the role of stiffness in stability (LeVeque and Yee, 1990). The identification and control of splitting errors has been a common subject (Spee, 1995; Wright, 1998; Lanser and Verwer, 1999; Ropp et al., 2004) as have been the transients (which occur when one switches from one operator to the other) (Spee et al., 1998) and the consequences of restarting a stiff integrator at each timestep (Verwer et al., 1995; 1999). An application of operator-splitting to address stiff chemical systems found in flames can be found in (Knio et al., 1999). A recent paper by Kværnø (Kozlov et al., 2004) analyses the behaviour of local error in splitting methods (both Strang and Godunov splitting) using Lie groups as well as singular perturbation theory. This was done since the classical theory for the local order of consistency is valid only for stepsizes which are smaller than the ones that are typically used in practice. Though very different, both approaches provide results which are consistent with numerical experiments.

In this paper, we present, for the first time, how high-order spatial discretisations may be used in a SAMR environment, to achieve an accurate yet economical discretisation of a domain. One can find interesting studies and results in the literature regarding high-order spatial discretisations for *single-level* (structured and unstructured) meshes with finite differences (Lele, 1992; Visbal and Gaitonde, 2002; Wang and Huang, 2002), finite volumes (Lilek and Perić, 1995), finite elements (Cockburn and Shu, 1998), and spectral elements (Karniadakis and Sherwin, 1999). However, no published results exist regarding high-order (>2) schemes applied to *multi-level* block-structured adaptive meshes (SAMR). In this paper we address various numerical consistency issues and present a study of the computational savings achieved with fourth-order approaches vis-a-vis second order ones for SAMR meshes. We also adapt an extended stability explicit time-integration strategy to operate in a *time-refined* (see Section 2.2) manner on such meshes. This is coupled with a conventional backward differentiation scheme (BDF) in an operator-split construction to solve a set of coupled nonlinear stiff PDEs. Specifically, we use a 5th order scheme. The order may seem excessively high given the second-order operator-split construction, but BDF5 was empirically observed to converge the fastest when used in our problems.

This work is effectively a prelude to the ultimate aim of creating a toolkit for the simulation of flames using the low Mach-number approximation of the Navier-Stokes equation. It describes and validates the numerical and software techniques that will be employed to create the toolkit. In Section 2, we present an outline of high-order discretisations and a numerical time-integration scheme developed to work on SAMR meshes. This is followed, in Section 3, by an account of the CCA (software) methodology that we use to implement the schemes described in Section 2. In Section 4 we apply and validate our schemes on three test cases modelled as reaction-diffusion systems of increasing complexity. We conclude in Section 5.

2 NUMERICAL METHODS

In this section we present an outline of the SAMR approach and a description of how high-order spatial discretisations and extended stability explicit time-integration schemes are adapted to work efficiently (time-refinement) on such meshes.

2.1 Structured adaptive mesh refinement (SAMR)

SAMR (Berger and Oliger, 1984; Berger and Colella, 1989) is a multiscale algorithm for Cartesian meshes employed to achieve higher grid resolution locally, in regions of the domain where it is required. Numerical simulations of reacting flows and multidimensional simulations in particular, can be very expensive in terms of computing power. The SAMR method can be summarised as follows: a coarse Cartesian mesh (called the Level 0 mesh) is overlaid over a rectangular domain, and, based on a suitably defined error, the grid points which require further refinement are identified. These grid points are flagged and collated into rectangular children patches on which another denser Cartesian mesh (called Level 1) is imposed. This is done recursively, to create a grid hierarchy (GH). Figure 1 shows one such GH. The refinement factor between the parent and the child mesh is usually kept constant for a given problem. In this paper, the refinement ratio is two. The more accurate solution from the finest meshes is periodically interpolated onto the coarser ones.

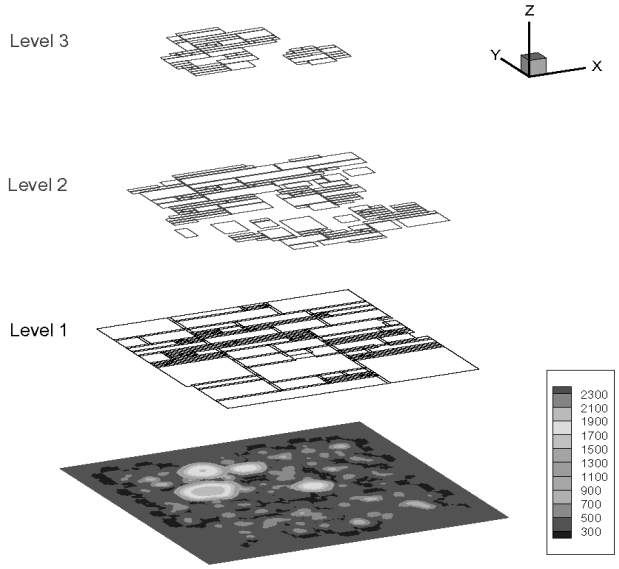

Figure 1 *A 4-level Grid Hierarchy, taken from a simulation of an igniting H_2–Air mixture. The temperature field is plotted on the Level 0 mesh (the coarsest mesh). Higher levels of refinement are shown as a disjoint union of boxes containing meshes of increasing fineness*

Although SAMR techniques are fairly simple and straightforward conceptually, in practice there are issues associated with their implementation. For example, in explicit schemes the global timestep is restricted by the stability criterion on the finest mesh resulting in a very small global timestep. To address this issue one can use time-refined explicit schemes (Berger and Oliger, 1984) with hierarchies of patches, where each patch is sub-cycled with a different stable timestep. A description can be found in Section 2.2. Further, SAMR simulations involve temporal and spatial interpolations between different levels. Figure 2 shows a 1D example. The solution Φ^n on Level 0 (L0) is advanced first to Φ^{n+1}. Boundary conditions are used to evaluate the 'domain boundary points', marked as ovals on L0 in the figure and their value is used to evaluate spatial derivatives near the boundaries. When the same time-integration step is repeated at L1, the 'patch boundary points' (shown as ovals on L1) are interpolated from Φ^n from the coarse mesh. This coarse-to-fine interpolation is called *prolongation*. Since explicit time-integration schemes are subject to a constraint on Δt (because of stability), L1 is sub-cycled twice with $\Delta t_1 = \Delta t_0/2$, i.e., $\Phi^n \rightarrow \Phi^{n+1/2} \rightarrow \Phi^{n+1}$, assuming a convection stability constraint. During the second cycle, $\Phi^{n+1/2} \rightarrow \Phi^{n+1}$, the 'patch boundary points' (ovals on L1) require interpolated data from L0, but at time $t = \Delta t_0/2$. This is obtained by first *spatially* interpolating both Φ^n and Φ^{n+1} from L0 to the oval point and then *temporally* interpolating between the Φ^n and Φ^{n+1} interpolated at the oval point. Interpolations from fine-to-coarse meshes are also done to synchronise data between fine and coarse meshes. The fine-to-coarse interpolation is called *restriction*. See Section 2.2 for details. Spatial interpolations add a significant cost to SAMR approaches, especially in 3D.

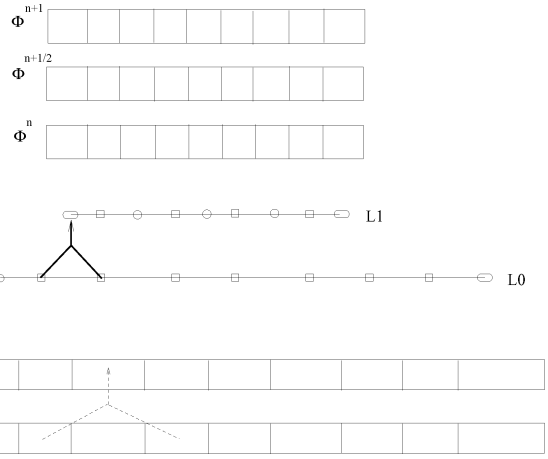

Figure 2 *A 1D example of interpolations that occur in SAMR simulations. In order that one may avoid skewed spatial derivative stencils, boundary conditions are used to evaluate the 'domain boundary points' (ovals on L0). On L1, the 'patch boundary points' (ovals on L1) are interpolated from L0. For linear interpolations, the dependence of the L1 oval on L0 grid points are shown with arrows. Time sub-cycling is shown by the requirement to solve for 1 solution on L0 (Φ^{n+1}) and 2 on L1 (Φ^{n+1} and Φ^{n+1})*

The criteria for adaptation (refining and coarsening regions in space) pose a challenge in most adaptive simulations. While a rigorous approach based on Richardson extrapolation is outlined in Berger and Colella (1989), it is expensive and is frequently replaced by heuristic work-arounds, thus creating a degree of uncertainty in the accuracy of the solutions. Further, the use of SAMR on parallel computers poses significant load balancing problems. It poses the requirement of keeping parents and children on the same processor to avoid sending prolonged and restricted data for entire patches over the network. Thus, the distribution of computational load (as a function of space) becomes extremely non-smooth, i.e., regions of

multiple levels of refinement have high loads because load scales exponentially with the level of refinement, and one is posed the task of partitioning such a domain into equally loaded parts. Unless the L0 mesh is quite fine (i.e., the partitioning can be achieved in a fine-grained manner), this is extremely difficult, resulting in poor domain-partitioning. Since L0 is typically very coarse, severe load-imbalances are quite common when operating with deep grid hierarchies. There are two possible approaches in order to avoid deep hierarchies: the first is to increase the refinement factor (e.g., instead of having a refinement factor of two, have a factor of four), thus reducing the required levels of refinement; the second is to implement a high order method to discretise the spatial derivatives thus obtaining higher accuracy with coarser grids, i.e., with fewer levels of refinement. The first approach reduces the hierarchy depth but does not appreciably reduce the number of grid points, and hence we chose to explore the second approach.

To illustrate the potential efficiency gains of high-order spatial discretisations, one may consider first-derivative operators. The qualitative conclusions for derivative operators may then be applied to interpolation operators provided Runge phenomena are not experienced. Jameson (2000) has attempted to quantify the relative efficiencies of different orders-of-accuracy. Table 1 lists the number of grid points needed by explicit, centred derivative operators of various orders to resolve a particular wavenumber mode to a given error tolerance. Even a modest increase in order (from 2nd to 4th) drops the resolution requirements in 1D by a factor of 3 for a 10^{-2} error tolerance; for tighter tolerances a factor of 20 is achieved; and at higher orders, one achieves almost 2 orders of magnitude savings in resolution. Given that these savings will be squared and cubed as one proceeds to 2D and 3D respectively, the advantages of a high-order scheme become evident.

