PRISM: Piecewise Reusable Implementation of Solution Mapping. An Economical Strategy for Chemical Kinetics.

Shaheen R. Tonse,[a1] Nigel W. Moriarty,[b] Nancy J. Brown[a] and Michael Frenklach[a,b]
[a] Environmental Energy Technologies Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA.
[b] Department of Mechanical Engineering, University of California at Berkeley, Berkeley, CA 94720–1740, USA.

(Received 7 December 1998)

Abstract. In a chemical kinetics calculation a solution-mapping procedure is applied to parametrize the solution of the initial-value ordinary differential equation system as a set of algebraic polynomial equations. To increase the accuracy, the parametrization is done piecewise, dividing the multi-dimensional chemical composition space into hypercubes and constructing polynomials for each hypercube. A differential equation solver is used to provide the solution at selected points throughout a hypercube, and from these solutions the polynomial coefficients are determined. Factorial design methods are used to reduce the required number of computed points. The polynomial coefficients for each hypercube are stored in a data structure for subsequent re-use, since over the duration of a flame simulation it is likely that a particular set of concentrations and temperature will occur repeatedly at different times and positions.

The method is applied to H_2-air combustion using an 8-species reaction set. After N_2 is added as an inert species and enthalpy is considered, this results in a 10-dimensional chemical composition space. To add the capability of using a variable time-step, time-step is added as an additional dimension, making an 11-dimensional space. Reactive fluid dynamical simulations of a 1-D laminar premixed flame and a 2-D turbulent non-premixed jet are performed. The results are compared to identical control runs which use an ordinary differential equation solver to calculate the chemical kinetic rate equations. The resulting accuracy is very good, and a factor of 10 increase in computational efficiency is attained.

1 Introduction

The study of combustion through finite-difference numerical simulations typically involves a marriage of computational fluid dynamics (CFD) and computational chemical kinetics. The role of the CFD includes determination of velocities, pressure, convection across cell boundaries, the equation of state, the modelling of turbulence, and the diffusion of chemical species across cell boundaries. The responsibility of the chemistry portion is to determine the changes in concentration of each chemical species and enthalpy in response to the chemical source terms. This is usually accomplished through the solution of a system of coupled ordinary differential equations (ODE's), one for each of the chemical species and one for the enthalpy. It often proves to be computationally expensive depending on the size of the reaction set and the method of solution. Some numerical codes simultaneously solve the diffusion, convection and chemical source terms for the species rate equations, while others solve them sequentially, in a procedure called operator-splitting. The approach that we formulate in this paper is suited for an operator-split calculation.

Although it would appear that the CFD bears the brunt of the computational effort, it is in fact the chemistry which demands the most CPU time. For example, a 2-dimensional calculation with the CFD portion performing all of the above-mentioned tasks, and with the chemistry of a H_2-air mixture, using a reaction set of 9 chemical species and 29 reactions, 85 to 90% of the CPU time is spent on the chemistry.[1] Yet this hydrogen reaction set is a comparatively simple one. Considering that

[1]Author to whom correspondence should be addressed. Email: tonse@lbl.gov

reaction sets with 500-800 chemical species are envisioned in future studies of diesel combustion the fraction of CPU time spent on chemistry will then be a matter of determining the number of "9"s following the decimal point in 99.9%.

A brute-force calculation with a large number of species and reactions and wide ranges in time-scales is not practical on today's computers. In order to reduce the severity of a brute-force approach to chemical kinetics problems varied approaches are used, often in combination with one another, for example: 1) reduction of the reaction set, both in the number of chemical species and the number of reactions, in a systematic way after examining sensitivities and reaction fluxes;[2–6] 2) steady-state and partial equilibrium approximations;[7,8] 3) intrinsic low dimensional manifolds;[9] 4) computational singular perturbation;[10] and 5) principal component analysis.[11] These methods reduce the severity of the computational problem, but still require the solution of differential equations.

The basic function of a chemical ODE solver is to take a set of N_s input chemical species concentrations, C_i^t, and temperature, T^t, at the beginning of a time-step, evolve them over a time interval Δt, and return a set of output species concentrations, $C_i^{t+\Delta t}$, and temperature, $T^{t+\Delta t}$. In essence it has mapped an input point, $\mathbf{r}^t(C_1^t, C_2^t, \ldots, C_{N_\mathrm{s}}^t, T^t)$, in $N_\mathrm{s}+1$ dimensional space onto an output point, $\mathbf{r}^{t+\Delta t}(C_1^{t+\Delta t}, C_2^{t+\Delta t}, \ldots, C_{N_\mathrm{s}}^{t+\Delta t}, T^{t+\Delta t})$, giving the answer which corresponds to the evolution of the system after time Δt. Our goal is to parametrize the numerical solutions of the ODE and replace them with a set of algebraic expressions. The simpler algebraic expressions have a limited domain of validity. Covering all of chemical composition space requires multiple expressions, each valid over a different portion of composition space, calculated and stored in a data structure.

Methods which retrieve and re-use existing results can be used either independently or in combination with some of the previous methods. A parametrization method by itself does nothing to reduce the number of species and reactions; however because of calculational speedup it can make other methods unnecessary. Since most of the other methods require some effort and intuition, it is preferable to use the parametrization method. Past and current approaches that utilize storage and re-use of chemical calculations include: 1) solution mapping[5,12] – parametrization of kinetic model output as a polynomial of input variables developed with the computer runs arranged in factorial designs; 2) fifth to eighth order polynomial algebraic representation of the ODE solutions;[13] 3) laminar flamelet libraries;[14] and 4) dynamically increasing table, with calculations performed only for those parts of chemical composition space actually visited and using linear extrapolation about the point neighborhood to obtain solutions over a range.[15,16] In an earlier application of the solution mapping method, the dependence of output species concentrations was developed as a second-order polynomial of input species concentrations using a single response surface.[12] The use of a single response surface was possible in that case because of rather limited ranges of species concentrations and constant reaction temperature. Attempts to apply this technique to a laminar premixed flame encountered difficulties.[17]

In the present work we implement a combination of solution mapping and data structure organization of response surfaces as a computer code we call PRISM: Piecewise Reusable Implementation of Solution Mapping. Second-order polynomial expressions are used, with relatively large regions of validity, 0.25–0.5 of an order of magnitude along each species concentration axis. Entries are constructed only for those parts of chemical composition space actually accessed by the reaction trajectory. The latter part of the present approach has some similarity to the approach of Pope and co-workers.[15] Unique features of our approach are that the individual data entries are not point-slope data but a set of polynomials covering a *region* of chemical composition space, and the use of different basic data structures. In addition, whereas existing methods are restricted to fixed time-step Δt, in PRISM Δt is declared as an extra dimension in the input space (thus now $\mathrm{N_s}+2$ dimensions), included in the polynomial expression, and treated as a variable. This is important because most chemistry packages are subroutines of CFD programs in which the time-step is determined by stability criteria and hence varies from cycle to cycle.

