Optimization Methods and Software
Vol. 00, No. 00, Month 200x, 1–10

RESEARCH ARTICLE

Sparse Interpolatory Reduced-Order Models for Simulation of Light-Induced Molecular Transformations

David Mokrauer and C T. Kelley

(*Received 00 Month 200x; in final form 00 Month 200x*)

We describe a method for using interpolatory models to accurately and efficiently simulate molecular excitation and relaxation. We use sparse interpolation for efficiency and local error estimation and control for robustness and accuracy.

Keywords: Sparse Interpolation, Molecular Dynamics, Error Estimation and Control

AMS Subject Classification: 65K10; 65L05; 81V55

1. Introduction

The purpose of this paper is to describe an efficient algorithm for simulation of light-induced molecular transformations. The simulation seeks to follow the relaxation path of a molecule after excitation by light. The simulator is a predictive tool to see if light excitation and subsequent return to the unexcited or ground state will produce a different configuration than the initial one.

We simulate the results of the excitation, rather than the excitation itself. The excitation will change the quantum state of a molecule. Our objective is to design software that will let one explore the possible changes in a molecule after a sequence of excitations.

The goals of the simulation are not only to identify the end point, but to report the entire path in an high-dimensional configuration space so that one can look for nearby paths to interesting configurations and examine the energy landscape near the path to see if low energy barriers make jumping to a different path possible. We have used earlier versions of the algorithm from this paper in previous work [11–13] and our software has also been applied to sensors in [4].

Our approach uses the Smolyak sparse interpolation [19] method to build a surrogate model of an expensive molecular dynamics code (in our particular case Gaussian [7]), and uses that model to drive a numerical ODE integrator. In this paper we describe a new version of the algorithm which incorporates error estimation and control and response to the performance of Gaussian's internal optimization. We then apply the new algorithm to the application from [4] to illustrate the ideas and demonstrate the quality of the error estimate.

We begin in § 1.1 with a precise statement of the problem and the stages of model reduction we will need to make a solution computationally tractable. In § 2 we describe the details of the latest version of the method. Finally, in § 3 we give

Version of December 21, 2011. This work has been partially supported by National Science Foundation Grant CDI-0941253 and Army Research Office Grants W911NF-07-1-0112 and W911NF-11-1-0367.
North Carolina State University, Department of Mathematics, Box 8205, Raleigh, N. C. 27695-8205, USA (mokrauerd@gmail.com,Tim_Kelley@ncsu.edu)

ISSN: 1055-6788 print/ISSN 1029-4937 online
© 200x Taylor & Francis
DOI: 10.1080/03081080xxxxxxxxx
http://www.informaworld.com

two examples that illustrate the utility of the software and show how our error estimates compare with computational results.

1.1. *Problem Formulation*

The task is to develop a design tool which will simulate molecular changes after excitation. Ideally, we would do this by computing the ground state, simulating excitation with a quantum chemistry code (in our case Gaussian [7]), and then following the gradient descent path in molecular configuration space (bond lengths, valence angles, and torsion angles).

Molecules of interest can have hundreds of atoms and degrees of freedom. One cannot vary all of the degrees of freedom at once in a dynamic simulation because the simulator is too computationally costly. Hence, we must apply several layers of model reduction.

The first of these is to isolate a few molecular coordinates of particular interest and vary only those in the dynamic simulation. The simulator computes an energy $\mathcal{E}_\ell(p)$ as a function of a vector of configuration variables (torsion and bond angles) $p \in R^N$ and the quantum state $\ell = 0, 1, \dots$. We split

$$p = \begin{pmatrix} x \\ \xi \end{pmatrix}$$

into a low-dimensional vector of design variables x and the remainder ξ. Given ℓ, we compute the energy $E_\ell(x)$ as a function of x alone via

$$E_\ell(x) = \min_\xi \mathcal{E}_\ell \begin{pmatrix} x \\ \xi \end{pmatrix} \tag{1}$$

Gaussian approximates the solution of the optimization problem in (1) with a variant of the the BFGS [3, 5, 6, 8, 9, 17] algorithm.

The objective of this project is to simulate excitation of a molecule from one quantum state ℓ to another ℓ' and the subsequent relaxation to a local minimum of $E_{\ell'}$. The simulator will do this repeatedly for a sequence of states, beginning and ending at $\ell = 0$. We seek a sequence of states for which the initial configuration at the ground state ($\ell = 0$) is different from the one at the end of the sequence of excitations.

For each ℓ, we would ideally compute the local minimum with the gradient descent method, *i. e.* integrate the dynamics

$$\dot{x} = -\nabla E_\ell(x) \tag{2}$$

with initial data given either by the ground state configuration (for $\ell = 0$ at the start of the simulation) or by the result of excitation from a local minimum at another state. The evaluation of ∇E_ℓ is too expensive to drive the numerical solution of (2), so we apply a second level of model reduction and replace E_ℓ with a piecewise polynomial interpolant. This is sufficient, if done carefully, to solve simple problems with as many as two degrees of freedom (*i. e.* $\ell \in R^2$), and we report on results with a simple two-dimensional spline interpolation in [11, 12]. The problems with such an interpolation are that the number of nodes increases exponentially with the number of degrees of freedom (the size of x) and one must take care with the ordering of the evaluation of the nodes to make sure that the internal optimization has a sufficiently good initial iteration. The internal optimization in

Gaussian can fail if one does not pay attention to the latter problem.