Table 1 *Number of grid points per wavelength required for a chosen error tolerance ε when the first derivative of a sine wave is computed using an explicit central difference stencil of 2nd, 4th, 6th, 8th and 10th order (2E, 4E, 6E, 8E and 10E respectively)*

ε	2E	4E	6E	8E	10E
$10^{-2.0}$	25.65	8.344	5.712	4.696	4.156
$10^{-2.5}$	45.53	11.22	7.012	5.516	4.753
$10^{-3.0}$	81.07	15.03	8.572	6.444	5.412
$10^{-3.5}$	144.1	20.20	10.44	7.507	6.136
$10^{-4.0}$	256.5	26.85	12.69	8.727	6.943
$10^{-4.5}$	455.3	35.70	15.44	10.12	7.834
$10^{-5.0}$	816.0	47.60	18.70	11.72	8.837

The use of high-order stencils is not without its drawbacks. High-order stencils involve more computations to evaluate a derivative. In addition, derivative schemes above fourth-order may best be closed to a lower order at the domain boundaries to remain time-stable. Further, they need to be coupled with high-order prolongation and restriction operators to preserve the accuracy at all levels of the GH. While the increased computational cost is relatively inconsequential with derivative operators, interpolation operators are evaluated by using a linear combination of as many as p^d (where p is the order of accuracy and d is the dimensionality) values and their respective coefficients. In an explicit time integration procedure, the larger eigenvalues of the high-order derivative operator matrix may reduce the maximum step-size by a third or so. However, the sparser grid (small data size) of a high-order formulation will benefit tremendously from the higher cache-hit rate and it is unclear whether the 'time-to-solution' of a high-order scheme will be vastly different.

2.1.1 High-order discretisations

A high-order spatial discretisation for vertex-centred AMR with a refinement factor of two requires derivative, coarse-to-fine interpolant, and filter operators. In this paper, we investigate 4th order discretisations. All derivatives used in our work use first-derivative operators, applied repeatedly if necessary. For example, a fourth-order first derivative evaluated at (i) on a uniform mesh is of the general form:

$$f_i' = \frac{c_L f_{i-3}}{(\Delta x)} + \frac{b_L f_{i-2}}{(\Delta x)} + \frac{a_L f_{i-1}}{(\Delta x)} + \frac{\Upsilon f_i}{(\Delta x)} + \frac{a_R f_{i+1}}{(\Delta x)} + \frac{b_R f_{i+2}}{(\Delta x)} + \frac{c_R f_{i+3}}{(\Delta x)}. \quad (1)$$

where $\{a, b, c\}$ are stencil coefficients for the points right (R) and left (L) of (i) and Υ is the i-point stencil coefficient. Several examples of varying accuracies are listed in Table 2. At the domain boundaries, the derivatives can be calculated either by creating a halo of grid points around the domain and using the same stencils or by closing, possibly to lower order, using skewed stencils. We use the second approach because of its better stability characteristics. Table 3 lists several possible stencils. The approach for high orders is identical.

Table 2 *Stencil coefficients for centred (E) and upwinded (U) first-derivative operators of different orders of accuracy on uniform grids. S is the name of the stencil, LOTE is an abbreviation of Leading Order Truncation Error and ξ is a scaled wavenumber. O(p) stencils have $O(\xi^{p+1})$ LOTE. These results were obtained for a uniform mesh*

S	c_L	b_L	a_L	Υ	a_R	b_R	c_R	LOTE
2E	0	0	–1/2	0	1/2	0	0	$-(1/6)\xi^3$
3U	0	0	–1/3	–1/2	1	–1/6	0	$-(1/12)\xi^4$
4U	0	0	–1/4	–5/6	3/2	–1/2	1/12	$+(1/20)\xi^5$
4E	0	1/12	–2/3	0	2/3	–1/12	0	$-(1/30)\xi^5$

Table 3 *Stencil coefficients of skewed lower-order, first-derivative operators near boundary points. These can be used with fourth-order derivative discretisations. These results were obtained for a uniform mesh*

c_L	b_L	a_L	γ	a_R	b_R	c_R	LOTE
0	0	0	–1	1	0	0	$-(1/2)\xi^2$
0	0	0	–11/6	3	3/2	–1/3	$+(1/4)\xi^4$
0	0	–1/3	–1/2	1	–1/6	0	$-(1/12)\xi^4$
0	0	–1	1	0	0	0	$+(1/2)\xi^2$
–1/3	3/2	–3	11/6	0	0	0	$-(1/4)\xi^4$
0	1/6	–1	1/2	1/3	0	0	$+(1/12)\xi^4$

As mentioned in Section 2.1, SAMR simulations periodically *prolong* data from coarse meshes to a halo of points around their finer children so that centred stencils may be used throughout. This process is skipped on a patch boundary if it abuts a domain boundary. More accurate data are also *restricted* from a finer patch to its coarser parent. In two-dimensions, interpolation stencils take data off from a four-squared block of points (for fourth-order interpolations) and interpolate a value onto the geometric centre of a cell. Within the interior of the domain, the interpolated point is the geometric centre of the block but near physical boundaries, the block will skew relative to the interpolation point. Whereas second-order coarse-to-fine interpolation requires no boundary closure, fourth-order does. At second-order, there is one stencil coefficient, i.e., for the points $(i \pm 1/2, j \pm 1/2) = 1/4$. At fourth-order, there are three unique stencil coefficients; $(i \pm 1/2,\ j \pm 1/2) = 81/256$, $(i \pm 1/2, j \pm 3/2) = (i \pm 3/2, j \pm 1/2) = -9/256$, and $(i \pm 3/2,\ j \pm 3/2) = 1/256$. At higher order, the number of unique stencils increases.

Filtering (Kennedy and Carpenter, 1994) is a potential way to cleanly remove high-wavenumber information from the grid. There are two compelling reasons to filter. The first is that any finite difference numerical method has limited accuracy, therefore higher-wavenumber information that is unresolvable by the numerical method needs to be removed before it interferes with the resolved wave-numbers. Secondly, in order to avoid failure of the interpolant operators, no wavenumber that represents less than approximately six grid points may be present (Trefethen and Weideman, 1991). Filters on a vector f^{old} are implemented as

$$f^{\text{new}} = (I - \nu D) f^{\text{old}}$$

where D is a discrete operator corresponding to a high order spatial derivative, f^{new} is the filtered vector, I is the identity matrix, $\nu = (-1)^n 2^{-2n}$ and $2n$ is the order of the filter. Filters are an important element in the overall numerical method but significant care has to be exercised when using them, since the use of a filter is approximately equivalent to adding a high order spatial derivative to the PDEs being solved. This can lead to non-physical results. We use 8th order filters in our study.

To investigate if high order spatial interpolation approaches with SAMR are indeed more economical, we solve a model test case, the FitzHugh-Nagumo (Fall et al., 2001) (FN) equation in Section 4.1. FN has an analytical travelling wave solution which simplifies the evaluation of errors. We use 2nd and 4th order discretisations on SAMR meshes and compare against the theoretical convergence rate. A variety of interpolants are used to determine the appropriate (discretisation, interpolant) pair, and a computational cost analysis is performed.

2.2 Time integration strategy

We are interested in PDEs which are of the general form:

$$\frac{\partial \Phi}{\partial t} = \mathbf{F}(\Phi, \nabla\Phi, \nabla^2\Phi, \ldots) + \mathbf{G}(\Phi) \tag{2}$$

where Φ is the vector of unknowns (usually temperature and the concentration of species). If $\mathbf{F}$ in equation (2) includes spatial derivatives where each point in linked to its neighbours, an implicit integration approach leads to the formation, after linearising, of an $[\mathbf{A}]\mathbf{x} = \mathbf{b}$ linear system where $\mathbf{x}$ includes Φ for all mesh points. Solving such a system on a decomposed (across processors) domain is non-trivial. Conversely, if no spatial derivatives are included, $\mathbf{x} = \Phi$, for a mesh point which can be solved rather easily, mesh point by mesh point. Details of implicit and explicit integrators can be found in (Tannehill et al., 1997).

$\mathbf{G}$, in equation (2), is stiff (the ratio of the largest and the smallest eigenvalues of $\partial\mathbf{G}/\partial\Phi$ is large) while $\mathbf{F}$ is non-stiff. We employ the following technique, based on operator-splitting (Strang, 1968) to time-advance equations like equation (2). For a given timestep (called global or coarse mesh timestep) Δt_c, advancing from t^n to $t^{n+1} = t^n + \Delta t_c$:

- Integrate $\Phi_t = \mathbf{G}(\Phi)$. The initial condition is Φ^n and the output is kept in $\tilde{\Phi}$. This is done using an implicit integrator (Cvode, Cohen and Hindmarsh, 1994), on a point-by-point basis, for a timestep of $\Delta t_c/2$.
- Integrate $\Phi_t = \mathbf{F}(\Phi, \nabla\Phi,\ \nabla^2\Phi, \ldots)$ using an explicit integrator (Runge-Kutta-Chebyshev, Shampine et al., 1998). The initial condition for the step is $\tilde{\Phi}$; the output is kept in $\hat{\Phi}$, integrated over a period of Δt_c.
- Integrate $\Phi_t = \mathbf{G}(\Phi)$ again. The initial condition is $\hat{\Phi}$ and the output is kept in Φ^{n+1}. This is identical to step 1 and is done for a timestep of $\Delta t_c/2$.

This procedure for splitting the stiff chemical operator $\mathbf{G}$ from the non-stiff spatial transport operator $\mathbf{F}$, to form an **FGF** sequence was outlined by Knio et al. (1999). The present construction, employing a **GFG** sequence, follows Sportisse (2000) by letting the stiff operator be the last in the time step, in order to achieve higher accuracy in the data reported at the end of the time step.

Explicit integrators suffer from a stability restriction (due to the transport term $\mathbf{F}$) such that the maximum Δt_c depends on the mesh density. Thus, fine patches need to be

stepped at finer temporal resolutions than coarser ones; consequently for every step of a coarse patch, its (higher resolution) children undergo multiple steps. In our case, children patches take R steps for each step of the parent patch, R being the factor of refinement (2 in this study). This is best illustrated in 1D. In Figure 3, we see a three level GH, with a refinement factor of 2. As per the CFL criterion, the stable timestep Δt will also vary by factors of 2. Level 0 is integrated first with Δt_c. This is followed by one integration of Level 1 with $\Delta t_1 = \Delta t_c/2$ and another one on Level 2 with $\Delta t_2 = \Delta t_1/2$. Since there are no more levels to recurse to, a second integration of Δt_2 is done on Level 2. This brings Level 1 and 2 solutions to the same point in time (Δt_1) and the (accurate) solution from Level 2 is *restricted* (fine-to-coarse interpolation) to Level 1. The same process is repeated to advance Level 1 up to Δt_c, with Level 2 undergoing two more integration phases. Thus the integration sequence is ABCCBCC (where A, B and C are the meshes on Level 0, 1 and 2 respectively), the pre-order traversal of a binary tree. The entire process of sub-cycling different patches at different timesteps is called *time refinement*. This sub-cycling increases the complexity of the time-integration scheme substantially and incurs significant recursion overheads if the GH is deep (large number of refinement levels).

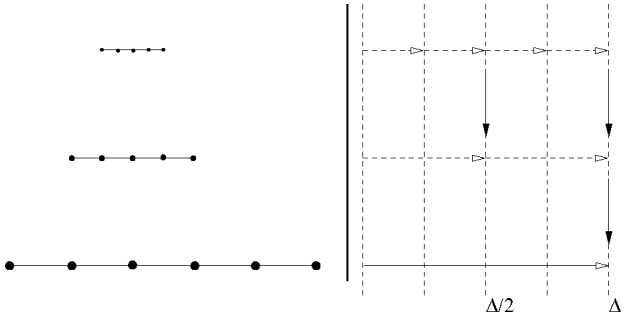

Figure 3 *Time-refinement/sub-cycling strategy to avoid the timestep constraint imposed by the finest mesh. Finer levels are subcyled (integrated more frequently but with smaller Δt) and the solution is synchronised between (coarse) parent and (fine) children using restrictions (fine-to-coarse interpolations)*

The patch hierarchy is periodically recreated. The solution is passed through a filter (with an appropriately defined error criterion) to determine regions needing finer meshes, whereby new patches are created and initialised with prolonged data from the coarse meshes (provided there does not exist a fine patch over that subdomain, wholly or partly). Regions which are deemed over-refined have their fine patches destroyed.