In Section 2 we discuss in more detail the method used to construct the polynomials, the retrieval and evaluation methods, as well as the data structure used. We also discuss performance from the standpoint of accuracy and economy, the systematics of varying the operational parameters, and the result of applying *a posteriori* enthalpy and elemental mass conservation to improve the solution. In Section 3 we describe performance of the PRISM method for three reactive flow applications, a burn of premixed H_2-air in zero spatial dimensions, a premixed H_2-air 1-dimensional laminar flame, and a non-premixed 2-dimensional turbulent jet with coaxial H_2 and air flows. The H_2-air mechanism details are shown in Table 1. The concern is for propagation of small errors with time, which are quantified by a comparison between the final solutions obtained from an ODE solver and from PRISM. Finally we provide discussion of the results and computational speedup.

Table 1: Hydrogen mechanism containing the reactions and Arrhenius parameters.

reaction			forward rate coefficient[a] A	n	E
2O+M	$\rightleftharpoons$	O_2+M	1.200×10^{17}	-1.000	
O+H+M	$\rightleftharpoons$	OH+M	5.000×10^{17}	-1.000	
O+H_2	$\rightleftharpoons$	H+OH	5.000×10^{4}	2.670	6290.00
O+HO_2	$\rightleftharpoons$	OH+O_2	2.000×10^{13}		
O+H_2O_2	$\rightleftharpoons$	OH+HO_2	9.630×10^{6}	2.000	4000.00
H+O_2+M	$\rightleftharpoons$	HO_2+M	2.800×10^{18}	-.860	
H+2O_2	$\rightleftharpoons$	HO_2+O_2	3.000×10^{20}	-1.720	
H+O_2+H_2O	$\rightleftharpoons$	HO_2+H_2O	9.380×10^{18}	-.760	
H+O_2	$\rightleftharpoons$	O+OH	8.300×10^{13}		14413.00
2H+M	$\rightleftharpoons$	H_2+M	1.000×10^{18}	-1.000	
2H+H_2	$\rightleftharpoons$	2H_2	9.000×10^{16}	-.600	
2H+H_2O	$\rightleftharpoons$	H_2+H_2O	6.000×10^{19}	-1.250	
H+OH+M	$\rightleftharpoons$	H_2O+M	2.200×10^{22}	-2.000	
H+HO_2	$\rightleftharpoons$	O+H_2O	3.970×10^{12}		671.00
H+HO_2	$\rightleftharpoons$	O_2+H_2	2.800×10^{13}		1068.00
H+HO_2	$\rightleftharpoons$	2OH	1.340×10^{14}		635.00
H+H_2O_2	$\rightleftharpoons$	HO_2+H_2	1.210×10^{7}	2.000	5200.00
H+H_2O_2	$\rightleftharpoons$	OH+H_2O	1.000×10^{13}		3600.00
OH+H_2	$\rightleftharpoons$	H+H_2O	2.160×10^{8}	1.510	3430.00
2OH(+M)	$\rightleftharpoons$	H_2O_2(+M)	7.400×10^{13}	-.370	
2OH	$\rightleftharpoons$	O+H_2O	3.570×10^{4}	2.400	-2110.00
OH+HO_2	$\rightleftharpoons$	O_2+H_2O	2.900×10^{13}		-500.00
$\left\{\begin{array}{l}\text{OH} + \text{H}_2\text{O}_2\\ \text{OH} + \text{H}_2\text{O}_2\end{array}\right.$	$\rightleftharpoons$	$\begin{array}{l}\text{HO}_2 + \text{H}_2\text{O}\\ \text{HO}_2 + \text{H}_2\text{O}\end{array}$	$\begin{array}{l}1.750 \times 10^{12}\\ 5.800 \times 10^{14}\end{array}$		$\begin{array}{r}320.00\\ 9560.00\end{array}$
$\left\{\begin{array}{l}2\text{HO}_2\\ 2\text{HO}_2\end{array}\right.$	$\rightleftharpoons$	$\begin{array}{l}\text{O}_2 + \text{H}_2\text{O}_2\\ \text{O}_2 + \text{H}_2\text{O}_2\end{array}$	$\begin{array}{l}1.300 \times 10^{11}\\ 4.200 \times 10^{14}\end{array}$		$\begin{array}{r}-1630.00\\ 12000.00\end{array}$

[a] The forward rate coeffiecents $k = AT^n e^{-E/\mathrm{RT}}$; R is the universal gas constant, T is the temperature in K, the units of E are cal/mol.

