FINAL REPORT

December, 1999

Sloan - US. Department of Energy Joint Postdoctoral Fellowship in Computational Molecular Biology

Project Title: Canonical Nonlinear Methods for Modeling and Analyzing Gene Circuits and Spatial Variations during Pattern Formation in Embryonic Development

Investigators: Grenmarie Agresar and Michael Savageau

Prepared for: The U.S. Department of Energy

Award No: DE-FG02-97ER62436

The modeling and analysis of the complex interactions between genes and metabolites during development require computational approaches. However, existing methods cannot efficiently account for the large number of interacting players, the nonlinear nature of the interactions or the disparate scales involved. The latter represents a challenge in modeling developmental systems since reaction rates and diffusion times can vary by several orders of magnitude (depending on the molecular system). Modeling processes of this type results in the pathology of stiffness. Numerically, stiffness occurs when in order to prevent large amplification of errors typical (non-stiff) algorithms require a step size much smaller than the scale at which the solution is changing. This causes excessive computational effort and likely results in unacceptable accumulation of errors. Stiff methods circumvent these difficulties at the expense of complexity, increased storage requirements, and additional operations per step. The most common of these have been used successfully to solve small systems of equations, but their utility is limited for larger systems. On the other hand, the demands on the size of developmental models are ever increasing as the complexity of the underlying biology and the availability of data increases. Therefore, the need for more economical stiff methods prevails and the research for optimal methods is still ongoing.

In this project we developed a new method to solve large stiff systems of equations in the non-linear power law form. The power-law formalism is a proven powerful tool for biological systems modeling [5], and has many advantages over other formalisms used for this purpose. The advantages include the fact that it is canonical (arbitrary functions can be exactly recasted into this form [4]) and that it is an accurate local approximation to any type of interaction (useful if the mathematical form is not known). In addition, a very efficient and robust non-stiff method has been developed to solve differential equations in this form [3]. This method uses truncated Taylor series to approximate the solution, and adaptively optimizes the order of the expansion and the step size.

DOE Patent Clearance Granted

MP Dvorscak

Mark P. Dvorscak

(630) 252-2393

E-mail: mark.dvorscak@ch.doe.gov

Office of Intellectual Property Law

DOE Chicago Operations Office

May 7, 2001
Date

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

By analyzing the behavior of this non-stiff method in the presence of stiffness we have developed a stiff method that treats stiffness only when it is actually present. Each time step is attempted with the relatively inexpensive non-stiff method and the solution is “probed” for the presence of stiffness. An inexpensive stiffness-detection test has been developed based on elements already computed by the non-stiff method. When stiffness is detected, the more expensive stiff method is invoked. The stiff algorithm is based on the assumption that the solution can be partitioned locally (at each time step) into two parts: 1) a stiff portion defined by the components with high error amplification power but that contribute little to the solution during the particular time step; and 2) a non-stiff portion defined by the remaining components of the solution. The stiff components are isolated using the Lanczos algorithm with the Jacobian of the system and an initial vector pre-processed by the Power Method to accentuate the stiff directions. A stiff component is accepted if the amplification power of the Jacobian is larger than the local change in the solution (measured as a norm of the first derivative). The vectors produced by the Lanczos algorithm are used to remove the stiff components from the Taylor series expansion. The size of the step for advancing the solution is determined by the residual of this non-stiff projection of the Taylor series, and should correspond to the time-scale at which the solution is changing. The solution is then corrected using a Trapezoidal method in the stiff space. This correction is only necessary if the local stiff components contribute more than the specified error tolerance to the solution during the step.

The method has been tested with a set of original problems as well as with benchmark problems of the literature [1, 2]. The test cases are of small dimension to maintain the analysis tractable. However, they were carefully chosen to challenge our method with many different stiff situations, and the challenging elements of these situations will be the same in larger problems. The chosen test cases include problems with rapidly changing Jacobians (where both the magnitude of the eigenvalues and the number and identity of the stiff components change rapidly as a function of time), repeated eigenvalues, complex eigenvalues, and eigenvalues of very large magnitude. We now present representative results from these tests.

Figure 1 shows results from an original test case in which the Jacobian norm (representing the amplification power of the system) changes rapidly as a function of the independent variable. The behavior of the eigenvalues of the Jacobian is shown in the middle frame of this figure. The top frame shows the solutions to the problem and the bottom frame shows the number of stiff components used by the stiff method (a value of 0 means that the non-stiff method was used for this step). Comparing the frames shows how the solutions change relatively slowly in the regions where the Jacobian norm is the largest. These regions are properly identified as stiff by our method. In addition, the method is able to find the appropriate number of stiff components (corresponding to the number of eigenvalues with large magnitude) throughout the computation. These components represent local, linear combinations of the variables that are almost invariant during the step. Therefore, this feature of the method has potential to be exploited as an analytical tool in the modeling of biological systems.