2.2.1 Runge-Kutta-Chebyshev explicit integration

Shallow grid hierarchies benefit from explicit time-integrators capable of large timesteps. Given an initial value problem $\Phi_t = \mathbf{F}(\Phi)$, one can design Runge-Kutta schemes with extended stability regions along the negative real axis, i.e., schemes which are suitable for problems where the eigenvalues of $\partial\mathbf{F}/\partial\Phi$ are near the negative real axis. We use one such second-order Runge-Kutta-Chebyshev (RKC) scheme (Shampine et al., 1998) with a stability region which grows approximately as $0.6s^2$ along the negative real axis, where s is the number of stages in the scheme. Local error analysis (Shampine et al., 1998) reveals that the error term over a timestep τ is of $O(\tau^3)$, but is also a function of s. Convergence results are in (Verwer et al., 1990). Briefly, each timestep can be written as:

$$
\begin{aligned}
&\mathbf{Y}_0 = \Phi^n \\
&\mathbf{Y}_1 = \mathbf{Y}_0 + \tilde{\mu}_1 \tau \mathbf{F}_0 \\
&\mathbf{Y}_j = (1-\mu_j-\nu_j)\mathbf{Y}_0 + \mu_j \mathbf{Y}_{j-1} + \nu_j \mathbf{Y}_{j-2} \\
&\qquad + \tilde{\mu}_j \tau \mathbf{F}_{j-1} + \tilde{\gamma}_j \tau \mathbf{F}_0, \quad j = 2 \ldots s \\
&\Phi^{n+1} = \mathbf{Y}_s
\end{aligned}
\tag{3}
$$

where the time-advancement of Φ^n to Φ^{n+1} is over a timestep τ. Note that $\tilde{\mu}_j$, $\tilde{\gamma}_j$, μ_j and ν_j are known analytical functions of s. The stage timestep size varies as $\Delta t_j/\tau \approx (j^2-1)/(s^2-1)$, where j, is the stage number. The stability region for this scheme occupies a fairly narrow region around the negative real axis (in the complex stability plane). Thus, if the system being integrated has eigenvalues with a significant imaginary component, it is unlikely that RKC will be very advantageous. However, in most reactive diffusive-advective problems of interest this has not been the case, and RKC-based approaches have worked quite well (Najm and Knio, 2003; Lanser and Verwer, 1999). Recently, RKC has also been coupled with an implicit-explicit (IMEX) scheme (Verwer and Sommeijer, 2004) to solve reaction-diffusion equations. Within the setting of linear stability theory, the incorporation of IMEX was not seen to affect the stability region of the RKC scheme.

The extension of a traditional explicit Runge-Kutta method (second order) for time-refined stepping on SAMR grids was outlined in (Berger and Oliger, 1984). The extension of RKC is conceptually similar. Each RKC stage is first-order, requiring only linear (temporal) interpolation (from a parent mesh's data) at the child patch edges (where it becomes impossible to evaluate centred spatial stencils due to lack of grid points on one side). Implementation details involve significant book-keeping and temporal interpolations from coarse to fine meshes based on the stage number of the integration process on the fine mesh.

We used RKC (modified for time-refined time-stepping on SAMR meshes) to solve $\phi_t = \mu\nabla^2\phi$ on a unit square with zero gradient boundary conditions and a Gaussian distribution ($\phi_0(\mathbf{x}) = \exp(-r^2/\delta^2)$, $\delta = 0.05$, where $r = |\mathbf{x} - \mathbf{x}_0|$, $\mathbf{x}_0 = \{0.5, 0.5\}$) for the initial condition. A second order discretisation was chosen. The region $0.3 \le x \le 0.7$, $0.3 \le y \le 0.7$ was refined. Two levels of refinement (i.e., a 3-level GH) were allowed. The timestep size Δt was much smaller than the stable limit of an 8-stage RKC. We evaluate the error vis-a-vis the analytical solution

$$\phi(\mathbf{x},t) = \frac{\delta^2}{\delta^2 + 4\mu t}\exp\left(-\frac{r^2}{\delta^2 + 4\mu t}\right)$$

and plot it in Figure 4 for a range of Δt. We see that the error convergence is second-order in time.

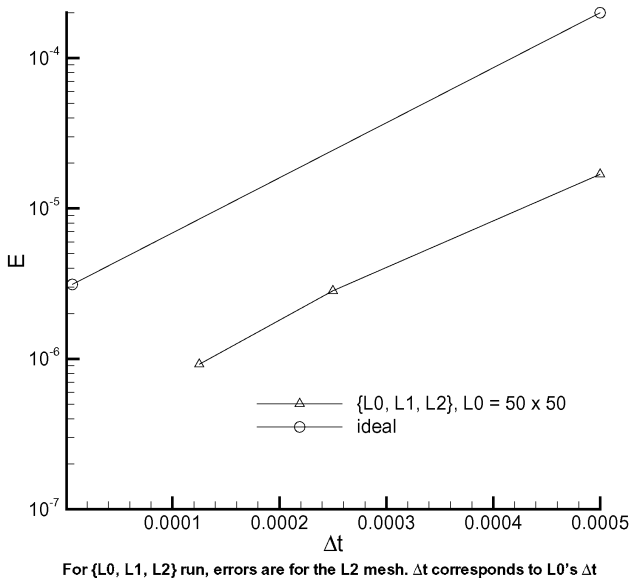

Figure 4 *Convergence of the temporal error is the solution of the heat equation. A second-order spatial discretisation was used on a mesh with two levels of refinement. Errors show 2nd order convergence in time*

In Section 3 we describe the software architecture within which these numerical algorithms are implemented. Emphasis is on high performance and modularity, so as to enable plug-and-play and promote experimentation with algorithms.

3 SOFTWARE METHODOLOGY

Our goal is to design a flexible, modular, reusable toolkit. A valuable pointer for achieving modularity in scientific computing comes from commercial practices. The business world has implemented modularity by adopting the *component* model (e.g., Visual Basic, http://msdn.microsoft.com/vbasic, CORBA, http://www.omg.com and Java Beans, Englander and Loukides, 1997). In this model an object implements a functionality, which is exploited via interfaces; an object's adherence to a specification (dictated by the component model) transforms it into a component within the framework. Components are peers, i.e., they do not inherit from other components, and are easily extensible since components implementing an agreed to, well defined interface can be developed in complete isolation. While component-based software design enhances the cooperative development of applications, the commercial model is unsuitable for scientific computing (Allan et al., 2002), the main drawbacks being high latency and lack of support for parallel (not distributed) computing. The CCA (Common Component Architecture) component model was designed (Armstrong et al., 1999) to meet the high performance requirements of scientific computing; to date three CCA-compliant framework implementations have been demonstrated (Ccaffeine, Allan et al., 2002, Uintah, de St. Germain et al., 2000 and XCAT, http://www.extreme.indiana.edu/ccat/). The CCA standard is flexible; while allowing an evolutionary path forward for new scientific applications, it is also an efficient way of 'wrapping' legacy codes.

In the following we present how one identifies a class of significant computational scientific problems, involving realistic physical models, non-linear PDEs and a spectrum of time and length scales, and creates a CCA-compliant component-based software infrastructure to solve them. This infrastructure consists of *scientific components*, where each component has a distinct functionality in terms of physical modelling or implementation of a numerical algorithm. Components are assembled into component assemblies to solve numerical problems. Since most complex problems are solved in a hierarchical manner, component assemblies (for simpler problems) appear as sub-assemblies in the solution strategies for larger, more complex ones. This ease of reuse results in significant savings in programming effort and promotes experimentations with unconventional approaches, often contributed by experts. In this section we outline a description of the Common Component Architecture and the design rationale of our components. Section 4 illustrates how components are reused and two competing solution strategies may be tried out with little difficulty.

3.1 The CCA component model

The CCA model (Armstrong et al., 1999) uses the *provides-uses* design pattern. Components *provide* functionalities through interfaces that they export; they *use* other components' functionalities via interfaces. These interfaces are called *Ports*; thus a component has ProvidesPorts and UsesPorts. Components are peers and are independent. They are created and exist inside a framework; this is where they register themselves, declare their UsesPorts and ProvidesPorts and connect with other components.

Ccaffeine (Allan et al., 2002) is the CCA framework that we employ. It is a low latency framework for high performance (parallel) scientific computations. Components can be written in most languages within the framework; we develop most of our components in C++ or as sets of C or F77 libraries wrapped in C++. Every component is compiled into a shared object library, i.e., a dynamically loadable library. Most of the Ports we implement are domain specific and their design is a result of agreed-to interfaces.

One of the features of dynamically loadable libraries is that when loaded into memory, their symbols (variable and subroutine names) are preserved in a separate namespace. This drastically reduces the chances of symbol conflicts and erroneous symbol resolution at runtime. This feature is extremely helpful when one integrates a legacy software package into an existing scientific code. In its absence, one would have to resolve the conflicts manually, an exercise prone to error if the package being modified is ill-understood.

3.2 Design of components and interfaces

In this section we demonstrate the rationale for the recursive decomposition of the solution strategy of equation (2) into software subsystems and the interfaces developed. A *software subsystem* is a collection of components that embodies a physical or numerical functionality (e.g., an *explicit integration subsystem* includes the time integrator, the RHS evaluator and miscellaneous components that identify the largest eigenvalue of the discretisation matrix to enable dynamic time-step sizing). The software subsystems we identified are:

- *Mesh*. It serves as a means of declaring and maintaining patches in the mesh hierarchy. It is geometric in nature, and determines and administers the child-parent-sibling relationships and the spatiotemporal location of patches. Load balancing and domain decomposition functionalities are implemented here.
- *Data object*. It maintains the collection of arrays which contain data declared on patches, one array per patch. Typically a number of related variables are stored together in a data object; equally typically, a simulation would contain 2–3 data objects. This subsystem implements the actual movement/copying of data between patches and the packing/unpacking of data before/after message passing. Currently we have wrapped GrACE (http://www.caip.rutgers.edu/TASSL/; Parashar and Browne, 2000) into a CCA component to perform the data object and the mesh tasks.
- *Initial condition*. This subsystem consists of a set of components that impose initial conditions on a data object.
- *Explicit integration subsystem*. It consists of a recursive time integrator that advances a set of data objects over a time step as well as components that evaluate and assemble the right hand side (RHS), one patch at a time. The evaluation of the RHS can be done by one component or by a further subsystem of components. This also contains components that analyse the field to determine an approximation of the highest eigenvalue that the integrator will encounter. This information is used by the integrator to dynamically adjust the timestep.
- *Implicit integration subsystem*. This consists of an implicit time integrator, which advances a vector of variables, RHS component(s), and an adaptor that collates data from a patch to a vector.
- *Interpolation components*. These implement various spatial and temporal interpolation operators.
- *Boundary condition*. It is applied on a patch by patch basis. BCs are applied at each of the stages of a multi-stage integration scheme; hence application of the boundary conditions has to be done on a finer basis than one data object at a time. Thus, the granularity will be a patch.
- *Database components*. These components store certain parameters (e.g., mesh size, gas properties, etc.) that are retrieved using a key-value pair mechanism. They are essentially maps between the (character string) property name and a number.
- *Adaptors*. Depending on the physical problem at hand, case-specific adaptors are often used to consolidate and filter outputs from various physics components.