2 Method

We partition chemical composition space into non-overlapping block-shaped volumes (*hypercubes*), each adjacent to another, with edges and corners permitted only at regular intervals along the axes. This allows for a simple indexing of each hypercube, which permits fast and efficient searching. Although hypercube placement is determined at the beginning of a simulation, polynomial expression calculations are performed for a hypercube only when the reaction trajectory enters it for the first time. The first hypercube for which polynomial expressions are constructed is that containing the initial point t=t_0:

$$\mathbf{r}^{t_0} \equiv [C_1^{t_0}, C_2^{t_0}, \ldots, C_{N_\mathrm{s}}^{t_0}, T^{t_0}]$$

where $C_1^{t_0}$ is the concentration of species 1 at time t_0. Often for a few time-steps (t_1=t_0+Δt_0, t_2=t_1+Δt_1, t_3=t_2+Δt_2, ...) the polynomials of this hypercube will map successive points, $\mathbf{r}^{t_1}$=$\phi(\mathbf{r}^{t_0},\Delta t_0)$, $\mathbf{r}^{t_2}$=$\phi(\mathbf{r}^{t_1},\Delta t_1)$, $\mathbf{r}^{t_3}$=$\phi(\mathbf{r}^{t_2},\Delta t_2)$,... all the while remaining in the same hypercube (see Figure 1 for a two dimensional example), but at some time $t = t_n$, the solution $\mathbf{r}^{t_{n+1}} = \phi(\mathbf{r}^{t_n}, \Delta t_n)$ will fall outside the hypercube. When this occurs we determine the location of the hypercube where $\mathbf{r}^{t_{n+1}}$ lies and construct polynomials for it. Once constructed, hypercube and polynomial information is stored in a data structure complex. The stored data is reused by searching for an existing entry before attempting to construct new polynomials.

2.1 Polynomial Construction

The N_s+2 dimensional space is divided into hypercubes of a predetermined size, with one axis assigned to each species, one to temperature, and one to the time-step Δt. The concentrations and the time-step are transformed into their logarithms and temperature into its reciprocal, reflecting the Arrhenius form of the reaction rate equation. This transformation produces a better quadratic parametrization; for instance, use of reciprocal temperature results in error reduction of approximately 30%.

Determination of the polynomial expression begins by locating the hypercube that contains a given input point. Tests showed that the hypercube sides can be relatively large, 0.25 to 0.5 of an order of magnitude in concentra-

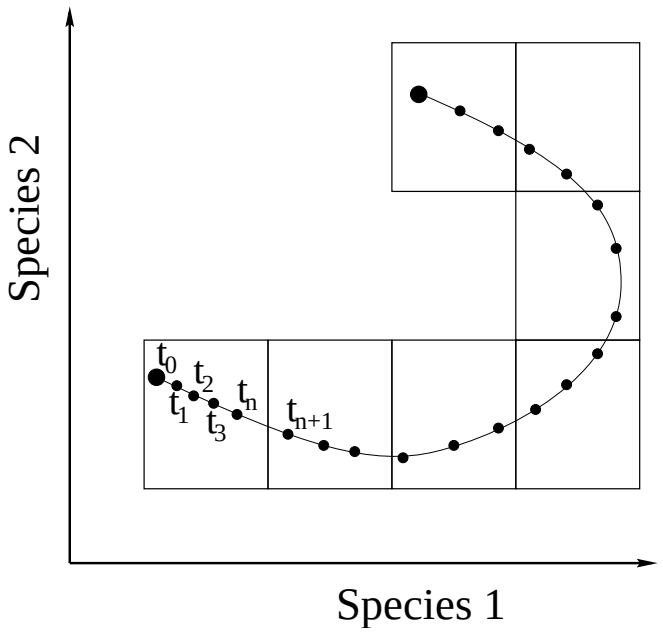

Figure 1: The temporal progress of a 2-dimensional reaction trajectory.

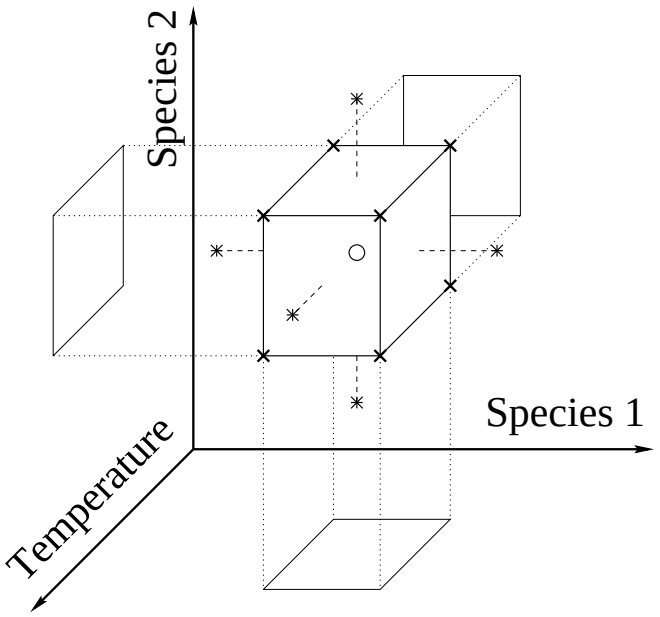

Figure 2: A diagramatic respresentation of the points about a hypercube used to generate the polynomials. The circle is at the center and the crosses are the corners of the hypercube. The star points are also shown.

tion and time-step, and 10 to 20 K for temperature. To parametrize the response of the ODE solver, the solver needs to be called repeatedly at selected points about the hypercube. Each point corresponds to a set of concentrations, a temperature and a time-step length. The concentrations and temperature are propagated by the ODE solver for the length of time specified, returning a set of concentrations and temperature.

At first glance, the number of selected points required is 2^{N_s+2}, a quickly intractible number for multidimensional systems. However, using the methods of surface response theory, the number of points can be reduced to a manageable level. This is accomplished by defining the variables along some of the dimensions as functions of the variables along other dimensions using orthogonality. The optimal placement of these points in our problem is determined by the use of orthogonal composite designs based on the 2_V^{11-4}, 2_{IV}^{11-5} and 2_{IV}^{11-6} fractional factorial designs.[18] The points are largely on corners of the hypercube, with an additional point at the hypercube center and one more outside each face, known as a star point (see Fig. 2 for a diagramatic 3 D representation). The number of points on the corners is specified by the superscript on the design nomenclature. The Roman numeral subscript is a measure of the mixing of the polynomial coefficients. The first design, 2_V^{11-4}, has $2^{11-4} = 128$ points on the corners. The hypercube center point and star points increase this to 151. The three factorial designs specified above use 151, 87, and 55 points, respectively. A detailed example of the procedure is provided in previous work.[19] Orthogonal designs have an important numerical advantage. Determination of the polynomial expressions from the computed data points is streamlined because the covariance matrix is diagonal, thus avoiding the general solution of the normal equations.

