

Western States Section of the Combustion Institute–Fall 2017 Meeting  
Hosted by the University of Wyoming  
October 2–3, 2017

## Investigating stiffness detection metrics for chemical kinetics

Andrew Alferman<sup>1</sup> and Kyle E. Niemeyer<sup>1,\*</sup>

<sup>1</sup>*School of Mechanical, Industrial, and Manufacturing Engineering, Oregon State University,  
Corvallis, OR 97331, USA*

<sup>\*</sup>*Corresponding author: Kyle.Niemeyer@oregonstate.edu*

**Abstract:** Simulations of combustion and reacting flows often encounter stiffness in the equations governing chemical kinetics. Explicit solvers for these ordinary differential equations offer low computational expense, but typically cannot efficiently handle stiff systems. In contrast, implicit methods demand greater expense but offer unconditional stability?as a result, most reactive flow solvers rely on these methods by default (other than explicit direct numerical simulation solvers). However, explicit or stabilized explicit methods can instead be used to reduce the computational expense while remaining stable and accurate if the chemical kinetics systems exhibit low-to-moderate stiffness. This study investigates metrics for quantifying stiffness, with the goal of identifying one capable of efficiently and robustly determining the appropriate category of integrator required. Methods of measuring the stiffness of chemical kinetics states will be investigated, including stiffness ratio, stiffness index, stiffness indicator, and chemical explosive mode. These will be applied to simulations of hydrogen/carbon monoxide and methane autoignition using initial conditions representative of realistic turbulent combustion, obtained from partially stirred reactor simulations. The stiffness quantification metrics will be compared with the number of right hand side function calls required to integrate using implicit and explicit methods. We will conclude with preliminary performance analysis of an integrator scheduler using these metrics.

**Keywords:** *Stiffness quantification, Ordinary differential equations, Chemical kinetics, Computational cost reduction*

### 1. Introduction

An increasing demand for greater efficiency and lower CO<sub>2</sub> emissions in the US energy supply has driven the development of next-generation combustion technologies that use alternative fuels [1]. Computational modeling is one important tool that increasingly drives design and development of these new technologies. Shortening the design cycle and time-to-market of novel and efficient energy devices optimized for alternative fuels requires fundamental advances in combustion science and fuel chemistry; both depend on high-fidelity computational tools [2]. The need to develop new, high-performance computational methods to support combustion research has been recognized by multiple federal agencies as a key objective towards implementing predictive models [3–6].

Species time scales can range from the order of nanoseconds to seconds, requiring greater computational expense for integration by conventional methods (i.e., numerical stiffness) [7]. Most combustion modeling approaches rely on a single, implicit ODE integration method to handle the chemistry in all spatial locations, but these methods are computationally expensive, especially for larger mechanisms. The computational cost of chemistry is either a quadratic or cubic function of the number of species [7]; reaction mechanisms of sizes comparable to those shown above pose computational difficulties even in zero-dimensional (homogeneous) simulations.

Depending on the local conditions, computational fluid dynamics flow time-step size, and chemical mechanism being used, it is possible to encounter a wide range of stiffness within a single simulation. We can exploit this situation to reduce the overall simulation expense by selecting the most computationally efficient ODE solver on-the-fly based on local conditions. For example, a low-cost explicit method may be used far away from the reaction zone, while an implicit—or otherwise “stiff”—solver is more economical inside a flame. Making this selection requires a method to detect and classify stiffness [2].

Although the concept of stiffness has been identified for over 60 years, the term has not been precisely defined despite repeated efforts. The diverse set of problems considered to be “stiff” and the large variety of characterizations used to describe stiffness are amongst the difficulties that have been encountered in developing a precise definition [8]. Nonetheless, a variety of stiffness quantification methods have been developed with the goal of providing a practical means of evaluating a systems ODEs [8–12]. We look to these methods of stiffness quantification to determine their usefulness with respect to the equations governing combustion, and to determine if a reliable and efficient means of switching methods can be developed from it. In doing so, we hope to reduce the computational expense of combustion simulations.

