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[1] *and Kyle E. Niemeyer*[1,*]

[1]*School of Mechanical, Industrial, and Manufacturing Engineering, Oregon State University, Corvallis, OR 97331, USA*
[*]*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 time required to integrate using implicit and explicit methods. We will conclude by analyzing 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_2 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 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 (H_2/CO) model of Burke et al. [13] and GRI Mech 3.0 [14], which uses CH_4 as fuel. The H_2/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 13×13 and 53×53, respectively. These relatively small Jacobian matrices allow 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 autoignition simulations and calculate 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 to compare explicit to implicit methods. The first integrator was `vode`, which uses an implicit Adams method for non-stiff problems and a method based on the backwards differentiation formulae (BDF) for stiff problems [17]. The explicit integrator used was `dopri5`, which uses a Runge-Kutta method of order (4)5 developed by Dormand and Prince with automatic step size control [18]. The `vode` integrator was set to use the default behavior of using first-order finite differencing to provide an estimate of the Jacobian matrix. Both integrators use internal time-steps in between the user defined time-steps for accuracy control and make use of `FORTRAN` libraries to perform the integrations. `NumPy` [19] was also extensively used in the code to determine eigenvalues and perform numerous basic mathematical functions. The integrator used default values of 10^{-6} and 10^{-12} for the error control parameters `rtol` and `atol`, respectively.

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.

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 particle of the PaSR model, the evaluation could be simplified to a zero-dimensional analysis. The H_2/CO simulation used a different set of PaSR data than the GRI Mech 3.0 simulation. Both sets of 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.

The wall clock time required by `vode` and `dopri5` to advance one user specified time step was measured for each particle of the PaSR simulation. The number of RHS function calls used by the integrator to advance one time-step was measured by adding a global variable, and the number of internal time-steps used by `vode` was measured. Measurement of the time-steps used by `vode` was accomplished without manipulation of the original `FORTRAN` code by limiting the integrator to one time step each time the integrator was called. `dopri5` is explicit and uses six function calls to advance each internal time step, therefore the number of internal time-steps for this integrator was calculated by dividing the number of RHS function calls by six.

The four stiffness quantification methods discussed within this paper are the stiffness ratio, the stiffness index, the stiffness indicator, and the chemical explosive mode. Variable names have been changed in the equations below to maintain consistency throughout this paper.

2.1 Stiffness Ratio

One commonly 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

$$\text{Stiffness Ratio} = \frac{\max|\lambda_p|}{\min|\lambda_p|} \tag{1}$$

in which λ_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

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

or alternatively,

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

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, τ represents the specified tolerance, $\rho[A]$ represents the spectral radius of the Jacobian matrix A, γ is a constant characteristic of the formula, and ξ 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

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

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

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

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). A large value of the stiffness index at a given point is an indication that the system of ODEs is locally stiff at that point.

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 the vector defined as

$$\Phi = \{T, Y_1, Y_2, \ldots, Y_{N_{sp}}\}^\mathsf{T} , \tag{6}$$

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

$$\frac{\partial \Phi}{\partial t} = \left\{ \frac{\partial Y_1}{\partial t}, \frac{\partial Y_2}{\partial t}, \ldots, \frac{\partial Y_{N_{sp}}}{\partial t} \right\}^\mathsf{T} , \tag{7}$$

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 user defined 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 for all computations performed in this paper.

2.3 Stiffness Indicator

The "stiffness indicator" of a system of ODEs was proposed by Söderlind et. al. as a mathematically rigorous approach to characterize stiffness that works independent of the integration method used or operational criteria [8]. Söderlind notes that "large" negative values of the stiffness indicator are a necessary condition for stiffness. Söderlind further quantifies how "large" the negative values must be for an equation to be considered locally stiff for a given problem, however we are primarily interested in the values of the stiffness indicator itself to facilitate comparison between different simulations using different models and different timescales.