We found it useful to slightly shift the corner points toward the hypercube center because placing a point exactly at a corner wastes resources by unnecessarily parametrizing space outside the physical hypercube. By trial and error reducing the distance of these points to the hypercube center by 25% maximized the accuracy.

To obtain the concentrations, $C_i^{t+\Delta t}$, and temperature, $T^{t+\Delta t}$, at the end of a time-step, the construction process results in a polynomial expression composed of quadratic terms of each species concentration, C_i^t, the temperature, T^t, and time-step, Δt. Thus the concentration of species i, $C_i^{t+\Delta t}$, is given by:

$$\begin{aligned}
\ln C_i^{t+\Delta t} \;=\; & a_{i,0} + \sum_j^{N_s} a_{i,j} \ln C_j^t + a_{i,N_s+1} \frac{1}{T^t} \\
& + a_{i,N_s+2} \ln \Delta t + \sum_j^{N_s} \sum_{k \le j}^{N_s} a_{i,jk} \ln C_j^t \ln C_k^t \\
& + a_{i,[N_s+1][N_s+1]} \frac{1}{T^t} \cdot \frac{1}{T^t} \\
& + a_{i,[N_s+2][N_s+2]} \ln \Delta t \,.\, \ln \Delta t \\
& + \text{ cross terms} \qquad (1)
\end{aligned}$$

where a_i are the polynomial coefficients determined in this procedure. It is now possible to evaluate the N_s+1 polynomials and obtain the time evolution of *any* input point within the hypercube.

2.2 Storage and Retrieval

We now organize hypercube information for placement into a data structure for future re-use. First the hypercube is indexed via the coordinates at its center (concentrations, temperature and Δt) and because we have restricted hypercubes to lie only at regular positions the indices take simple integer values. A key is constructed by concatenating the indices in each dimension in a way that is reproducible and unique. The information placed in the data structure consists of this key, the hypercube size and position, and the polynomial coefficients for N_p=N_s+1 polynomials in N_v=N_s+2 variables. These coefficients account for the major portion of the entry size. The number of coefficients

$$N_p \times \left[1 + (N_v) + \frac{(N_v)(N_v+1)}{2}\right] \qquad (2)$$

times the floating point representation size (8 bytes) makes the entry in the order of 8 kilobytes for N_s=9.

The data structure complex consists of a memory-resident binary search tree, a memory-resident doubly linked list and a direct access disk file, all associated through cross-referencing pointers. The binary tree has at each node the key of a hypercube. During a search for a hypercube, when a key is matched, the polynomial coefficients are retrieved from either the memory-resident list or the disk file. The memory-resident list feature of PRISM was added to enhance retrieval speed as we are working on a MPP machine (CRAY T3E) with no paging of random access memory onto disk-resident virtual memory. Because only a limited number of hypercubes can be held in active memory we implement an algorithm, based on time since last use, to keep only the most-frequently used as a memory-resident list, while the less frequently used hypercubes slowly drift toward the tail of the list and finally "fall off the edge." Since copies of all hypercubes exist on the disk file from the moment they are created, they can always be brought back if needed. Additionally, the hypercubes can be stored on disk between simulations and reused in subsequent runs which have the same reaction set, a substantial saving on repeat calculations.

In summary, given a point $\mathbf{r}^t(C_1^1, C_2^t, \ldots, C_{N_s}^t, T^t)$ and a time-step, Δt the PRISM procedure is comprised of the following steps:

1. Determine within which hypercube it lies and calculate the key
2. Traverse binary search tree until a successful key match is made
3. Retrieve the coefficients from either memory or disk, and
4. Evaluate the polynomials to obtain $\mathbf{r}^{t+\Delta t}(C_1^{t+\Delta t}, C_2^{t+\Delta t}, \ldots, C_{N_s}^{t+\Delta t}, T^{t+\Delta t}) = \phi(\mathbf{r}^t, \Delta t)$.

2.3 Accuracy

For our method to be useful it needs two qualities: accuracy and economy. The second is quite obvious: a significant calculational speedup is all that is required, and we discuss this in Section 4. The question of accuracy is more difficult to quantify. What is an acceptable error per time-step? Do these errors propagate in simulation time giving inaccurate results? If so, can these errors be reduced to provide accurate results? Care must be taken for time-dependent problems as we will not know whether our solution is unacceptable unless it appears physically implausible.

A measure of accuracy available at the moment of construction would be very helpful to gauge the validity of a hypercube's polynomial expressions. Since we determined the exact solution at many points within the hypercube, we evaluated the polynomials at the same points and obtained a mean-square difference to use as a measure of accuracy of the hypercube. This value was useful for a coarse screening of hypercubes with poorly fitted polynomials, however, it was not effective at smaller error levels. The use of a single mean-square value does not provide us with an adequate quality gauge. This was not pursued further during the current study. Instead a conservative approach was taken, setting the operational parameters of *all* hypercubes to levels where there is an acceptable error per time-step. The procedure used to determine the operational parameters and investigate the errors per time-step is to first run a combustion simulation with an ODE solver, and write to file thousands of sample points in chemical composition space at the beginning and end of a time-step. We then input these points to PRISM, treating each one independently, propagating each for just one time-step, and comparing with the ODE solver "exact" solution. The error for the concentrations is expressed as

$$\frac{(\mathbf{C}_j^{\mathrm{PRISM}} - \mathbf{C}_j^{\mathrm{exact}})}{\mathbf{C}_j^{\mathrm{exact}}}$$

with a similar expression for temperature. In this way we isolate the error per time-step from the effects of error accumulation over time and evaluate accuracy of the method over a large region of chemical composition space.