## 2. Methods

We considered two different chemical kinetic models in this study: the relatively small hydrogen/carbon monoxide ( $\text{H}_2/\text{CO}$ ) model of Burke et al. [13] and GRI Mech 3.0 [14], which uses  $\text{CH}_4$  as fuel. The  $\text{H}_2/\text{CO}$  model uses 13 species while the GRI Mech 3.0 is larger and uses 53 species, allowing for Jacobian matrices of system of ODEs of  $13 \times 13$  and  $53 \times 53$ , respectively. These relatively small Jacobian matrix allows for much faster evaluations of stiffness than would be possible with larger, more complicated kinetic systems, which can have hundreds to hundreds of thousands of chemical species [15].

The code used to run each autoignition simulation and calculate all stiffness metrics was written in Python and made use of open-source software including the ode integrator function of SciPy [16]. Two different integrators of the ode integrator function were used: the implicit vode integrator, which uses an implicit Adams method for non-stiff problems and a method based on the backwards differentiation formulae (BDF) for stiff problems [17], and the explicit dopri5 integrator, which uses a Runge-Kutta method of order (4)5 developed by Dormand and Prince with automatic step size control [18]. NumPy [19] was also extensively used in the code to determine eigenvalues and perform numerous basic mathematical functions.

Computation of the right hand side (RHS) function and the Jacobian matrix of the systems of ODEs were readily achieved using pyJac [20], a Python-based open-source program that generates analytical Jacobian matrices for use in chemical kinetics modeling and analysis [21]. pyJac uses a chemical mechanism developed using Cantera [22] software and a set of initial conditions to generate functions that return the required values. A global variable was added to the RHS function to count the number of calls from the integrator for each time step.

Our analysis was conducted using information gathered from partially stirred reactor (PaSR) simulations. As described by Niemeyer et al. [21], the PaSR model consists of a number of particles, each with a time-varying composition. At discrete time steps, events including inflow, outflow, and pairing cause particles to change composition; between these time steps, mixing and reaction fractional steps evolve the composition of all particles. By using data from a single par-

title of the PaSR model, the evaluation could be simplified to a zero-dimensional analysis. The PaSR data included nine different configurations of temperature and pressure; the initial temperature values were 400, 600, and 800 K, while the pressure values were 1, 10, and 25 atm. Pressure remained constant throughout the simulations.

## 2.1 Stiffness Ratio

One oft-referenced measure of the stiffness of a system of ODEs is the “stiffness ratio.” The stiffness ratio of the system of ODEs is defined as

$$\text{Stiffness Ratio} = \frac{\max|\lambda_p|}{\min|\lambda_p|} \quad (1)$$

in which  $\lambda_p$  is an eigenvalue of the Jacobian matrix of the system of ODEs [23]. A large stiffness ratio is an indication of a large range of time scales in the problem, which is a necessary component for stiffness to arise. This method is readily implemented and carries little computational cost beyond determining the eigenvalues of the Jacobian matrix.

## 2.2 Stiffness Index

One method of interest regarding stiffness quantification was the “IA-Stiffness Index” proposed by Shampine [24]. This method enables comparison of values of the stiffness index between different equations and different methods. Additionally, the method takes into account the impact of the order of the method selected when evaluating the stiffness. The method is relatively straightforward to implement, requiring computation of a vector of the derivatives of the system of equations, as well as either a weighted norm or the spectral radius of the Jacobian matrix.

The IA-stiffness index of a method of order  $p$  introduced by Shampine [24] is

$$\frac{h_{acc}}{h_{iter}} \doteq \tau^{1/(p+1)} \rho[A] \|y^{(p+1)}(x_n)\|^{-1/(p+1)} \left( \frac{|\xi|^{-1/(p+1)}}{|\gamma|} \right) \quad (2)$$

or alternatively,

$$\frac{h_{acc}}{h_{iter}} \doteq \tau^{1/(p+1)} \|A\| \|y^{(p+1)}(x_n)\|^{-1/(p+1)} \left( \frac{|\xi|^{-1/(p+1)}}{|\gamma|} \right) \quad (3)$$

where  $h_{acc}$  represents the largest step size which would result in a local accuracy test being passed,  $h_{iter}$  represents the minimum step size that will lead to divergence of simple iteration,  $\tau$  represents the specified tolerance,  $\rho[A]$  represents the spectral radius of the Jacobian matrix  $A$ ,  $\gamma$  is a constant characteristic of the formula, and  $\xi$  represents a constant characterizing the accuracy of a reference method. Note that these equations require the  $p + 1$  derivative of the function  $y(x_n)$ . Additionally, the matrix norm  $\|M\|$  of matrix  $M$  with  $i$  columns and  $j$  rows is given by

$$\|M\| = \max_i \frac{1}{w_i} \sum_j |M_{ij}| w_j \quad (4)$$

in which  $w_i$  and  $w_j$  are positive weights of the matrix [9].