In the chemical kinetic models of interest, the stiffness indicator is calculated for a given Jacobian matrix $A \in \mathbb{R}^{n\times n}$ as:

$$\text{Stiffness Indicator} = \frac{m[A] + M[A]}{2} \tag{8}$$

where

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

with $\lambda\,[A]$ denoting the eigenvalues of matrix A and $\text{He}(A)$ denoting the Hermitian part of the matrix A, which is defined as

$$\text{He}(A) = \frac{A + A^{\intercal}}{2}. \tag{10}$$

While determination of the stiffness indicator does require calculation of eigenvalues following a transpose and addition of the Jacobian matrix, it is readily implemented and unlike the stiffness index it does not require storage of prior values of the solution.

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]. Though this metric was not originally intended as a device to quantify the numerical stiffness of a system of ODEs, we are interested in determining if correlations exist between identification of the flame structure and the computational work required to advance the simulation for either explicit or implicit methods. The chemical explosive modes are associated with positive eigenvalues of the Jacobian of the system of ODEs:

$$\text{Re}(\lambda) > 0. \tag{11}$$

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

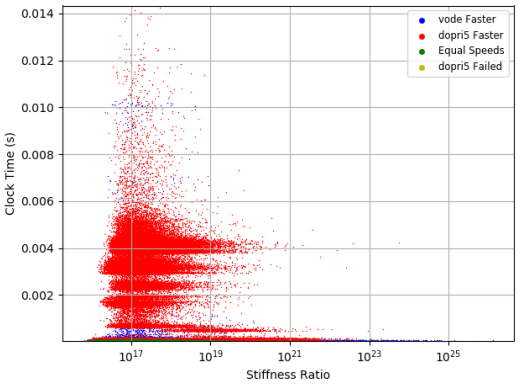
3. Results and Discussion

An analysis of each stiffness metric 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. Because a fourth order central difference formula was used to numerically approximate the second derivatives in the calculation of the stiffness index, the simulation was advanced until five solution values existed at each particle. The only information needed to calculate the stiffness ratio, stiffness indicator, and chemical explosive modes is the Jacobian matrix, therefore these values could be calculated at every time-step as soon as the solution was known. Constant step sizes of 1×10^{-7} s and 1×10^{-8} s were used for computations of the H_2/CO model and GRI Mech 3.0 models, respectively, however the built in methods of the `ode` package have automatic step size control to ensure that values remain within tolerance, and the integrator may take many steps in between the specified time-step. 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. As previously discussed, the wall clock time, number of RHS function calls, and internal time-steps were measured once at each particle, however we are most interested in wall clock time for the purposes of this study. The simulation was performed on an early 2015 MacBook Pro with a 2.9 GHz Intel Core i5 processor.