Table 2: Dependence of accuracy on variation of operational parameters. Concentration bin is $\log_{10}$ of concentration, eg. 0.5 implies a bin width of half an order of magnitude. T bin is in Kelvin. The Δ quantities are relative errors, ie ((PRISM - Exact)/Exact) and are shown for temperature, one major species and one minor species. The temperature enters the polynomials as a reciprocal in all cases except line 8 for which it is linear.

	number of	bin width		accuracy ($\times 10^3$)		
	data points	$\log_{10}$(conc)	T (K)	ΔT	ΔH_2O	ΔOH
1	87	0.5	20	1.02	6.34	9.46
2	87	0.25	20	0.15	1.09	1.86
3	87	1.0	20	2.89	40.64	41.35
4	87	0.5	10	1.00	6.44	11.06
5	87	0.5	40	1.13	7.03	9.49
6	151	0.5	20	0.79	4.70	9.12
7	55	0.5	20	3.00	23.71	38.74
8	87	0.5	20	1.29	6.35	9.46
9	151	0.25	20	0.091	0.658	0.678

The factors that influence the accuracy of the polynomial expression are:

Hypercube size: Accuracy improves as hypercube size decreases and parametrization becomes more accurate. The hypercube should be as large as possible without accuracy being sacrificed because this reduces the total number of hypercubes, e.g., filling a given region of space with hypercubes of half the size would require 2^{N_s+2} as many hypercubes! With respect to expense of generation as well as storage requirements, larger sides are advantageous. The trend of decreasing error with decreasing edge size can be seen in lines 1,2, & 3 of Table 2 where the size is varied from 0.25 to a full order of magnitude. Independently we have varied the temperature interval, (see lines 1,4 & 5) and see the same behaviour.

Number of points: The accuracy also depends on the number of points used in the polynomial fitting procedure. Details on the decisions that affect the number and placement of these points are described earlier in sub-section 2.1. Lines 1,6 & 7 of Table 2 show the degradation of accuracy as the number of points is decreased. For our 11-dimensional application we changed the number of points used and considered 151, 87 and 55 points. Increasing from 87 to 151 points slightly improves accuracy, and decreasing to 55 points reduces accuracy significantly. Note that these accuracies are obtained with far fewer than the $2^{N_s+2} = 2048$ points that one has without the benefit of surface response theory.

Enthalpy and elemental mass conservation: The problems that we consider have the property of being constrained by conservation relations, of which we take advantage. The mapping of a point over a time-step should conserve both total enthalpy and number of atoms of each element involved. While the ODE solver that supplies the points conserves these quantities very well, the resulting fitted polynomials do not always do so. Part of the error inherent in using a parametrized polynomial may be reduced by enforcing conservation. Our final solution is mapped a small distance from the exact final solution. In N_s+1 space the exact reaction trajectory of the system will always follow a hypersurface on which enthalpy and mass are conserved. By applying mass and enthalpy conservation our final point is shifted back onto that hypersurface, not necessarily onto the exact final point but hopefully closer to it. (We have an under-determined system with N_s+1 variables but only N_{elem}+1 conservation equations where N_{elem} is the number of elements in the system.) We impose conservation after the polynomial has been evaluated by adjusting the final values of the species concentrations using a simple approach consisting of the following steps: Set up N_{elem}+1 conservation equations; allow N_{elem}+1 species concentrations to vary, requiring that the remaining concentrations be constant; and finally solve the resulting N_{elem}+1 simultaneous linear equations.

3 Results

We test PRISM on three different combustion simulations and compare the result at the end of the each simulation (several milli-seconds) to that obtained from using an ODE solver directly. All simulations use the same 8-species H_2 reaction set with inert N_2, (see Table 1) evolve with time and allow for the investigation of the temporal propagation of errors. Based on the errors in Table 2 we have chosen a set of operational parameters (line 9) that give an error per time-step of approximately 1.0×10^{-3}. When constructing hypercubes we use 151 points for the polynomial fitting procedure, with hypercube sides of 0.25 of order of magnitude for species concentrations and Δt, and 20 K for temperature. For the latter two simulations described below the CFD was done with Coyote[20] within which the chemistry calculation used the DVODE differential equation solver[21] and the CHEMKIN thermodynamic library.[22]

3.1 Zero-Dimensional

In the first example we consider premixed combustion that takes place purely in chemical composition space, with zero physical dimensions so that the reaction trajectory is determined solely by chemical kinetics. The initial concentration corresponds to a mixture of stoichiometric H_2-air with an initial temperature of 1200 K, high enough for burning to commence. The time-step is fixed at 1 μs. The simulation is started, the fuel is consumed and the system comes to rest adiabatically at an equilibrium temperature of 2819 K after 0.1 ms (100 time-steps). We continue the simulation for an additional 1.9 ms (1900 time-steps). Figure 3 shows the progress of the temperature and concentrations of H_2O and OH with time. The final temperatures in Fig. 3a are 2819 K and 2815 K for PRISM and the ODE solver respectively. Similarly for H_2O in Fig. 3b the final concentrations are 2.475×10^{-6} and 2.480×10^{-6} moles/cm^3 respectively. The agreement in rise time, final temperature and concentrations, and the fact that the final equilibrium state stays correct for a long period indicate that errors are not accumulating over simulation time.

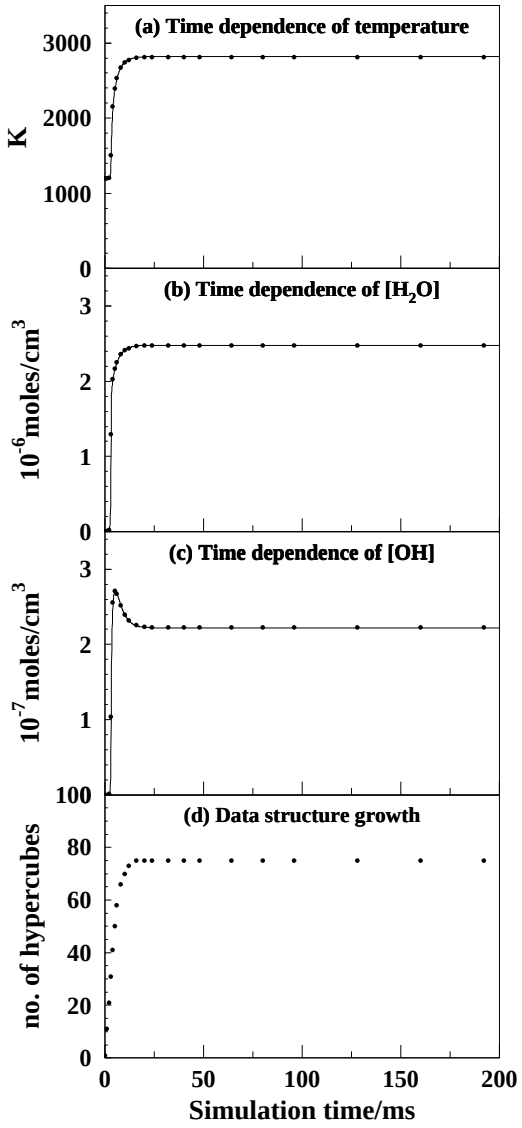

Figure 3: Comparison of results from ODE solver (solid line) to PRISM (symbol) in a zero-dimensional burn showing the temporal progress (ms) of (a) temperature (K), (b) one major species concentration (moles/cm^3), (c) one minor species concentration (moles/cm^3), and (d) data structure size.