Although the IA-stiffness index is useful in determining the stiffness of a method for a given system of equations, we are interested in quantifying the stiffness inherent to the system of equations itself. Such a quantification is necessary for investigating method switching mechanisms. Shampine notes that the scalar quantity

$$\text{Stiffness Index} = \rho[A] \|y^{(p+1)}(x_n)\|^{-1/(p+1)} \quad (5)$$

provides a fair “stiffness index” to a given problem [9]. For the remainder of this document, all references to the stiffness index will refer to the value obtained using equation (5).

As previously noted, use of the above equations requires calculation of a vector of the derivatives of a system of equations. In the case of this investigation, this vector is comprised of the derivatives of the thermochemical composition vector with respect to time, with the vector defined as

$$\Phi = \{T, Y_1, Y_2, \dots, Y_{N_{sp}}\}^T, \quad (6)$$

where  $T$  is the temperature,  $Y_i$  are the species mass fractions, and  $N_{sp}$  is the number of species in the mechanism [21]. The vector needed to compute the stiffness index is therefore

$$\frac{\partial \Phi}{\partial t} = \left\{ \frac{\partial Y_1}{\partial t}, \frac{\partial Y_2}{\partial t}, \dots, \frac{\partial Y_{N_{sp}}}{\partial t} \right\}^T, \quad (7)$$

where time is denoted by  $t$  [21]. In addition to this vector of derivatives, computation of the stiffness index requires either the weighted norm or the spectral radius of the Jacobian matrix for the thermochemical composition vector.

After generating this data using pyJac, the values of the second derivative of the thermochemical composition vector were calculated numerically using a fourth-order central differencing formula. Numerical approximations of higher-level derivatives may also be generated using the same approach. Use of the central differencing formula was possible because the stiffness index was calculated after solving for the thermochemical composition vector, and the reaction was modeled using time steps of a constant size. Forward and backward difference formulae were used at the boundaries where central differencing could not be used. Variable time step methods for calculating the second derivative or higher-level derivatives may also be used, however these methods were not necessary at the current stage of the investigation.

To facilitate comparison of the results obtained by Shampine [9], the order of the method was assumed to be 1.

### 2.3 Stiffness Indicator

The “stiffness indicator” of the system of ODEs is given by

$$\text{Stiffness Indicator} = \frac{m[f_y(x_n, y(x_n))] + M[f_y(x_n, y(x_n))]}{2} \quad (8)$$

where

$$m[A] = \min \lambda[\text{He}(A)]; \quad M[A] = \max \lambda[\text{He}(A)], \quad (9)$$

with  $\lambda[A]$  denoting the eigenvalues of matrix  $A$ ,  $\text{He}(A)$  denoting the Hermitian part of the matrix  $A$ , which is equal to  $(A + A^T)/2$ .

## 2.4 Chemical Explosive Mode

The “chemical explosive mode analysis” (CEMA) was developed as a diagnostic to identify flame and ignition structure in complex flows [25]. The chemical explosive modes are associated with positive eigenvalues of the Jacobian of the system of ODEs:

$$\text{Re}(\lambda) > 0. \quad (10)$$

Similar to the stiffness ratio, the CEMA is readily implemented and carries little computational cost beyond determining the eigenvalues of the Jacobian matrix.

## 3. Results and Discussion

Initial conditions were picked using particles of the PaSR simulation where weak reactions were taking place but ignition had not yet occurred. The simulation was run until 0.2 seconds of data was obtained, in which time the particles exhibited autoignition. Using state data obtained during the simulation, the Jacobian matrix was calculated at each time step using pyJac, which allowed the numerical values of each stiffness matrix to be calculated.

An alternative analysis of each stiffness metric and required RHS function calls was performed using every particle in the PaSR simulation. In this analysis, a simulation at each particle of the PaSR was advanced only as long as needed to obtain an accurate numerical value of each stiffness metric. The number of RHS function calls was measured just like the earlier analysis, however a wider range of conditions could be represented by obtaining data at each particle of the PaSR model.