Given the functional description above, it is clear what types of Ports (interfaces) are needed:

- Port(s) (provided by the mesh component) that allow
  - geometrical manipulation of the domain,
  - the declaration of fields on the mesh (via data objects)
  - tasks like setting/querying of domain-decomposition details.

  Our design for such a Port is called MeshPort (http://www.caip.rutgers.edu/~jaray/CCA/documentation.html).
- An abstract interface for the data object allowing manipulation of patches and the data defined on them.
- Ports that accept an array of data objects and act on them in a synchronised manner. Integrators usually support these ports.
- Ports that accept an array from a patch.
- Ports that accept vectors.
- Ports that allow setting/querying of key-value pairs.

A detailed description of our software design can be found in Lefantzi et al. (2003).

In Sections 4.2 and 4.3 we will show the use and reuse of a set of components developed for solving mathematical systems like equation (2). These relate, physically, to a 0D/homogeneous ignition, and ignition in a two-dimensional (2D) reaction–diffusion system. We will also show how various changes in the capability of simulation codes may be achieved by simply replacing/substituting components, thus illustrating the power of a modular and recomposable software strategy.

4 TEST CASES

In this section we will seek answers to the following questions:

- Can high-order stencils be used in a SAMR setting to achieve higher accuracies? How should discretisations of a certain order be paired with interpolations to achieve the desired accuracy? Can high order methods

be used to reduce run-times, i.e., are they economical vis-a-vis second-order approaches?
- What design and algorithmic principles need to be followed so that one achieves reusable software modules? Can a component based software methodology be shown to be flexible enough for scientific computing?
- Is the plug-and-play nature of CCA components powerful enough to enable the testing of numerical approaches/algorithms in non-trivial problems in a relatively straightforward manner?

To answer these questions, we conduct the following tests:

- We solve a model reaction-diffusion equation (FitzHugh-Nagumo equation) on SAMR meshes and compare the numerical results with the exact solution. Convergence of error with the effective grid resolution (finest Δx) will be used to check if high-order accuracy has been achieved.
- A 0D ignition simulation of a CH_4–Air mixture using CCA components on 1 CPU. This test case will demonstrate a minimal set of components for treating chemical reactions.
- A 2D parallel simulation of H_2–Air reaction-diffusion ignition using SAMR, extended stability time-integration and high-order schemes. This will use the chemistry-related component developed for the 0D simulation. Further, second-order and fourth-order spatial discretisations will be used in separate runs.

4.1 FitzHugh-Nagumo equation

We solve the following 1D problem:

$$U_t = DU_{xx} + AU(1-U)(U-\alpha) \tag{4}$$

subject to the boundary condition $U_x = 0$ at $x = \pm\infty$. This equation admits an analytical travelling wave solution:

$$U(\xi,t) = \frac{1}{2}\left(1+\tanh\frac{\xi}{2\varepsilon}\right)$$

where $\xi = x + st$, and

$$\varepsilon = \sqrt{\frac{2D}{A}} \quad \text{and} \quad s = \sqrt{\frac{AD}{2}}(1-2\alpha)$$

Details of the derivation are in (Fall et al., 2001). We chose $D = 1.0$, $A = 2 \times 10^4$ and $\alpha = 0.3$, giving $\varepsilon = 10^{-2}$ and $s = 40$.

The problem is solved numerically in a $0 \le x \le 1$ domain discretised with a coarse (Level 0) mesh of 100 points. The front is initialised at $x = 0.5$. The region $0.26 \le x \le 0.61$ is covered by a Level 1 patch and $0.3 \le x \le 0.55$ by a Level 2 patch. The problem is integrated, in a time-refined manner, using a second-order explicit Runge-Kutta scheme (Heun's method), with a Δt on the coarse mesh of 6.25×10^{-7}, chosen so that the $O((\Delta t)^2)$ temporal errors are far smaller than the spatial errors. The solution is advanced up to $t = 1.25 \times 10^{-3}$, corresponding to a 5ε traversal of the wave. Figure 5 shows the profile on the Level 0 mesh at the start and at the end of the run. We see that the front is defined by about ten grid points on the coarse mesh and better as higher level patches are added. The refinement factor is two.

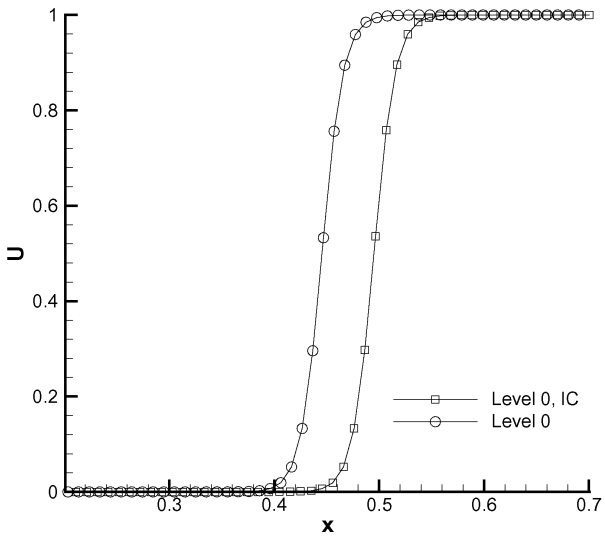

Figure 5 *Profile of U at t = 0, 1.25 × 10^{-3} (graphs with □ and o, respectively) on the coarse mesh. The data at the Level 0 mesh points are marked with symbols; the levels refine by a factor of two. This particular run had three levels and the duration of integration was chosen to that the front would move a distance equal to 5ε, as shown*

In Figure 6 we plot the RMS error with respect to the analytical solution on the individual levels on the grid hierarchy. We see that second and fourth-order convergence is obtained. No filtering is done; with ten coarse grid points in the front, the Runge phenomenon was not observed. In Figure 6 we see that, as expected, the fourth-order simulation is far more accurate than the second-order one and the difference in accuracy increases dramatically with resolution. Further, a given level of error (a horizontal line in Figure 6) can be achieved with a coarser mesh (fewer levels of refinement) when the fourth-order (as opposed to a second-order) approach is employed. Thus, if numerical accuracy, to the exclusion of everything else, is the objective, a high-order approach is the obvious choice.

However, the choice of a numerical scheme has to be tempered with its cost. We measured the floating point operation count on Intel Pentium III processors with 256 kB L2 cache and running at 1 GHz. The interpolation and discretisations were written in Fortran77 and compiled with Portland Group F77 compiler pgf77. The overarching control code was in C++ compiled with the GNU suite of compilers gcc-2.96. The problems were run on the IA-32 cluster at NCSA (platinum.ncsa.uiuc.edu) and the floating point operations were measured using PAPI (Browne et al., 2000).

Figure 7 shows the total number of floating point operations, normalised by the count from a 1-level

second-order run, as a function of the RMS error. This data obtained from the runs is plotted in Figure 6. It is clear, that for this problem, the fourth-order simulation is *cheaper* than the second order approach, and it gets progressively *more* economical as the error tolerance becomes more stringent.

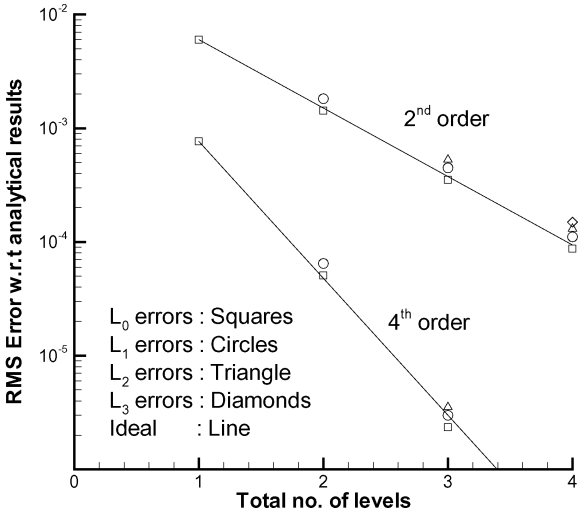

Figure 6 *The RMS error on the individual levels, as the simulation is run on a 1-, 2-, 3- and 4-level grid hierarchy. The errors on levels 0 to 3 are indicated by □, ○, △ and ◊. The ideal behaviour is given by the solid line. Results for both second and fourth-order discretisations are shown. A 1-level grid hierarchy has an effective resolution of 100 in the [0,1] domain; 2-, 3- and 4-level grid hierarchies are correspondingly equivalent to a 200-, 400- and 800-grid point uniform mesh*

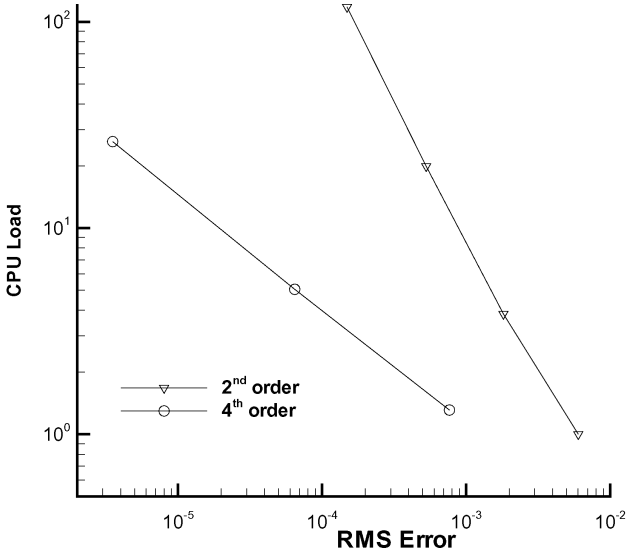

Figure 7 *The computational load vs. RMS error for the second and fourth-order approaches. Results have been normalised by the computational load of a second-order, 1-level grid hierarchy run (1,208,728,001 floating point operations). It is clear that for this problem, the fourth-order simulation is cheaper than the second-order approach, and it gets progressively more economical to use the high-order approach for a given level of error as the error tolerances become stringent*

Note that, if one chooses to operate in a marginally resolved manner, approaching the Trefethen and Weideman limit (Trefethen and Weideman, 1991), the large errors resulting from the two approaches may very well be comparable (see left extreme of Figure 6). Further, judging from the right limit of Figure 7, the high-order approach would not offer much economy either.

We now address the issue of the correct choice of interpolant with a given discretisation. Since U on each grid is eventually the result of interpolation based on local polynomials, one cannot differentiate this interpolated data indefinitely. Interpolated data of order p_I which is differentiated k times will not preserve its original accuracy and consequently care has to be exercised to ensure that the final differentiated term does not fall below the desired accuracy.