3.2 1-Dimensional Laminar Flame

In the second example we consider a more complex simulation that includes the influence of inter-cellular convection and diffusion: a propagating premixed laminar flame in an open 1-dimensional tube. The physical configuration is a 1 cm long (200 cells) tube, closed at one end and open at the other end to atmospheric pressure. A small portion of the tube near the open end is filled with hot burned gas, while the remainder is filled with stoichiometric H_2-air at room temperature. A flame forms at the interface between the two gas mixtures and propagates toward the closed end. Panels a and b of Fig. 4 show snap-shots of the flame front (temperature and species concentration profiles) for both a direct ODE run and a

PRISM run. The flame speed is reproduced very closely. More importantly, no error accumulation is evident even after 90,000 time-steps. We also see from Fig. 4c that most of the required polynomials have been constructed by the time the flame front has reached 0.2 cm (1 ms). Thereafter, PRISM proceeds mostly by retrieving polynomials from the established data structure.

3.3 2-D Axisymmetric Turbulent Jet

In the final example we have simulated a non-premixed 2-dimensional turbulent jet with coaxial H_2 and air inflows, starting from a quiescent non-combusting state and proceeding until a quasi-steady turbulent flame is attained. The physical configuration is a cylindrical chamber of radius 8 cm and height 20 cm, open at the top to atmospheric pressure with two concentric inlets at the center of the base. The conditions are H_2 at 21 m/s and 300 K in the inner jet of radius 0.35 cm, and air at 1 m/s and 300 K in the outer jet, which has radial extent from 0.5 to 8 cm. The chamber is initially filled with air at 1600 K to initiate combustion. We run for sufficient time that fuel initially entering the chamber has made a transit to the upper (exhaust) boundary, and until the quasi-steady state is reached. The elapsed problem time is about 275 ms. Once again the direct ODE is compared to PRISM. The emphasis is on the similarity between ODE and PRISM profiles, and not on any features of the profiles themselves, which are a result of our physical configuration and initial conditions. Panels a and b of Fig. 5 show a snapshot of the profiles of temperature and several species from the direct ODE solver and PRISM. This profile is taken at a horizontal slice half-way up the chamber, at an intermediate time of 45 ms. Figure 5c shows the time variation of major species concentrations from 0 to 275 ms taken at a sample point midway up the chamber. The agreement again is very good. The overall ranges in species concentrations that occurred over the duration of this simulation are shown on Fig. 6. The prediction of PRISM follows that of the ODE solver over time across 5–6 orders of magnitude. The growth of the data structure (Fig. 5d) levels off at about 150 ms with a size of 140000 entries, 125 ms before steady state is reached.

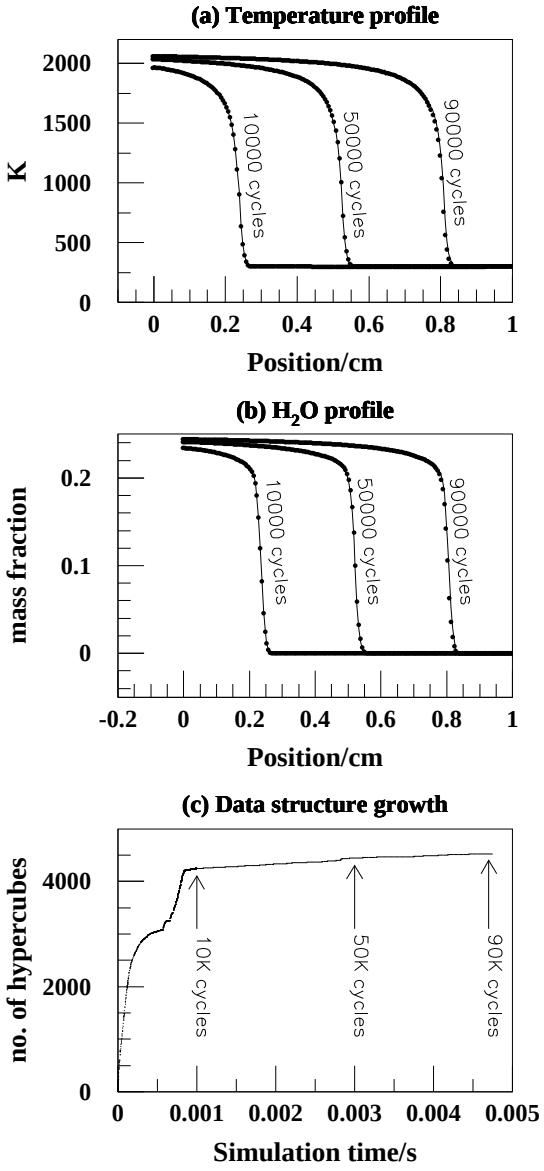

Figure 4: Comparison of results from ODE solver (solid line) to PRISM (symbol) in a 1-D laminar flame tube showing the temporal progress of the flame front by observing the (a) temperature (K) profile and (b) H_2O mass-fraction profile. (c) The size of the data structure vs simulation time (seconds) is also plotted.

4 Discussion

For the PRISM method to be successfully used in chemical kinetic calculations, it requires ease of setup, accuracy, and run-time economy.

Ease of setup: The computer codes have been written modularly and documented with the expectation of future improvement and inclusion of interfaces to generic operator-split CFD codes.