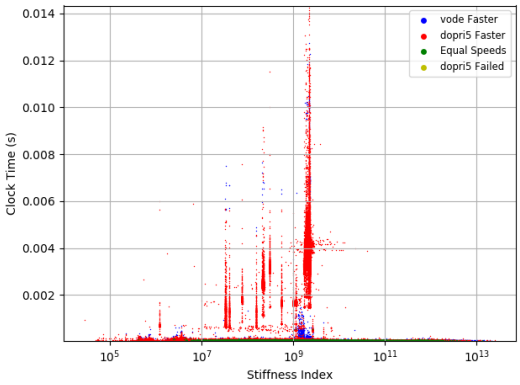
3.1 Results

The H_2/CO model was examined first due to its relative simplicity. The PaSR data for the this model consisted of 900900 different particles. It was found that `vode` had a smaller clock time than `dopri5` for 729384 particles or 81.0% of the PaSR data for this model, while `dopri5` had a smaller clock time for 165001 particles or 18.3% of the PaSR data. `vode` and `dopri5` had the same clock time within the numerical tolerance of the Python code for 6515 particles, and `dopri5` did not fail when integrating at any particle. In general, `vode` was likely to outperform `dopri5` when pressure was 1 atm, and also when pressure was 10 atm at 400K. At higher pressures and temperatures, the explicit `dopri5` was more likely to outperform the implicit `vode` when using this model. This may be because the H_2/CO model is generally more limited by accuracy requirements as opposed to stability requirements. `vode` uses backward differentiation formulae, which require multiple data points for higher order approximations. At the boundary conditions where few or no adjacent data points exist, lower order approximations are used. The use of lower order methods may be advantageous when few internal time-steps are needed because fewer RHS function calls are required to provide an approximation. Higher order methods may be advantageous when an intermediate number of time-steps are needed to provide the required accuracy because higher order methods are often able to make use of larger time-steps. Additionally, unlike implicit methods, explicit methods do not require calculation of Jacobian matrices to advance time.

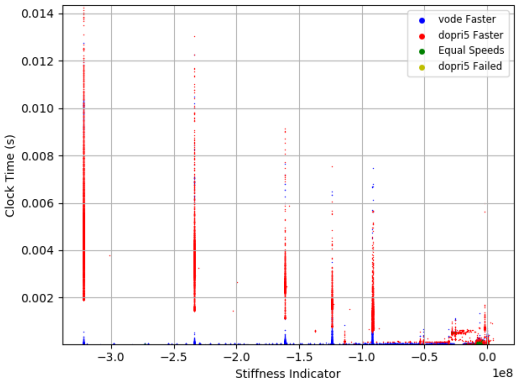
As seen in Figure 1a, the stiffness ratio was not highly correlated with wall clock time required for either `vode` or `dopri5`, nor did it provide a particularly efficient indicator of the relative performance of either method. `dopri5` appears to outperform `vode` at values of the stiffness ratio greater than 10^{19}, however neither method had a clear advantage when values of the stiffness ratio were less than this. The stiffness index did not appear to perform any better than the stiffness ratio in the current set of data from this simulation, as seen in Figure 1b. Neither method had a clear advantage



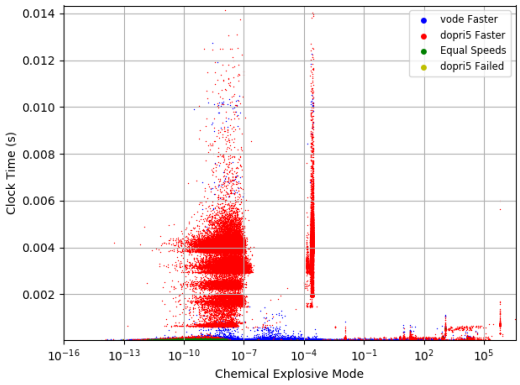
(a) Stiffness ratio versus wall clock time used to integrate one time-step at each particle of the PaSR model.



(b) Stiffness index versus wall clock time used to integrate one time-step at each particle of the PaSR model.



(c) Stiffness indicator versus wall clock time used to integrate one time-step at each particle of the PaSR model.



(d) Chemical explosive mode versus wall clock time used to integrate one time-step at each particle of the PaSR model.

Figure 1: Stiffness metrics versus RHS function calls used to advance one time-step of the H_2/CO model using either `vode` or `dopri5`. Calculated with a user defined constant step size of 10^{-7} s for every initial condition represented in the PaSR model.

at any range of the stiffness index, nor was the stiffness index a good indicator of wall clock time. As is the case in all of the plots obtained from this simulation, many of the integrations used very little clock time, making it difficult to draw correlations. Better correlations may become apparent if the user defined step size were increased to a value larger than 1×10^{-7} s. Similar to the stiffness ratio and the stiffness index, the chemical explosive mode was not highly correlated with the wall clock time required to integrate one step. When the chemical explosive mode values were greater than 10^2, `dopri5` appears to generally outperform `vode`.

The stiffness indicator had a better correlation with the wall clock time and the number of function calls used by either `vode` or `dopri5` than the other stiffness metrics, with a larger negative stiffness indicator correlating to a longer wall clock time as seen in Figure 1c. Further investigation will determine if this trend becomes more apparent when the step size is increased to a value larger than 1×10^{-7} s. There did not appear to be a clear correlation between the value of the stiffness indicator and the relative performance of either method in the current set of data.