We solve the same problem, but now only the region $0.4 \le x \le 0.49$ is covered by a fine mesh. As mentioned before (Section 2.1.1) interpolants are used to initialise a halo of cells at the edges of the fine mesh so that centred discretisations may be used everywhere. These interpolation errors manifest themselves in the right hand side (RHS) term of the equation. It thus becomes necessary that these interpolation-generated errors be comparable to the discretisation error so that their effect is observable in the evaluation of the RHS. In order to do so, we terminate the fine mesh at $x = 0.49$ so that these interpolations are performed in a region of high gradient (and consequently large errors). The correct choice of interpolant will ensure that the order of convergence of the interpolation errors is equal to or larger than that of the discretisation. The error introduced by this interpolation changes as the wave moves into the fine mesh and the interpolation is done in a region of small gradient. For that reason during our tests we integrate for only a small period in time, so that the gradient at the edge of the fine mesh remains large. Further, the interaction of interpolation and discretisation errors occurs only in the U_{xx} term, which in certain regions may be overwhelmed by the *reaction term* $AU(1-U)(U-\alpha)$. Therefore, in order to measure convergence, we measure the error in U_{xx} vis-a-vis the exact solution. Experiments are done on a hierarchy with two levels of refinement. Errors are calculated (for convergence testing) by varying the coarse mesh resolution.

In Figure 8 we plot the convergence of the U_{xx} term as a function of the resolution of the coarse mesh when different discretisation and interpolant pairs are used. For fourth-order discretisations (Figure 8) 6th and 8th order interpolants are seen to preserve the order. We did not use odd-ordered interpolants to restrict ourselves to dissipative errors. Thus, for problems where the largest spatial derivative is of order 2, the sufficient condition for p_Dth order convergence appears to be $p_I \ge p_D + 1$. A similar study, done with 6th order discretisations (Lefantzi et al., 2003), also corroborates this condition. Further, the *refluxing* done in (Berger and Colella, 1989; Almgren et al., 1998) is omitted here and the overall scheme is thus non-conservative at the boundaries of coarse and fine patches.

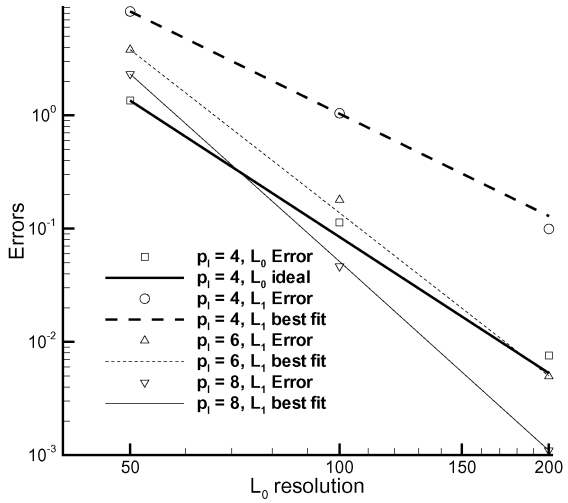

Figure 8 *We plot the RMS error on L_0 and L_1, as a function of the coarse-mesh resolution. The ideal and actual convergence of the errors in U_{xx} are plotted in the heavy black line and □ respectively. Fourth-order discretisations are used. Actual errors on L_1 mesh from a 4th order interpolant run are plotted as o and the heavy dashed line is the best linear fit. The convergence (for o) is less than 4th order. △ and ▽ are the results for 6th and 8th order interpolants. The linear fits show a convergence greater than the coarse mesh, i.e., 4th order*

4.2 Zero dimensional ignition

Our next test case is a 0D/homogeneous ignition of a stoichiometric CH_4–Air mixture described by the system:

$$\frac{d\Phi}{dt} = \mathbf{G}(\Phi), \qquad \Phi = \begin{Bmatrix} T \\ Y_1 \\ \vdots \\ Y_{N-1} \\ P_0 \end{Bmatrix},$$

$$\mathbf{G} = \begin{Bmatrix} \frac{1}{\rho c_p}\left[\frac{\gamma-1}{\gamma}\frac{dP_0}{dt} - \sum_{i=1}^{N} h_i\omega_i\right] \\ \frac{1}{\rho}\omega_1 \\ \vdots \\ \frac{1}{\rho}\omega_{N-1} \\ \rho\left(\frac{1}{\overline{W}}\frac{dT}{dt} + T\sum_{i=1}^{N}\frac{1}{W_i}\frac{dY_i}{dt}\right) \end{Bmatrix}, \quad \sum_{i=1}^{N} Y_i = 1 \tag{5}$$

where T is the gas temperature, Y_i is the mass fraction of species i in the mixture, P_0 is the stagnation pressure, N is the total number of species, h_i is the specific enthalpy of species i, ω_i is the chemical production rate of species i, W_i is the molecular weight of species i, $\overline{W}$ is the mixture molecular weight, ρ is the mixture density, c_p is the mixture constant-pressure specific heat and γ is the ratio of specific heats. Equation (5) is identical to equation (2) without the spatial derivative term (**F**). We use a CH_4–Air mechanism (GRI-Mech 1.2, Frenklach et al., 1995). Pressure rises as the mixture ignites, since the entire phenomenon occurs within rigid walls, i.e., volume and mass are constant. These equations are 0D reductions of the low Mach number form of the Navier-Stokes, energy, and species equations; details about the equations and the dependence of h_i, ρ, c_p, γ, ω_i, $\overline{W}$ on Φ can be found in (Majda and Sethian, 1985; Najm, 1996).

For the 0D ignition test case we designed and implemented six components (Figure 9):

- an initial condition component that imposes stoichiometric CH_4–Air mixture composition at 1500 K and 1 atm
- an implicit stiff/non-stiff integrator (Cvode Component) that time-advances the system as it ignites
- an adaptor (problemModeler) that assembles the RHS terms, i.e., the source terms and the pressure
- a component (ThermoChemistry) that embodies the chemical interactions providing the source terms for temperature and species
- a component (ref) which holds the reference values needed by ThermoChemistry and are problem dependent
- a component (dpdt) that computes the pressure term which depends on the boundary conditions of the problem; for the 0D ignition we have constant mass and volume, i.e., rigid adiabatic walls.

The code was tested successfully and results compared with results obtained with Senkin (Kee et al., 1988). After ignition, temperature and pressure reached their maximum values and the chemical system reached equilibrium (Lefantzi and Ray, 2003). This test case demonstrated that the CCA methodology was flexible enough to accommodate significant numerical complexity without loss of performance. Further, this allowed us to debug and optimise chemistry related components (the main source of computational loads) in isolation prior to their incorporation into a large parallel component assembly that solved multi-dimensional PDEs (described in the following section).

4.3 2D ignition

For this test case we expand the 0D ignition test case to include spatial terms to model diffusion. The equations are:

$$\frac{\partial\Phi}{\partial t} = \mathbf{K}\nabla\cdot(\mathbf{B}\nabla\Phi) + \mathbf{R},$$

$$\Phi = \begin{Bmatrix} T \\ Y_1 \\ \vdots \\ Y_{N-1} \end{Bmatrix}, \mathbf{K} = \begin{Bmatrix} \frac{1}{\rho c_p} \\ \frac{1}{\rho} \\ \vdots \\ \frac{1}{\rho} \end{Bmatrix}, \mathbf{B} = \begin{Bmatrix} \lambda \\ \rho \mathrm{D}_1 \\ \vdots \\ \rho D_{N-1} \end{Bmatrix},$$
$$\mathbf{R} = \begin{Bmatrix} \frac{1}{\rho c_p}\left[-\sum_{i=1}^{N} h_i \omega_i\right] \\ \frac{1}{\rho}\omega_1 \\ \vdots \\ \frac{1}{\rho}\omega_{N-1} \end{Bmatrix}, \sum_{i=1}^{N} Y_i = 1 \qquad (6)$$

where **R** is the reactive production of heat and species while $\nabla \cdot (\mathbf{B} \nabla \Phi)$ is the diffusive transport (by Fick's Law) of the heat and the species. λ is the thermal conductivity and D_i are the diffusion coefficients. The chemical production of a species is governed by the chemical reactions it undergoes; heat production (by chemical reactions) is governed by the summation over all the species of the enthalpy change of each species.

This system of PDEs models a reaction–diffusion system. It includes chemistry and the diffusion of heat and species. It does not include momentum, transport of heat by diffusion of species and radiation. Pressure is constant in time and space, i.e., burning in an open domain. A constant Lewis number model (different Le for different species) is used for diffusion. We use the operator-split scheme described in Section 2.2. Chemistry is modeled using 9 species, 19 reversible reactions H_2–Air mechanism (Dryer et al., 1991). The domain in 1 cm × 1 cm.

Figure 10 shows a graphical representation of the component assembly for the Reaction-Diffusion equations. The explicit integrator subsystem includes a Runge-Kutta-Chebyshev integrator (*Integrator*) and a component to calculate the diffusion fluxes (*DiffusionPhysics*). This component makes use of Lewis number based diffusion coefficients. We also see a component (*ErrorEstAndRegrid*) that estimates the error at a cell and flags regions for refinement/coarsening and the *InitialConditions* component which imposes the initial condition. The assembly of components used to solve the 0D problem in Section 4.2 (see Figure 9) forms the 'chemistry' subsystem in the 2D problem. Thus the component architecture accommodates and incorporates multiple physical models into the simulation software in an easy and intuitive manner, a feature critical to the extensibility of any simulation/solution architecture. The code assembly shown in Figure 10 was used to validate second-order discretisations by running a problem, a H_2–Air stoichiometric mixture with random temperature distribution hot enough in certain localised regions to ignite, on a 100 × 100, 200 × 200, 400 × 400 and 800 × 800 uniform mesh on 28 CPUs. Error at a given resolution was defined as its RMS difference from a solution obtained on a mesh twice as fine. It was concluded that a 12.5 μm resolution was required to obtain a 6% difference/error, and was accepted as the resolution of choice (details are in Lefantzi et al., 2003)).