FIG. 1: Top: Solution to problem NLH1 in logarithmic coordinates. Middle: "Stiff" eigenvalues as a function of time. Bottom: Number of stiff components as a function of time (a value of 0 indicates that the non-stiff method was used for that step)

Figure 2 shows results from a famous test case from the literature: the Oregonator [2]. The top frame shows the behavior of the solutions in logarithmic coordinates and the bottom frame shows the behavior of the method. The norm of the Jacobian remains relatively large throughout the interval with only one stiff component. An interesting feature of this problem is that it has two regions where, although the Jacobian norm is large, the solution changes rapidly, and our algorithm correctly uses the cheaper non-stiff method to compute the solution. Only one stiff component is found as expected.

We successfully solved all of the test cases tried using our method. We varied the error tolerance in integral exponents from 10^{-2} to 10^{-12} , and Tables 1-3 show the run characteristics for three representative tolerances (10^{-2} , 10^{-6} and 10^{-10}). These tables show that the method solves all problems in fewer steps, using significantly less CPU time and with equal accuracy than the non-stiff method. The results also show that the error control in the method work well since the computed error is close to the specified error tolerance. The number of stiff failures is minimal in most cases showing the consistency of the stiffness detection test and the criteria for accepting stiff components. The average number of stiff components found by the stiff method is equal to the average Krylov size in the tables. We carefully examined the eigenvalues of the Jacobians, and found these numbers to be appropriate in all cases.

In comparison to other stiff method, our method has several advantages. First, the Jacobian of the system can be computed analytically and economically due to the regular structure of the power-law equations. Second, the algorithm only involves vector multiplications, minimizing storage requirements since large matrices are not stored. Many of the operations in the method can be performed at the same time by processors working in parallel, and we expect a large increase in efficiency after such parallelization. Third, automatic switching between the non-stiff method and the stiff version accommodates the changes in the degree and sources of stiffness exhibited by non-linear problems, and ensures that the more expensive stiff method is used only when appropriate. In addition, the method provides information that can be exploited to analyze models of genetic circuits in development. Mathematical models are typically analyzed by examining the Jacobian of the system. However, since in non-linear problems the Jacobian changes with time, this analysis is difficult to do and is typically only done at the steady state. With our method, however, the Lanczos' tridiagonal matrix and vectors generated at each stiff step can be used to provide some of this information. For example, in the stiff regions, only the particular combinations of interactions that form the non-stiff components are important in determining the behavior of the system.

We have successfully accomplished the first task of our proposed project: to develop an optimal stiff algorithm to solve equations expressed in the canonical power-law form. This is a significant step towards the development of efficient, accurate, and robust computational methods for analyzing gene circuits involved in biological pattern formation. In addition, our method includes the unanticipated benefit of providing additional tools for analyzing biological models. We hope to continue these developments, and that other researchers can take advantage of these tools.

FIG 2. Solution in logarithmic coordinates (top) and number of stiff components (bottom) for the test case OREGO

	Stiff Method							Non-Stiff Method			
	$L_{\max}$	L_{ave}	# S	# NS	Fail	CPU	Kry	$L_{\max}$	L_{ave}	# steps	CPU
NLH1	1.83	2.247	48	88	0	0.03	1.333	2.086	2.305	2648	0.84
NLH2	1.61	1.93	26	71	0	0.03	3.885	2.107	2.308	19879	2.54
A2	2.21	2.471	267	4	0	0.1	1	2.299	2.299	120631	22.96
B5	1.9	2.169	1	31	0	0	2	1.804	2.12	732	0.07
C5	2.12	2.513	16	97	18	0.01	2.25	2.447	2.708	983	0.03
D2	2.12	2.129	43	231	0	0.01	1	2.343	2.627	57486	1.11
D3	1.54	1.548	27	213	0	0.02	1	1.508	1.528	3031	0.1
D4	2.69	3.031	53	0	0	0.01	1	0.06	0.757	265415	5.31
D5	1.65	2.185	18	0	0	0	1	1.537	2.22	26620	0.92
D6	2.14	2.146	92	127	0	0.02	1	3.728	3.909	19597664	527.7
E3	1.93	2.185	34	3	0	0	1	2.385	2.56	18890	0.41
E5	4.17	4.222	135	171	0	0.03	1	3.938	4.163	10116452	280.5
HIRES	2.65	3.417	314	355	1	0.14	1.847	2.622	3.185	18012	1.1
OREGO	1.23	1.806	492	313	0	0.06	1	2.306	2.471	5605417	125.3

TABLE 1: Run information for error tolerance = 10^{-2} . $L_{\max}$ and L_{ave} are the logarithm of the maximum error and the average error of the computed solution, respectively. # S, # NS and Fails are the number of steps computed with the stiff method (excluding failed steps), the number computed with the non-stiff method (including failed stiff steps) and the number failures of the stiff method, respectively. Finally, CPU is the CPU time used in seconds, Kry is the average Krylov size used by the stiff method, and # steps is the number of steps that the non-stiff method requires to complete the run without invoking the stiff method.