Accuracy: The results of Section 3 make it evident that a full turbulent initial condition simulation can be run for 275 ms without perceptible differences. We see from Fig. 5c that species concentrations begin to deviate from

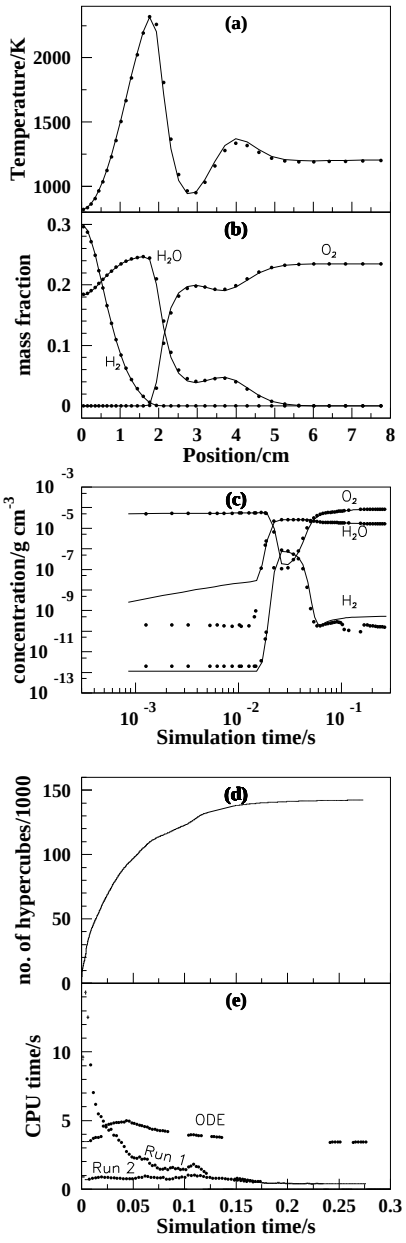

Figure 5: Comparison of results from ODE solver (solid line) to PRISM (symbols) in a 2-D turbulent jet. (a),(b) Profile snapshots of temperature (K) and several species mass-fractions taken at a horizontal slice midway up the chamber at time=45 ms. (c) Time variation (seconds) of major species concentrations (g/cm^3) taken at a sample point midway up the chamber. (d) Size of data structure vs simulation time (seconds) (e) CPU time (seconds) per cycle vs simulation time (seconds) over the course of an ODE run and 2 PRISM runs.

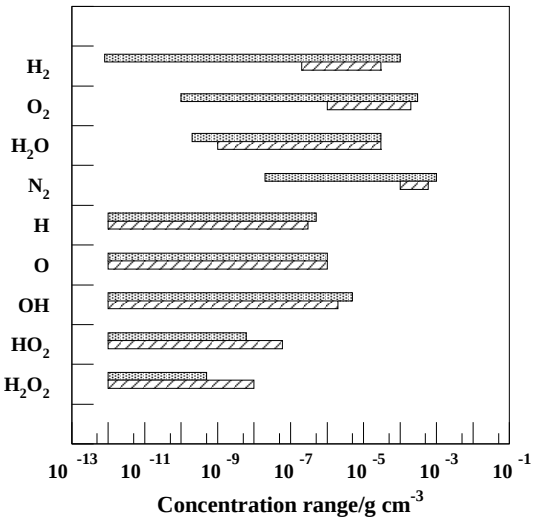

Figure 6: Ranges of species concentrations (g/cm^3) that occurred over the duration of the turbulent jet (shading) and laminar flame (diagonals) simulations.

the ODE solution only at orders of 1.0×10^{-10} g/cm^3.

Run-time economy: Whether the method is economical or not entails weighing the expense of using an ODE solver against factors such as the expense of the construction of the polynomials, the retrieval time, the evaluation time, and the re-use of existing polynomials. There is the additional question of whether a particular combustion simulation is to be run repeatedly with a few CFD parameter changes between runs. In such a case, a data structure written to disk would be read back and re-used. Below we present some benchmarks for the 9-species H_2 reaction set, run on a single Alpha processor of a Cray T3E:

The construction expense has contributions from repeatedly calling the ODE solver for all the points as well as the parametrization and storage of the coefficients. The mean CPU time for one ODE solver call is 3.7×10^{-3} seconds. The mean CPU time for generating and storing one hypercube is about 1 sec if 87 ODE calls are made and 1.7 sec for 151 ODE calls.

Retrieval consists of searching the binary tree for the correct hypercube and then recalling its polynomial coefficients from either the memory-resident list or the direct-access file. A disk file access incurs a real-time penalty which cannot be seen in the CPU numbers. This depends on the system load from other users and can take between a factor of 4 to 400 times longer than a memory load. The memory-resident list is sufficiently large that

disk accesses are few. We find that CPU time used has only a weak dependence on the depth of the binary search tree: $0.25\times10^{-3} + 1.5\times10^{-6}$ N_L where N_L is the number of tree levels descended. The tree search does not consume significant CPU time until N_L approaches 100. A perfectly balanced binary tree with N_{tot} nodes will have about $N_L \approx \log_2 N_{tot}$. In the turbulent jet case a balanced tree depth would have been $\log_2 140000 \approx 17$. In practice the tree is not balanced (perfect balance would require that nodes be added to the tree in a unique sequence) hence we periodically call a procedure to dynamically rebalance the tree when N_L exceeds $\log_2 N_{tot}$ significantly.

Evaluation of the polynomial is done by the algebraic calculation of Equation 1 followed by the mass/enthalpy conservation calculation. The mean CPU time for retrieval and evaluation is 0.4×10^{-3} seconds of which one quarter of the time is used by the mass conservation calculation.