### 3.1 Results

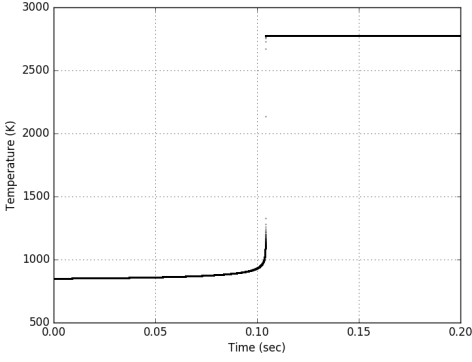
## 4. Conclusions

## 5. Acknowledgements

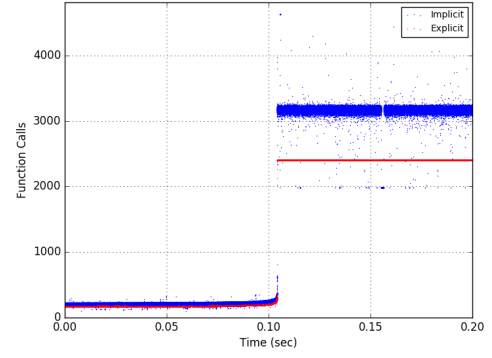
This material is based upon work supported by the National Science Foundation under grant ACI-1535065.

## References

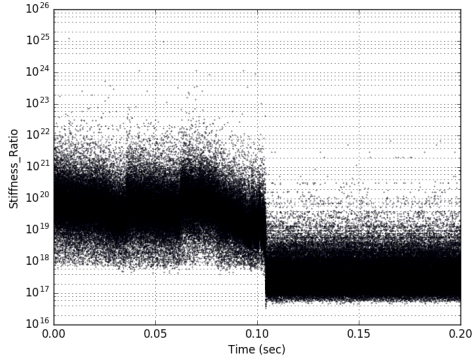
- [1] A. H. Epstein, Aircraft engines’ needs from combustion science and engineering, *Combustion and Flame* 159 (2012) 1791–1792. DOI: 10.1016/j.combustflame.2012.02.022.
- [2] K. E. Niemeyer and O. State, Collaborative Research : SI2-SSE : An intelligent and adaptive parallel CPU / GPU co-processing software library for accelerating reactive-flow simulations, (2015).
- [3] A. Trouvé, D. C. Haworth, J. H. Miller, L. K. Su, and A. Violi, Cyber-Based Combustion Science: Report on National Science Foundation, Available at <http://www-personal.umich.edu/~avioli/pdf/nsf-report.pdf>, 2006.



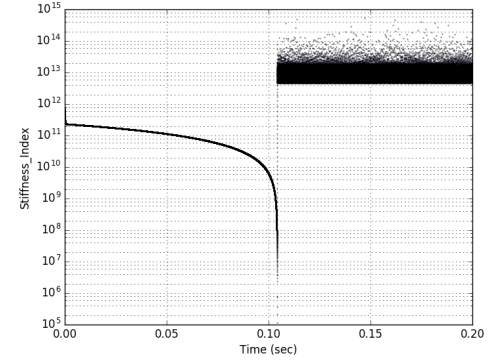
(a) The temperature of the simulation rises from approximately 850 K prior to autoignition to approximately 2775 K, with a residence time of approximately 0.10 seconds.



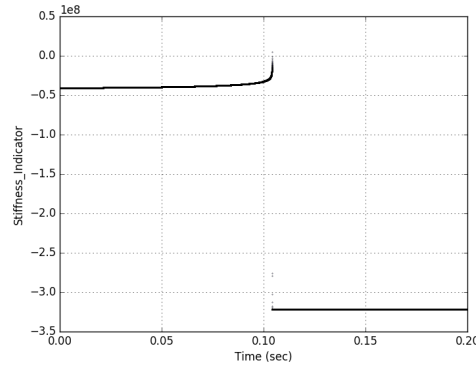
(b) The number of RHS function calls per time step increases slowly prior to autoignition and rapidly increases during ignition, then remains relatively constant and large after ignition.



(c) The stiffness ratio remains large throughout the simulation, but decreases in magnitude after ignition.