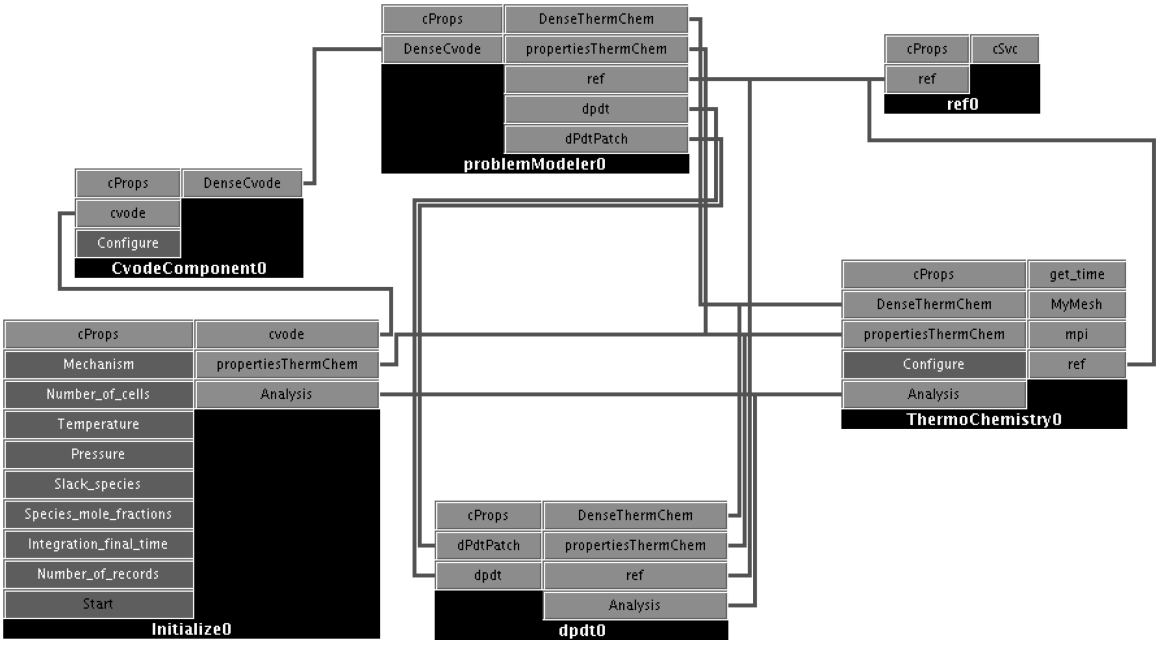

Figure 9 *A picture of the CCA component assembly to solve the 0D ignition problem. We see Initialise0 a component that imposes the initial condition, CvodeComponent0 which solves a set of ODEs, ThermoChemistry0 which models the chemistry and supplies the rate of change of temperature and species concentration due to chemistry given a thermodynamic state vector and dpdt0 which models the rise of pressure with temperature in a confined gas*

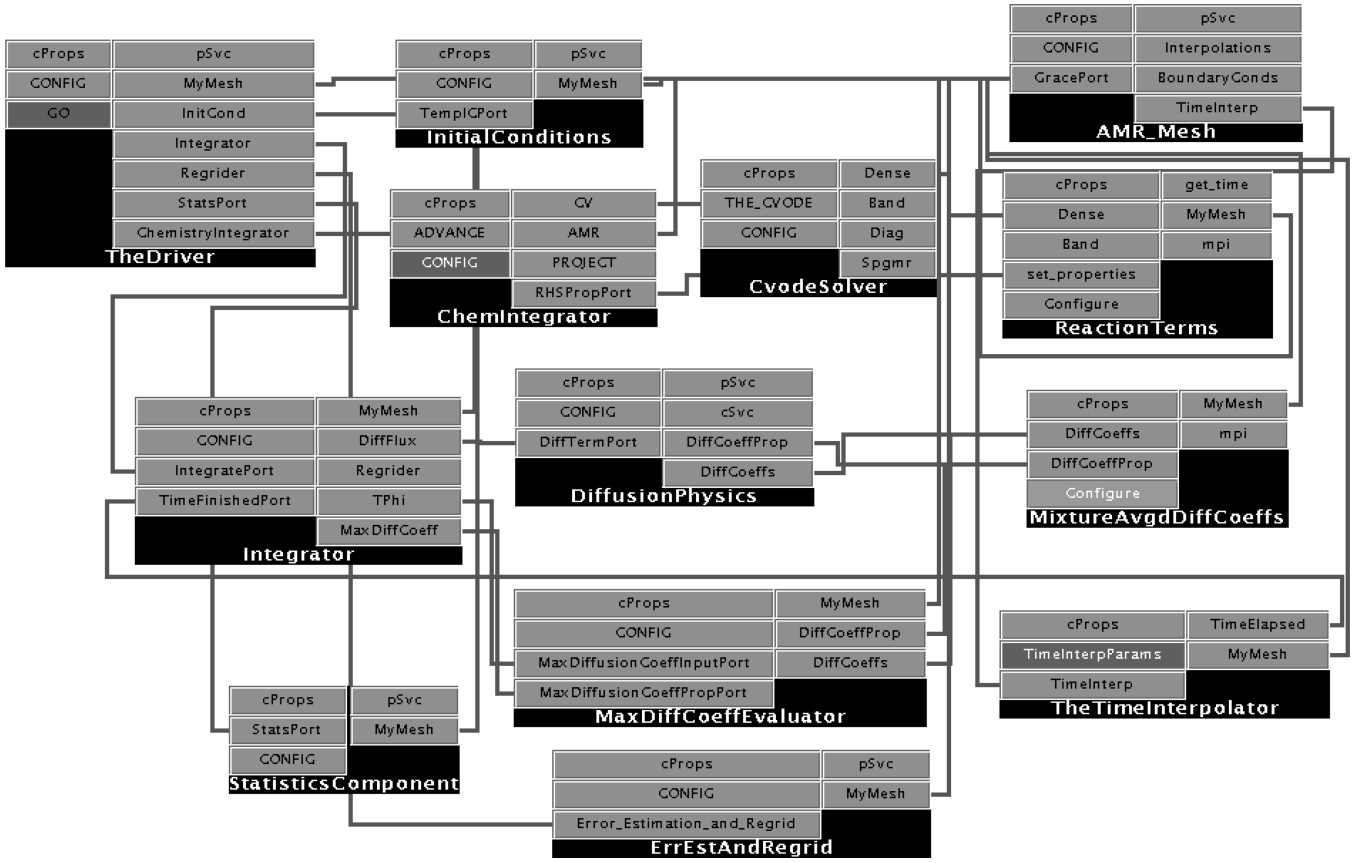

Figure 10 *A picture of the Reaction-Diffusion code linkage in the GUI. The 0D code is seen to be a subsystem, driven by the adaptor ChemIntegrator. The explicit integrator subsystem with a (Runge- Kutta-Chebyshev) Integrator, MixtureAvgdDiffCoeffs, DiffusionPhysics and MaxDiffCoeffEvaluator advances the spatial terms in equation (2) with dynamically adjusted timesteps. ErrEstAndRegrid is invoked periodically to keep the mesh correctly resolved. Sundry other components (StatisticsComponent and InitialConditions) complete the assembly*

We performed this analysis with fourth-order discretisations. The initial condition was a single Gaussian 'hot spot' in a uniformly distributed stoichiometric H_2–Air mixture. In Figure 11 we plot the results. We see that temperature T and OH concentration behave as expected, while HO_2 and H_2O_2 deviate from fourth-order. We define an 'error' $E_{\Delta x}$ at resolution Δx as $E_{\Delta x} = |\Phi_{\Delta x} - \Phi_{\Delta x/2}|$ where $||$ is some norm. Since the exact solution $\Phi_{\text{exact}} = \Phi_{\Delta x} + c(\Delta x)^p$, where p is the order of the discretisation error, one may derive an expression for p from $\phi^{(k)}$, the kth element of Φ, as

$$p(\Delta x) = \log_2 \frac{E^{(k)}_{2\Delta x}}{E^{(k)}_{\Delta x}} = \log_2 \frac{\left|\phi^{(k)}_{2\Delta x} - \phi^{(k)}_{\Delta x}\right|}{\left|\phi^{(k)}_{\Delta x} - \phi^{(k)}_{\Delta x/2}\right|}, \tag{7}$$

In our case, we use the L_2 norm to calculate $E^{(k)}_{\Delta x}$. We tabulate $p(\Delta x)$ for temperature and the concentrations of OH, HO_2 and H_2O_2 in Table 4. Clearly, $p(\Delta x)$ is seen to approach four with increasing resolution.

Table 4 *Order of convergence of the error $p(\Delta x)$ (equation (7)) of various variables as obtained by comparing results from two different resolutions. Variables with large structures like temperature and OH are seen to show nearly fourth-order convergence even at low resolutions, while HO_2 and H_2O_2 approach 4 only at higher resolutions. The errors used here are plotted in Figure 11*

Δx [cm]	T	OH	HO_2	H_2O_2
0.005	3.81	3.76	3.27	3.07
0.0025	3.81	3.91	3.85	3.77

Having established a resolution requirement, i.e., the resolution for the finest mesh to be 800×800, we performed both second and fourth-order simulations on SAMR meshes. The initial condition was a H_2–Air stoichiometric mixture with random temperature distribution. A 100×100 mesh was used as the Level 0 mesh. A refinement factor of 2 was used with 4 mesh levels to achieve an effective resolution of 800, i.e., 12.5 μm. RKC was used with a global timestep (on the Level 0 mesh) limited to 500 ns to reduce splitting errors. For regriding, we define a 'resolution metric' ε_{ij} such that

$$\varepsilon = \max_k \left(\left| \frac{\delta_x \phi_{ij}^{(k)}}{\phi_{\max}^{(k)}} \right|, \left| \frac{\delta_y \phi_{ij}^{(k)}}{\phi_{\max}^{(k)}} \right| \right), \tag{8}$$

where k is the index over all variables at a given cell (i, j) and $\phi^{(k)}$ are the elements of Φ. $\varepsilon_{ij} > 0.05$ was used as the condition for refinement. Regions with $\varepsilon_{ij} < 0.0125$ were coarsened. Second-order discretisations were paired with bicubic interpolations and fourth-order discretisations with sixth-order interpolants. In Figure 12 we see that some regions in the mixture ignite. The ignition fronts are seen to move through the mixture and annihilate sections of each other as they interact. The distribution of patches over different levels of refinement at two different time instants is also seen in Figure 12. As certain regions (initially with large temperature gradients) fail to ignite and lose their gradients via diffusion, the fine patches are removed, leaving the region with coarser meshes. In Figure 13 we see H_2O_2 profiles plotted on the finest mesh. We see that the resolution capability of the SAMR resolves the fine H_2O_2 profile with about ten points.

In Figure 14 we plot the HO_2 and H_2O_2 profiles extracted from the second and fourth-order runs. We see that they are very similar. The 20 grid points in the profiles on the finest mesh indicate that the solution is adequately resolved. The fourth-order run required that an 8th order filter be applied at the beginning of every timestep, a consequence of using the sixth-order prolongation operator. Filtering was not required in a separate run (not shown here) where the sixth-order interpolant was replaced by a fourth-order one. Filters were not required with sixth-order discretisations in Section 4.1, probably because the front was quite well resolved even on the coarse mesh. In this particular case, where the fronts *develop* in time, the sixth-order interpolants were observed to give unphysical answers when filtering is not applied, during the formative stages of the fronts (when steep, probably unresolved, gradients, would exist, especially in the coarser levels).