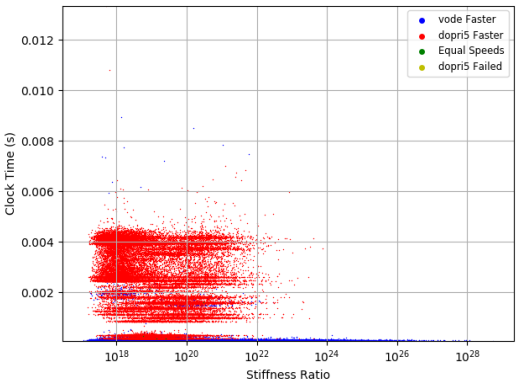
We looked to the GRI Mech 3.0 model next to determine how the stiffness metrics are impacted by using a stiffer model than the H_2/CO model. The PaSR data for GRI Mech 3.0 consisted of 450900 particles. It was found that `vode` had a smaller clock time than `dopri5` for 408232 particles or 90.5% of the PaSR data for this model, while `dopri5` had a smaller clock time for 165001 particles or 9.4% of the PaSR data. `vode` and `dopri5` had the same clock time within the numerical tolerance of the Python code for only six particles, and `dopri5` did not fail when integrating at any particle. The conditions in which `vode` was likely to outperform `dopri5` when using the GRI Mech 3.0 model ranged from low pressure and low temperature to high temperature and high pressure initial conditions of the PaSR data. Superior performance of `vode` in the high pressure and high temperature conditions is a potential indication that the internal step size of the integrator is limited by stability rather than accuracy requirements.

As seen in Figure 2a, the stiffness ratio was not correlated with the wall clock time required to integrate one time step, however `vode` appeared to generally outperform `dopri5` at values of the stiffness ratio greater than 10^{22}. At values of the stiffness ratio less than 10^{22}, it is not clear which method has better performance. The stiffness index behaved similarly to the stiffness ratio, as seen in Figure 2b While the stiffness ratio was not correlated with the wall clock time required to integrate one time step, `vode` appeared to outperform `dopri5` at index values greater than approximately 5×10^{10}.

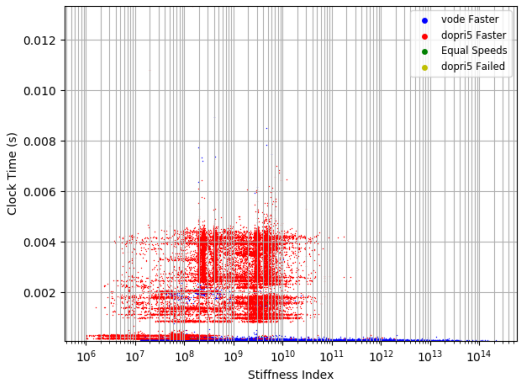
The usefulness of the stiffness indicator in predicting the wall clock time per time step was more clear in the set of data from the GRI Mech 3.0 model, as seen in Figure 2c. Larger negative values of the stiffness indicator correspond to a longer wall clock time per integration for the `dopri5` integrator. Further investigation will determine if additional trends may be made clear for the `vode` integrator if the time step were increased to a value larger than 10^{-8} s. `vode` generally outperformed `dopri5` at values of the stiffness indicator value less than -2×10^9 in the current set of data. Additional investigation will determine if the stiffness indicator can reliably be used to predict relative performance of the two methods.

The usefulness chemical explosive mode in the GRI Mech 3.0 simulation was similar to the usefulness of the chemical explosive mode in the H_2/CO simulation. While the chemical explosive mode was not useful in predicting values of the wall clock time used to integrate one time step, `dopri5` outperformed `vode` when values of the chemical explosive mode were greater than 1.5×10^2.