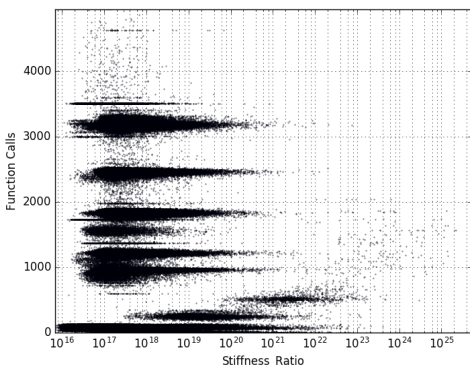


(d) The stiffness index slowly decreases prior to autoignition and increases following autoignition.

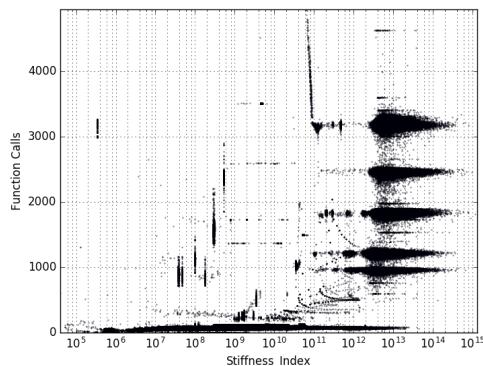


(e) A plot of the stiffness indicator resembles an upside down plot of RHS function calls.

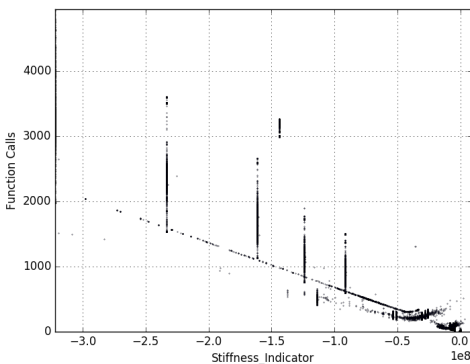
Figure 1: Temperature, RHS function calls, and stiffness metrics of the  $H_2/CO$  reaction calculated with a constant step size of  $10^{-6}$  s, integration time of 0.0–0.2 s.



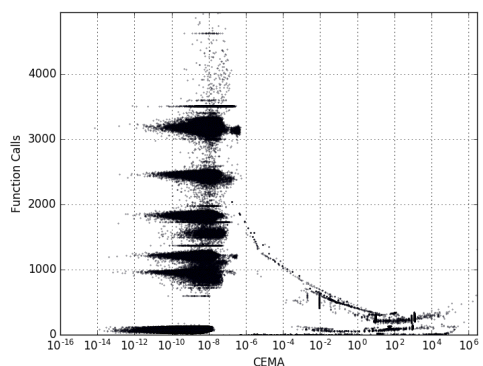
(a) Stiffness Ratio. The majority of particles in each pressure and temperature combination of PaSR data are grouped together.



(b) Stiffness Index. The majority of particles in each pressure and temperature combination of PaSR data are grouped together.



(c) Stiffness Indicator. The majority of particles in each pressure and temperature combination of PaSR data are grouped together, however particles near autoignition exhibit a linear trend.



(d) Chemical Explosive Mode. The majority of particles in each pressure and temperature combination of PaSR data are grouped together.

Figure 2: Stiffness metrics versus RHS function calls of the  $H_2/CO$  reaction calculated with a constant step size of  $10^{-6}$  s for every initial condition represented in the PaSR model.