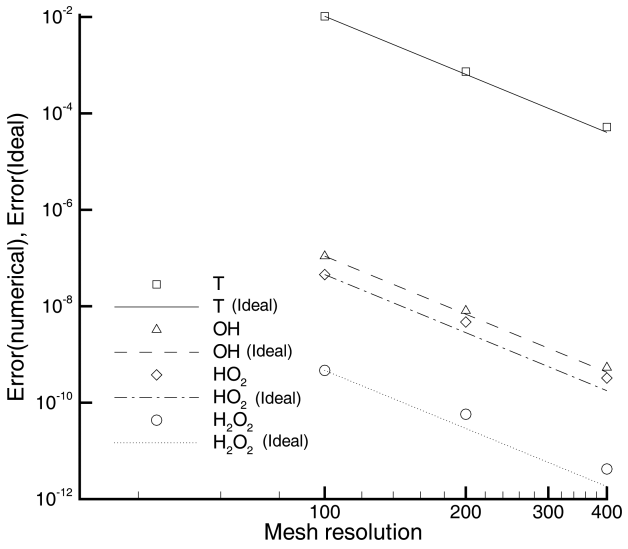

Figure 11 *Convergence of temperature and the concentrations of OH, HO_2 and H_2O_2 of runs done on 100×100, 200×200, 400×400 and 800×800 uniform meshes. The 'error' of a solution at a given resolution is the RMS difference from a solution obtained on a mesh twice as fine. We see that temperature (T) and OH concentration behave as expected, while HO_2 and H_2O_2 deviate from fourth-order. Lines represent the ideal behaviour, while symbols correspond to the data obtained with our simulations. In Table 4 we show the actual convergence rate, which is seen to tend to four with increasing resolution*

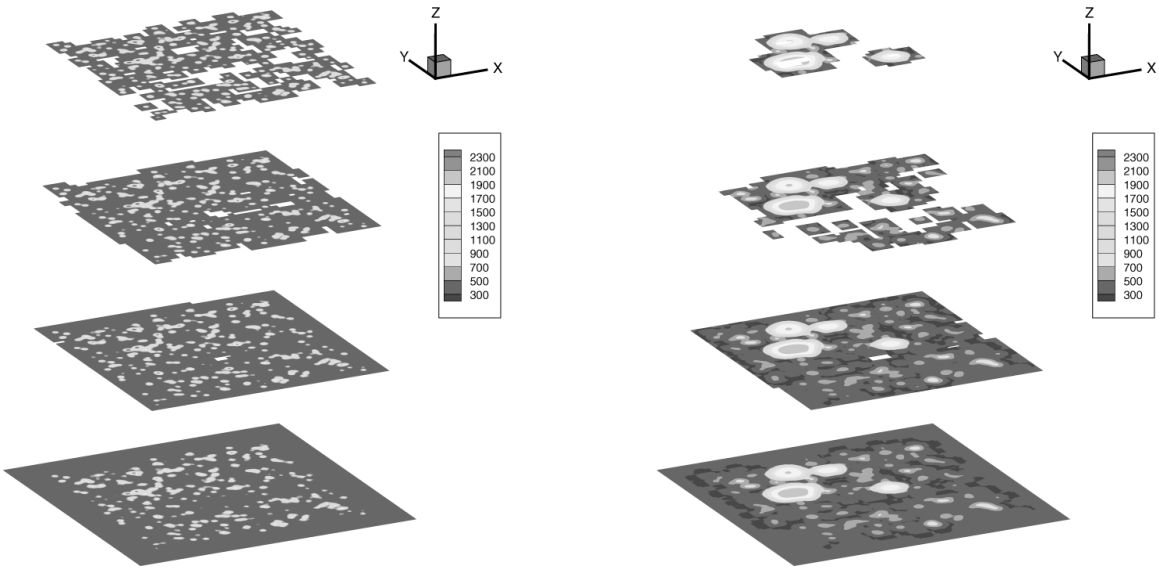

Figure 12 *Temperature field at time t = 0 (left) and 90 microseconds (right). Patches at different levels in the grid hierarchy are plotted separately. We see the vastly different hierarchy configuration at the two instants in time. Regions which have ignited and have steep reaction fronts are resolved with Level 3 grids. Regions which failed to ignite have had the fine meshes removed by t = 90 microseconds*

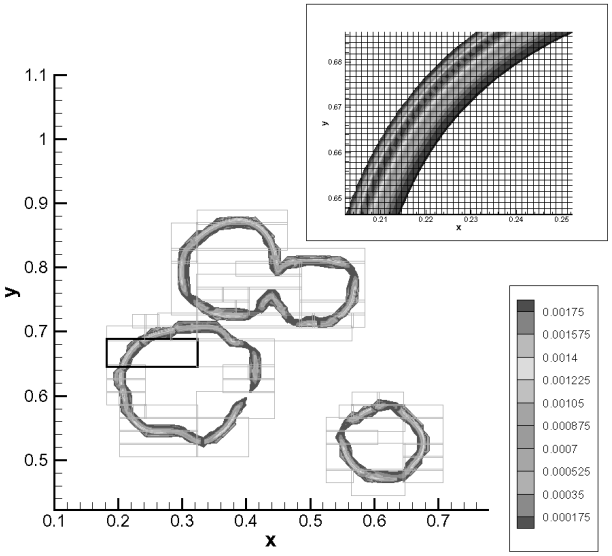

Figure 13 *H_2O_2 profiles plotted only on the finest mesh level. Patches on Levels 3 are shown. Inset: A zoom of the patch in black is shown overlayed by the mesh. We see the H_2O_2 profile being resolved by about ten grid points*

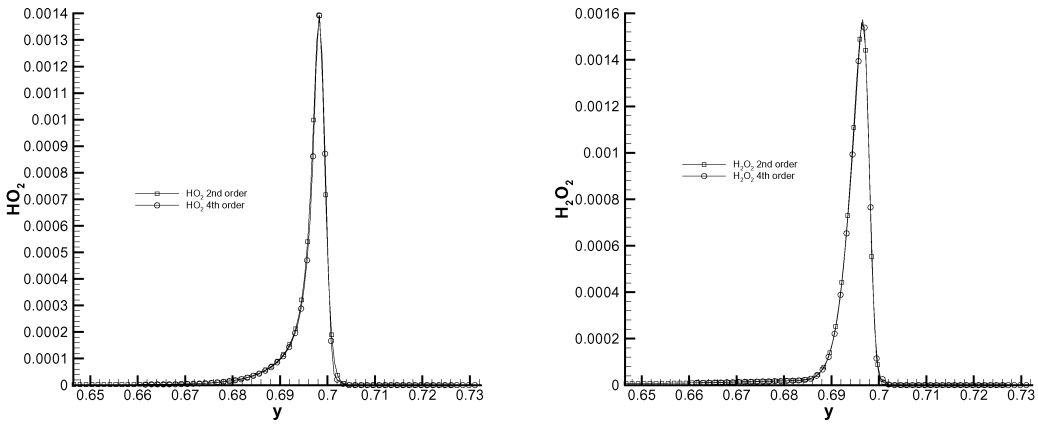

Figure 14 *HO_2 (left) and H_2O_2 (right) profiles on the 12.5 micron mesh level using the second- and fourth-order approaches, at x = 0.26 cm. The results are for t = 93 microseconds. We can see that the profiles are very similar. There are about 20 points in the profile on the finest mesh. ○ denote fourth-order results while □ denote second order*

5 CONCLUSIONS AND FUTURE DIRECTIONS

We have demonstrated the use of high-order spatial discretisations in a parallel multilevel block-SAMR environment to solve a set of coupled PDEs. Operator-splitting was used to incorporate an implicit integrator to integrate stiff chemical ODE systems while an extended stability RKC integrator was used, with subcycling, to advance the non-stiff component of the PDEs. Convergence analysis was used to determine if the requisite high-order accuracy was obtained in the solution. Timestep size was limited by splitting errors rather that the stability constraint of the extended stability RKC method. A stiff nonlinear problem was solved with second and fourth-order spatial discretisations and SAMR, and the results were compared.

In answer to the questions posed in Section 4, we find

- Fourth-order methods can be incorporated in a SAMR setting and used in realistic problems. In Section 4.1 we showed how a fourth-order discretisation needs to be coupled to a spatial interpolant of an appropriate order to obtain the required accuracy. Further, as shown in Figure 7, fourth-order methods can be more economical than second order approaches.
- The advanced discretisation and meshing techniques described in this paper were implemented within the CCA (component) software framework. Thus, a component based software paradigm is rich enough to support high-performance scientific simulation codes. It can also be used to incorporate legacy software. The software decomposition strategy is basically functional and is influenced by the nature of the numerical scheme.

- The modularity and plug-and-play characteristics of the Ccaffeine framework were exploited to assemble complicated solution systems in a hierarchal manner (2D simulation code contains a subset 15 that was used to solve the 0D ignition) as well as try out various numerical schemes (second- and fourth-order discretisations) with minimal changes. Its high performance nature and ability to accommodate parallel computing were used to solve our problems on multiple processors.

In the future, the current CCA toolkit will be augmented to solve DAEs rather than ODEs. Further, we will address the issue of flux conservation at coarse-fine patch boundaries to enable the construction of a conservative high-order scheme. Since the ultimate aim is to solve the low Mach approximation of the Navier-Stokes equation, it will involve a projection scheme, where the solution is projected to a manifold so that it satisfies an algebraic constraint. Mathematically it involves solving a Poisson equation on a SAMR mesh. Investigations are underway to determine how this may be achieved efficiently and in a consistent manner, given the dearth of literature on high-order approaches on block SAMR meshes.

ACKNOWLEDGEMENTS

The US Department of Energy (DOE), Office of Basic Energy Sciences, Division of Chemical Sciences, Geosciences, and Biosciences, and SciDAC Computational Chemistry Program supported this work. The DOE, Office of Advanced Scientific Computing Research, Division of Mathematical, Information and Computational Sciences also supported this work. Sandia National Laboratories is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94-AL85000.

REFERENCES

Allan, B.A., Armstrong, R.C., Wolfe, A.P., Ray, J., Bernholdt, D.E. and Kohl, J.A. (2002) 'The CCA core specifications in a distributed memory SPMD framework', *Concurrency: Practice and Experience*, Vol. 14, pp.323–345, also at http://www.cca-forum.org/ccafe/concurrency-paper/.

Almgren, A.S., Bell, J.B., Colella, P., Howell, L.H. and Welcome, M.L. (1998) 'A conservative adaptive projection method for the variable density incompressible Navier-Stokes equation', *J. Comput. Phys.*, Vol. 142, pp.1–46.

Armstrong, R., Gannon, D., Geist, A., Keahy, K., Kohn, S., McInnes, L., Parker, S. and Smolenski, B. (1999) 'Towards a common component architecture for high performance scientific computing', *Proceedings of the 8th International Symposium on High Performance Distributed Computing*, August, Redondo Beach, California.

Bell, J.B., Day, M.S., Almgren, A.S., Cheng, R.K. and Shepherd, I.G. (2003a) 'Numerical simulation of premixed turbulent methane combustion', *Proceedings of the Second MIT Conference on Computational Fluid and Solid Mechanics*, June, Elsevier Science, Boston, Mass.

Bell, J.B., Day, M.S., Shepherd, I.G., Johnson, M., Cheng, R.K., Beckner, V.E., Lijewski, M.J. and Grcar, J.F. (2003b) 'Numerical simulation of a laboratory-scale turbulent V-flame', *Technical Report LBNL-54198*, Lawrence Berkeley National Laboratory, Berkeley, CA.

Berger, M. and Oliger, J. (1984) 'Adaptive mesh refinement for hyperbolic partial differential equations', *Journal of Computational Phys.*, Vol. 53, pp.484–512.

Berger, M.J. and Colella, P. (1989) 'Local adaptive mesh refinement for shock hydrodynamics', *Journal of Computational Physics*, Vol. 82, pp.64–84.

Browne, S., Dongarra, J., Garner, N., Ho, G. and Mucci, P. (2000) 'A portable programming interface for performance evaluation on modern processors', *International Journal of High Performance Computing Application*, Vol. 14, No. 3, pp.189–204, //http://icl.cs.utk.edu/projects/papi/.

Chung, T.J. (2002) *Computational Fluid Dynamics*, Cambridge University Press.