Currently we have a factor of $3.7\times10^{-3} \div 0.4\times10^{-3} \approx 9$ decrease in CPU time over the ODE solver per calculation. Combining this information with the construction cost, to be economical in a single run, the average re-use of hypercubes needs to be about 300 before the construction cost is recouped, a figure which we easily meet in our applications. Average re-use was 8000 for the laminar flame and 7000 for the turbulent jet. Repeated runs that make use of the existing hypercubes have even better CPU economy. Figure 5e compares the CPU time used per cycle summed over 1500 CFD cells for an ODE run and two PRISM runs. PRISM Run 1 is started with no pre-existing hypercubes, i.e. all hypercubes need to be generated during the run. Run 2 is a repeat of the simulation that re-uses the hypercubes generated by Run 1. The ODE chemical calculation is more or less constant at 4 s per cycle. In Run 1, chemistry is more expensive in the beginning but after 20 ms of simulation time it becomes more economical, even though the data structure (Fig. 5d) has only reached one third of its final size. By 150 ms it uses 0.4 s of CPU time per cycle. Run 2 reads in the data structure disk file generated by Run 1. In principle these hypercubes should be sufficient for a repeat of the simulation, however because of errors, the reaction trajectory wanders into parts of chemical composition space that it had not visited during Run 1. Thus, for Run 2, generation of a few new hypercubes raises mean CPU usage to 0.8 s, that drops to 0.4 s later in the run. This wandering did not affect the accuracy of the results. A comparison of the areas under the various curves in Fig. 5e gives a good idea of the economy of the PRISM chemistry. The overall economy including the cost of the CFD (0.6 s per cycle) is 4.0 + 0.6 = 4.6 s for the ODE case, and 0.6 + 0.6 = 1.2 s for Run 2, a factor of four.

A discussion on economy would not be complete without mention of memory used by the data structure. A stored hypercube uses about 8KB of memory, mostly used for polynomial coefficients. This can be reduced by a factor of two by using 4-bytes of storage per word. At 7 significant digits the accuracy will not be degraded. At the end of the turbulent jet simulation the data structure contained 140000 hypercubes using about 1 gigabyte of disk space, which at US$20/GB is quite manageable. We allowed the most frequently used 15000 hypercubes to be memory-resident (120 MByte) which is what our available processor permitted. The advantage of dynamically growing the data structure for only those portions of chemical composition space actually needed is seen from Fig. 6 which shows the ranges in concentration observed for each simulation for each species. If a multi-dimensional "box" was made from these ranges and filled with hypercubes it would need on the order of 10^{12} hypercubes instead of the 140000 used for the turbulent jet case.

The method of using piecewise solution mapping in a 10 dimensional chemical composition space to reproduce the result of an ODE system for a complex, time-evolving combustion system, encompassing a wide range of concentrations and temperatures provides an accurate and economical result. The method is particularly attractive in cases where a series of CFD simulations are required, differing only slightly in a few parameters, so that an existing data structure can be reused many times.

Acknowledgement: This research was supported by the Director, Office of Energy Research, Office of Basic Energy Sciences, Chemical Sciences Division of the U.S. Dept. of Energy under contract No. DE-AC03-76SF00098.

References

[1] Tonse, S., unpublished result, obtained with the Coyote CFD reactive flow code.

[2] Frenklach, M.; Kailasanath, K.; Oran, E. S., *Progress in Astronautics and Aeronautics* **1986**, *105*, 365–376.

[3] Frenklach, M., In *Complex Chemical Reaction Systems, Mathematical Modelling and Simulation*, Warnatz, J.; Jäger, W., Eds., vol. 47 of *Springer Series in Chemical Physics*, Springer-Verlag, Berlin, 1987 pp. 2–16.

[4] Wang, H.; Frenklach, M., *Combust. Flame* **1991**, *87*, 365–370.

[5] Frenklach, M., In *Numerical Approaches to Combustion Modeling*, Oran, E. S.; Boris, J. P., Eds., American Institute of Aeronautics and Astronautics, Washington, D.C., 1991 pp. 129–154.

[6] Hewson, J. C.; Bollig, M., In *Twenty Sixth (International) Symposium on Combustion*, The Combustion Institute, 1996 pp. 2171–2179.

[7] Peters, N.; Williams, F. A., In *Complex Chemical Reaction Systems, Mathematical Modelling and Simulation*, Warnatz, J.; Jäger, W., Eds., vol. 47 of *Springer Series in Chemical Physics*, Springer-Verlag, Berlin, 1987 pp. 310–317.

[8] Ramshaw, J. D., *Phys. Fluid.* **1980**, *23*, 675.

[9] Maas, U.; Pope, S. B., *Combust. Flame* **1992**, *88*, 239–264.

[10] Lam, S. H.; Goussis, D. A., *Int. J. Chem. Kinet.* **1994**, *26*, 461–486.

[11] Brown, N. J.; Li, G.; Koszykowski, M. L., *Int. J. Chem. Kinet.* **1997**, *29*, 393–414.

[12] A. R. Marsden Jr.; Frenklach, M.; Reible, D. D., *J. Air Pollut. Control Assoc.* **1987**, *37*, 370–376.

[13] Turanyi, T., *Comp. Chem.* **1994**, *18*, 45–54.

[14] Bray, K. N. C.; Peters, N., In *Turbulent Reacting Flows*, Libby, P. A.; Williams, F. A., Eds., Academic Press, San Diego, CA, USA. 92101-4311, 1994 pp. 63–113.

[15] Pope, S. B., *Combust. Theory Modelling* **1997**, *1*, 41–63.

[16] Yang, B.; Pope, S. B., *Combust. Flame* **1998**, *112*, 85–112.

[17] Sun, F. S.; Markatou, P.; Frenklach, M., unpublished results.

[18] Box, G. E. P.; Draper, N. R., *Empirical Model Building and Response Surfaces*, John Wiley and Sons, **1987**.

[19] Frenklach, M.; Wang, H.; Rabinowitz, M. J., *Prog. Energy Combust. Sci.* **1992**, *18*, 47–73.

[20] Cloutman, L. D., Tech. Rep. UCRL-ID-103611, Lawrence Livermore National Laboratory, **1990**.

[21] Brown, P. N.; Byrne, G. D.; Hindmarsh, A. C., *SIAM J. Sci. Stat. Comput.* **1989**, *10*, 1038–1051, Also, LLNL Report UCRL-98412, June 1988.

[22] Kee, R. J.; Rupley, F. M.; Meeks, E.; Miller, J. A., Tech. Rep. SAND96-8216, UC-405, Sandia National Laboratory, **1996**.