- [4] U.S. Department of Energy, Basic Research Needs for Clean and Efficient Combustion of 21st Century Tra  
Available at [http://science.energy.gov/~media/bes/pdf/reports/files/ctf\\_rpt.pdf](http://science.energy.gov/~media/bes/pdf/reports/files/ctf_rpt.pdf), 2007.
- [5] National Research Council, Transforming Combustion Research through Cyberinfrastructure, Available at [http://www.nap.edu/catalog.php?record\\_id=13049](http://www.nap.edu/catalog.php?record_id=13049), The National Academies Press, Washington, DC, 2011.
- [6] National Research Council, Future Directions for NSF Advanced Computing Infrastructure to Support U.S. Science in 2017–2020, Available at [http://www.nap.edu/catalog.php?record\\_id=18972](http://www.nap.edu/catalog.php?record_id=18972), The National Academies Press, Washington, DC, 2014.
- [7] T. Lu and C. K. Law, Toward accommodating realistic fuel chemistry in large-scale computations, *Progress in Energy and Combustion Science* 35 (2009) 192–215. DOI: 10.1016/j.pecs.2008.10.002.
- [8] G. Söderlind, L. Jay, and M. Calvo, Stiffness 1952-2012: Sixty years in search of a definition, *BIT Numerical Mathematics* 55 (2014) 531–558.
- [9] L. F. Shampine, Measuring Stiffness, *Applied Numerical Mathematics* 1 (1985) 107–119. DOI: 10.1016/0168-9274(85)90020-0.
- [10] L. Brugnano, F. Mazzia, and D. Trigiante, Recent Advances in Computational and Applied Mathematics, *Recent Advances in Computational and Applied Mathematics* (2011) 1–21, arXiv: 0910.3780.
- [11] J. D. ( D. Lambert, *Computational methods in ordinary differential equations*, Wiley, 1973, p. 278.
- [12] E. Hairer and G. Wanner, *Solving Ordinary Differential Equations II*, vol. 14, Springer Series in Computational Mathematics, Springer Berlin Heidelberg, Berlin, Heidelberg, 1996.
- [13] M. P. Burke, M. Chaos, Y. Ju, F. L. Dryer, and S. J. Klippenstein, Comprehensive H<sub>2</sub>/O<sub>2</sub> kinetic model for high-pressure combustion, *International Journal of Chemical Kinetics* 44 (2012) 444–474. DOI: 10.1002/kin.20603.
- [14] G. P. Smith, D. M. Golden, M. Frenklach, N. W. Moriarty, B. Eiteneer, M. Goldenberg, C. T. Bowman, R. K. Hanson, S. Song, W. C. Gardiner, V. V. Lissianski, and Z. Qin, *GRI-Mech 3.0*, [http://www.me.berkeley.edu/gri\\_mech/](http://www.me.berkeley.edu/gri_mech/), 1999.
- [15] K. E. Niemeyer, *Reducing the Cost of Chemistry in Reactive-Flow Simulations: Novel Mechanism Reduction Strategies and Acceleration via Graphics Processing Units*, PhD thesis, Cleveland, OH: Case Western Reserve University, 2013.
- [16] E. Jones, T. Oliphant, P. Peterson, et al., SciPy: Open source scientific tools for Python, [Online; accessed 2017-02-28], 2001–, URL: <http://www.scipy.org/>.
- [17] P. N. Brown, G. D. Byrne, and A. C. Hindmarsh, VODE: A Variable-Coefficient ODE Solver, *SIAM Journal on Scientific and Statistical Computing* 10 (1989) 1038–1051.
- [18] E. Hairer SP Norsett and G. Wanner, Solving ordinary differential equations I: Nonstiff problems, *SIAM Review* 32 (1990) 485–486. DOI: 10.1016/0378-4754(87)90083-8, arXiv: arXiv:1011.1669v3.



- [19] S. Van Der Walt, S. C. Colbert, and G. Varoquaux, The NumPy array: A structure for efficient numerical computation, *Computing in Science and Engineering* 13 (2011) 22–30. DOI: 10.1109/MCSE.2011.37, arXiv: 1102.1523.
- [20] K. E. Niemeyer and N. J. Curtis, pyJac v1.0.2, 2017, DOI: 10.5281/zenodo.251144.
- [21] K. E. Niemeyer, N. J. Curtis, and C.-J. Sung, pyJac: analytical Jacobian generator for chemical kinetics, *Computer Physics Communications* (2017), DOI: 10.1016/j.cpc.2017.02.004.
- [22] D. G. Goodwin, H. K. Moffat, and R. L. Speth, Cantera: An Object-oriented Software Toolkit for Chemical <http://www.cantera.org>, Version 2.3.0, 2017, DOI: 10.5281/zenodo.170284.
- [23] R. J. LeVeque, *Finite Difference Methods for Ordinary and Partial Differential Equations*, 2007, pp. 1–356.
- [24] L. F. Shampine, Type-insensitive ODE codes based on implicit A-stable formulas, *Mathematics of Computation* 39 (1982) 109–123. DOI: 10.1090/S0025-5718-1982-0658216-2.
- [25] C. S. Yoo, R. Sankaran, and J. H. Chen, Three-dimensional direct numerical simulation of a turbulent lifted hydrogen jet flame in heated coflow: flame stabilization and structure, *Journal of Fluid Mechanics* 640 (2009) 453–481.