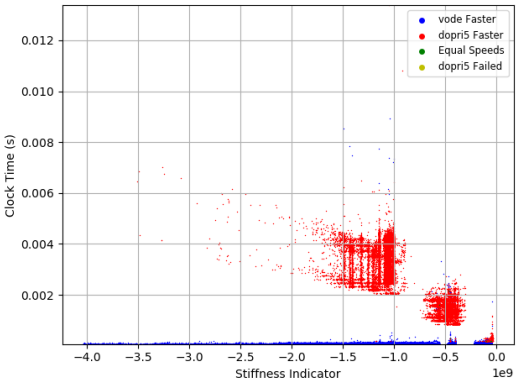
We are interested in determining the potential speedup offered by an integrator using these



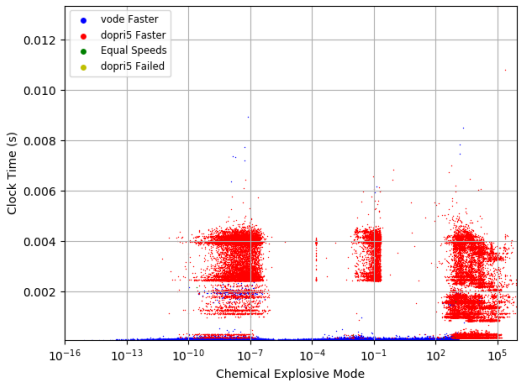
(a) Stiffness ratio versus the wall clock time used to integrate one time-step at each particle of the PaSR model.



(b) Stiffness index versus the wall clock time used to integrate one time-step at each particle of the PaSR model.



(c) Stiffness indicator versus the wall clock time used to integrate one time-step at each particle of the PaSR model.



(d) Chemical explosive mode versus the wall clock time used to integrate one time-step at each particle of the PaSR model.

Figure 2: Stiffness metrics versus RHS function calls used to advance one time-step of the GRI Mech 3.0 model using either `vode` or `dopri5`. Calculated with a constant step size of 10^{-8} s for every initial condition represented in the PaSR model.

metrics. Because implicit methods such as `vode` are used by default for most simulations of combustion, we are most interested in methods that predict when explicit methods such as `dopri5` have superior performance to explicit methods. Using the data obtained from the simulations above, we are able to select integration times from either integrator based on stiffness metric values at those points. For the set of data obtained from the GRI Mech 3.0 simulation, we selected the timing data from `vode` from particles that had a chemical explosive mode value of less than 1.5×10^2, and timing data from `dopri5` for all particles with a chemical explosive mode value of greater than 1.5×10^2. The total clock time `vode` needed to integrate one time step at each particle of the PaSR simulation was 287.3 s, while the "scheduler" had a total wall clock time of 238.5 s to complete all integrations, a speedup of approximately $1.2\times$ compared to `vode`. For comparison, a perfect scheduler that selects the faster method at every particle would have a total integration time of 136.1 s, a speedup of approximately $2.1\times$ compared to `vode`.

4. Conclusions

Four stiffness quantification metrics were investigated using partially stirred reactor simulations for two chemical kinetic models. These metrics were the stiffness ratio, stiffness index, stiffness indicator, and the chemical explosive mode. Our analysis demonstrated that the four stiffness quantification metrics investigated have the potential to be used by a scheduler that automatically selects the most efficient integration method at each point. As an example, it was found that the wall clock time needed to integrate 450900 time steps of 10^{-8} s in the GRI Mech 3.0 model could have been reduced by a factor of $1.2\times$ if a scheduler had selected an explicit method when the chemical explosive mode exceeded a value of 1.5×10^2. Further analysis is needed to determine which stiffness quantification metric may be used to provide the largest speedup, as well as the sensitivity of these metrics to the user defined step size. Additionally, the cost of determining the value of each stiffness quantification metric will be considered when implementing a scheduler in further research.

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.

[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, 2007.

[5] National Research Council, Transforming Combustion Research through Cyberinfrastructure, Available at 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, 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 H2/O2 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/, 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.