	Stiff Method							Non-Stiff Method			
	L _{max}	L _{ave}	# S	# NS	Fails	CPU	Kry	L _{max}	L _{ave}	# steps	CPU
NLH1	5.48	6.364	97	41	0	0.06	1.175	6.197	6.427	2016	0.74
NLH2	5.02	6.184	158	88	5	0.11	3.544	6.327	6.478	15900	2.53
A2	6.13	6.915	782	10	0	0.33	1	6.294	6.334	112644	22.34
B5	5.88	5.963	1	126	0	0.02	2	5.876	6.08	787	0.09
C5	6.25	6.269	28	96	9	0.01	2.833	6.426	6.85	855	0.03
D2	6.08	6.094	279	368	0	0.07	1	6.162	6.444	45736	1.4
D3	5.39	5.41	66	398	0	0.06	1	5.436	5.444	3024	0.15
D4	6.41	6.649	338	1	0	0.03	1	6.492	6.649	75985	2.32
D5	5.39	6.244	494	27	0	0.07	1	6.383	6.474	19700	1
D6	6.11	6.613	823	287	0	0.13	1	10.56	11.03	19337108	530.5
E3	5.55	5.8	435	14	0	0.04	1	6.53	6.927	15075	0.5
E5	6.09	6.087	804	189	0	0.12	1	8.351	9.212	9725584	278.9
HIRES	5.21	5.236	476	418	1	0.29	1.626	5.162	5.405	14247	1.33
OREGO	4.64	5.378	1356	362	2	0.16	1	6.287	6.887	4483748	152.3

TABLE 2: Run information for error tolerance = $1e^{-6}$. Definitions are the same as in Table 1.

	Stiff Method							Non-Stiff Method			
	L _{max}	L _{ave}	# S	# NS	Fails	CPU	Kry	L _{max}	L _{ave}	# steps	CPU
NLH1	8.21	10.17	114	244	6	0.15	1.202	10.28	10.47	1669	0.71
NLH2	8.48	10.15	230	115	8	0.16	3.67	10.33	10.48	14275	2.81
A2	9.27	10.55	1521	525	0	0.88	1	10.35	10.42	92803	20.92
B5	9.86	9.898	2	155	0	0.03	2	9.83	10.1	820	0.11
C5	10.2	10.3	31	230	8	0.04	2.276	10.6	10.85	652	0.04
D2	9.35	9.721	844	1450	208	0.3	1	10.16	10.53	41631	1.79
D3	9.29	9.3	97	1123	23	0.17	1	9.365	9.371	3233	0.25
D4	10.8	10.9	631	3	0	0.06	1	10.56	10.76	75983	2.32
D5	9.19	9.419	914	80	32	0.18	1	10.41	10.73	17764	1.22
D6	9.68	9.872	6934	737	0	0.99	1	10.2	10.48	15547771	622.7
E3	8.64	8.815	1621	418	132	0.22	1	9.745	10.27	13493	0.63
E5	9.1	9.104	3691	532	22	0.57	1	9.403	10.03	7847110	327.8
HIRES	7.27	7.639	832	1286	152	0.74	1.022	9.462	9.687	13015	1.66
OREGO	8.66	9.39	6710	1967	198	0.96	1	9.492	10.37	4474435	152

TABLE 3: Run information for error tolerance = $1e^{-10}$. Definitions are the same as in Table 1.

REFERENCES

- [1] ENRIGHT, W.H., AND J.D. PRYCE, *Two FORTRAN Packages for Assessing Initial Value Methods*, ACM Trans. on Math. Software 13(1), 1-27.
- [2] HAIRER, E. AND WANNER, G. "Solving Differential Equations II: Stiff and Differential-Algebraic Problems", second edition, Springer, Berlin (1997)
- [3] IRVINE, D.H., AND M.A. SAVAGEAU, *Efficient solution of nonlinear ordinary differential equations expressed in S-system canonical form*, SIAM J. Numerical Anal. 27, 704-735 (1990).
- [4] SAVAGEAU, M.A., AND E.O. VOIT, *Recasting nonlinear differential equations as S-systems: a canonical nonlinear form*, Math. Biosci. 87, 83-115 (1987).
- [5] SAVAGEAU, M.A., *Power-law formalism: A canonical nonlinear approach to modeling and analysis*, in "World Congress of Nonlinear Analysts 92," Vol. 4, pp. 3323-3334, V. Lakshmikantham, Ed., Walter de Gruyter Publishers, Berlin (1996).