Cockburn, B. and Shu, C-W. (1998) 'The local discontinuous Galerkin method for time-dependent convection-diffusion systems', *SIAM J. Num. Anal.*, Vol. 35, No. 6, pp.2440–2463.

Cohen, S.D. and Hindmarsh, A.C. (1994) 'CVODE user guide', *Technical Report UCRL-MA-118618*, Lawrence Livermore National Laboratory, Livermore, CA.

CORBA Component model webpage http://www.omg.com, accessed July 2002.

Day, M.S. and Bell, J.B. (2000) 'Numerical simulation of laminar reacting flows with complex chemistry', *Combust. Theory Modelling*, Vol. 4, pp.535–556.

de St. Germain, J.D., McCorquodale, J., Parker, S.G. and Johnson, C.R. (2000) 'UINTAH: a massively parallel problem solving environment', *HPDC '00: Ninth IEEE International Symposium on High Performance and Distributed Computing*, August.

Dryer, F.L., Yetter, R.A. and Rabitz, H. (1991) 'A comprehensive reaction mechanism for carbon monoxide/hydrogen/oxygen kinetics', *Combust. Sci. and Tech.*, Vol. 79, pp.97–128.

Englander, R. and Loukides, M. (1997) *Developing Java Beans (Java Series)*, O'Reilly and Associates, http://www.java.sun.com/products/javabeans.

Fall, C.P., Marland, E.S., Wagner, J.M. and Tyson, J.J. (Eds). (2001) *Computational Cell Biology*, *Interdisciplinary Applied Mathematics, Mathematical Biology*, Vol. 20, Springer.

Frenklach, M., Wang, H., Goldenberg, M., Smith, G.P., Golden, D.M., Bowman, C.T., Hanson, R.K., Gardiner, W.C. and Lissianski, V. (1995) 'GRI-Mech – an optimized detailed chemical reaction mechanism for methane combustion', *Technical Report GRI-95/0058*, Gas Research Institute, Gas Research Institute, 8600 West Bryn Mawr Avenue, Chicago, Illinois 60631-3562, November 1, http://diesel.me.berkeley.edu/_grimech/new21/version12/text12.html.

GrACE homepage http://www.caip.rutgers.edu/TASSL/

Hirschfelder, J.O., Curtiss, C.F. and Bird, R.B. (1954) *Molecular Theory of Gases and Liquids*, Wiley, New York.

Jameson, L. (2000) 'AMRvs. high-order schemes, wavelets as a guide', *Technical Report UCRL-ID-141537*, Lawrence Livermore National Laboratory, Livermore, California.

Karniadakis, G.E. and Sherwin, S.J. (1999) *Spectral/Hp Element Methods for CFD. Numerical Mathematics and Scientific Computation*, Oxford University Press.

Kee, R.J., Lutz, A.E. and Miller, J.A. (1988) 'SENKIN: a fortran program for predicting homogeneous gas phase chemical kinetics with sensitivity analysis', *Technical Report 87-8248*, Sandia National Laboratories, PO Box 969, Livermore, CA-94551.

Kennedy, C.A. and Carpenter, M.H. (1994) 'Several new numerical methods for compressible shear layer simulations', *Appl. Num. Math.*, Vol. 14, pp.397–433.

Knio, O.M., Najm, H.N. and Wyckoff, P.S. (1999) 'A semi-implicit numerical scheme for reacting flow. II. stiff, operator-split formulation', *J. Comp. Phys.*, Vol. 154, pp.428–467.

Kozlov, R., Kværnø, A. and Owren, B. (2004) 'The behaviour of the local error in splitting methods applied to stiff problems', *J. Comp. Phys.*, Vol. 195, No. 2, pp.576–593.

Lanser, D. and Verwer, J.G. (1999) 'Analysis of operator splitting for advection-diffusion-reaction problems from air pollution modelling', *J. Comp. Appl. Math.*, Vol. 111, No. 1, pp.201–216.

Lefantzi, S. and Ray, J. (2003) 'A component-based scientific toolkit for reacting flows', *Proceedings of the Second MIT Conference on Computational Fluid and Solid Mechanics*, Elsevier Science, Boston, Mass.

Lefantzi, S., Kennedy, C., Ray, J. and Najm, H.N. (2003) 'A study of the effect of higher order spatial discretizations in SAMR (structured adaptive mesh refinement) simulations', *Proceedings of the Fall Meeting of the Western States Section of the Combustion Institute*, Los Angeles, California, October, distributed on a CD.

Lefantzi, S., Ray, J. and Najm, H.N. (2003) 'Using the common component architecture to design high performance scientific simulation codes', *Proceedings of the International Parallel and Distributed Processing Symposium*, April, Nice, France.

Lele, S.K. (1992) 'Compact finite difference schemes with spectral-like resolution', *J. Comp. Phys.*, Vol. 103, pp.16–42.

LeVeque, R.J. and Yee, H.C. (1990) 'A study of numerical methods for hyperbolic conservation laws with stiff source terms', *J. Comp. Phys.*, Vol. 86, pp.187–210.

Lilek, Z. and Perić, M. (1995) 'A fourth-order finite volume method with collocated variable arrangement', *Computers & Fluids*, Vol. 24, No. 3, pp.239–252.

Majda, A. and Sethian, J. (1985) 'The derivation and numerical solution of the equations for zero Mach number combustion', *Comb. Sci. and Technology*, Vol. 42, pp.185–205.

Minion, M.L. (1996) 'A projection method for locally refined grids', *J. Comp. Phys.*, Vol. 127, No. 1, pp.158–178.

Najm, H.N. (1996) 'A conservative low Mach number projection method for reacting flow modeling', in Chan, S.H. (Ed.): *Transport Phenomena in Combustion*, Taylor and Francis, Wash. DC, Vol. 2, pp.921–932.

Najm, H.N., Knio, O.M. and Paul, P.H. (2003) 'A numerical scheme for modeling low Mach number reacting flow with detailed chemistry and transport', Technical Report, Sandia National Laboratories, Albuquerque, NM, USA, SAND2003-8412.

Najm, H.N., Knio, O.M., Paul, P.H. and Wyckoff, P.S. (1998) 'A study of flame observables in premixed methane-air flames', *Comb. Sci. Tech.*, Vol. 140, Nos. 1–6, pp.369–403.

Parashar, M. and Browne, J.C. (2000) 'System engineering for high performance computing software: the HDDA/DAGH infrastructure for implementation of parallel structured adaptive mesh refinement', in Gannon, D.B., Baden, S.B., Chrisochoides, M.P. and Norman, M.L. (Eds.): *Structured Adaptive Mesh Refinement*, *IMA*, Vol. 117, Springer-Verlag.

Pember, R.B., Howell, L.H., Bell, J.B., Colella, P., Crutchfield, W.Y., Fiveland, W.A. and Jesse, J.P. (1998) 'An adaptive projection method for unsteady low-Mach number combustion', *Combust. Sci. and Tech.*, Vol. 140, pp.123–168.

Peters, N. and Rogg, B. (1993) *Reduced Kinetic Mechanisms for Applications in Combustion Systems*, Springer-Verlag.

Pope, S.B. (2000) *Turbulent Flows*, Cambridge University Press.

Ropp, D.L., Shadid, J.N. and Ober, C.C. (2004) 'Studies of the accuracy of time integration methods for reaction-diffusion equations', *J. Comp. Phys.*, Vol. 194, No. 2, pp.544–574.

SAMR homepage http://www.caip.rutgers.edu/~jaray/CCA/documentation.html, last accessed October 4, 2002.

Shampine, L.F., Sommeijer, B.P. and Verwer, J.G. (1998) 'RKC: an explicit solver for parabolic PDEs', *J. Comp. Appl. Math.*, Vol. 88, pp.315–326.

Sheng, Q. (1989) 'Solving linear partial differential equations by exponential splitting', *IMA J. Numer. Anal.*, Vol. 9, pp.199–212.

Spee, E.J. (1995) 'Coupling advection and chemical kinetics in a global atmospheric test model', *Report NM-R9508*, CWI, Amsterdam.

Spee, E.J., Verwer, J.G., de Zeeuw, P.M., Blom, J.G. and Hundsdorfer, W. (1998) 'A numerical study for global atmospheric transport-chemistry problems', *Mathematics and Computers in Simulation*, Vol. 48, pp.177–204.

Sportisse, B. (2000) 'An analysis of operator splitting techniques in the stiff case', *J. Comp. Phys.*, Vol. 161, pp.140–168.

Strang, G. (1968) 'On the construction and comparison of difference schemes', *SIAM J. Numer. Anal.*, Vol. 5, No. 3, pp.506–517.

Sullivan, N., Jensen, A., Glarborg, P., Day, M.S., Grcar, J.F. and Bell, J.B. (2002) 'Ammonia conversion and NOx formation in laminar coflowing nonpremixed methane-air flames', *Combust. Flame*, Vol. 131, No. 3, pp.285–298.

Tannehill, J.C. Anderson, D.A. and Pletcher, R.C. (1997) *Computational Fluid Mechanics and Heat Transfer*, 2nd ed., Taylor and Francis.

Trefethen, L.N. and Weideman, J.A.C. (1991) 'Two results on polynomial interpolation in equally spaced points', *J. Approx. Theory*, Vol. 65, pp.247–260.

Verwer, J.G. and Sommeijer, B.P. (2004) 'An implicit-explicit Runge-Kutta Chebyshev scheme for diffusion-reaction equation', *SIAM J. Sci. Comp.*, Vol. 25, No. 5, pp.1824–1835.

Verwer, J.G., Blom, J.G., van Loon, M. and Spee, E.J. (1995) 'A comparison of stiff ODE solvers for atmospheric chemistry problems', *Atmos. Eviron.*, Vol. 30, pp.49–58.

Verwer, J.G., Hunsdorder, W.H. and Sommeijer, B.P. (1990) 'Convergence properties of the Runge-Kutta-Chebyshev method', *Num. Math.*, Vol. 57, pp.157–178.

Verwer, J.G., Spee, E.J., Blom, J.G. and Hundsdorfer, W.H. (1999) 'A second order rosenbrock method applied to photochemical dispersion problems', *SIAM J. Sci. Comput.*, Vol. 20, pp.1456–1480.

Visbal, M.R. and Gaitonde, D.V. (2002) 'On the use of higher-order finite-difference schemes on curvilinear and deforming meshes', *J. Comp. Phys.*, Vol. 181, pp.155–185.

Visual Basic webpage http://msdn.microsoft.com/vbasic, accessed July 2002.

Wang, Z. and Huang, G.P. (2002) 'An essentially nonoscillatory high-order Padè-type (ENOPadè) scheme, *J. Comp. Phys.*, Vol. 177, pp.37–58.

Wilcox, D.C. (1993) *Turbulence Modeling for CFD*, Ist ed., DCW Industries, Inc.

Williams, F.A. (1985) *Combustion Theory*, 2nd ed., Addison-Wesley, New York.

Wright, J.P. (1998) 'Numerical instability due to varying time steps in explicit wave propagation and mechanics calculations', *J. Comp. Phys.*, Vol. 140, pp.421–431.

XCAT homepage http://www.extreme.indiana.edu/ccat/, also http://www.extreme.indiana.edu/xcat/; accessed July 2002.

QUERY

Author please provide issue number for the highlighted references