In this paper we describe our most recent progress in addressing the two problems listed above. Sparse interpolation [19] will limit the growth in the number of nodes to a polynomial in the size of x, and we show how to couple sparse interpolation to an error estimation and control scheme which not only estimates the error in the dynamics, but also limits the failures in Gaussian's internal optimization.

1.2. *Example: 2-Butene C_4H_8*

In this section we give a simple example with only one degree of freedom. In this case it is possible to follow the relaxation after excitation by brute force, and this was done in [10].

This molecule has two stable geometries (cis and trans) for $\ell = 0$, and the transition path after excitation is well understood [10, 15, 18]. Figure 1 shows the two configurations. Note that they differ by a rotation of the torsion angle in the center of the molecule, hence a single degree of freedom will suffice to model the transition.

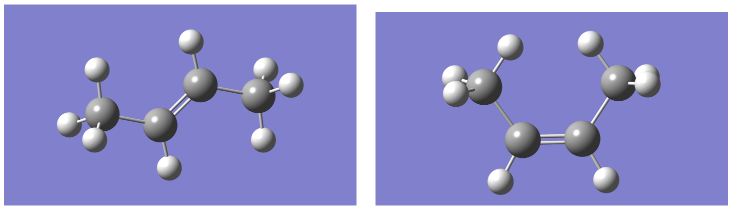


Figure 1. trans(a) and cis(b) 2-Butene

Figure 2 illustrates our independent duplication of the result from [10]. The arrows starting from the bottom left of the figure show excitation from the ground state (cis), followed by relaxation to a minimum in the excited state, then a drop to the ground state and relaxation to a different local minimum (trans).

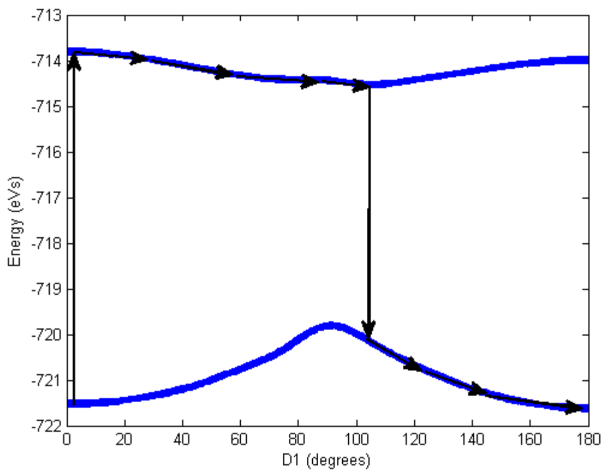


Figure 2. Simulation with one degree of freedom

2. Algorithms

Assume that $x \in R^d$. We will approximate E_ℓ on hypercubes in R^d, which we will refer to as "patches". At present the sides of the patches are parallel to coordinate axes in our software, but this is not necessary. We will let h denote the length of the sides of the patch, so the volume of the patch is h^d. We will approximate E_ℓ with a polynomial E_ℓ^k of degree of exactness k on each patch. Here, by degree of exactness we mean that the interpolation scheme will reproduce polynomials of degree k exactly, but may have degree higher than k.

We approximate the dynamics with a numerical integration on the patch of

$$\dot{x} = -\nabla E_\ell^k(x). \tag{3}$$

The simulations we report in § 3 all use $k = 2$ and $k = 3$. We used a standard Runge-Kutta 45 integration routine [1, 16] to integrate (3). The final step is to estimate the error of E on the patch and query the status of Gaussian's internal optimizer to determine the size of the new patch.

We begin the simulation at $\ell = 0$ with the ground state. The data for the simulation are

- The molecular data for Gaussian and the ground state.
- A sequence of quantum states $\{\ell_i\}_{i=1}^p$, where $\ell_1 = \ell_p = 0$.

The simulation loops over the quantum states, switching from one to the next when the integration stagnates at a local minimum. We create an initial patch and integrate (3) with a variable step size Runge-Kutta method [1, 16]. We detect local minima by terminating the integration when $\|\nabla E_\ell^k\|$ is sufficiently small and resolving the local minimum with a Newton iteration. On each patch we will either terminate on a local minima or the integration will collide with a patch boundary with one Runge-Kutta within the patch and the next outside. In the former case, we move to the next quantum state, in the latter we create a new patch located so that the last Runge-Kutta step in the old patch is on the "incoming" side of the new one. The size of the new patch depends on an estimate of the error in the old patch (see § 2.2) and the performance of Gaussian's internal optimization on the old patch.

2.1. *Sparse Interpolation*

On each patch we approximate the energy with a sparse interpolation [2, 14, 19, 20]. We will only sketch the formulae and refer the reader to the literature for the details. The method begins with interpolations in one dimension at the Chebyshev extrema

$$x_j^i = -\cos\left(\frac{\pi(j-1)}{m_i}\right), \; 1 \le j \le m_i,$$

where $m_i = 2^{i-1} + 1$. We let

$$\mathcal{U}^i(f)(x) = \sum_{j=1}^{m_i} f(x_j^i) l_j^i(x)$$

be the polynomial interpolant of f at the Chebyshev extrema. Here l_j^i are the usual Lagrange polynomials.

Now let $f : R^d \to R$. Using standard multi-index notation, let

$$\bar{\mathbf{i}} = (i_1, \ldots, i_d) \text{ and } |\bar{\mathbf{i}}| = \sum_{j=1}^{d} i_j.$$

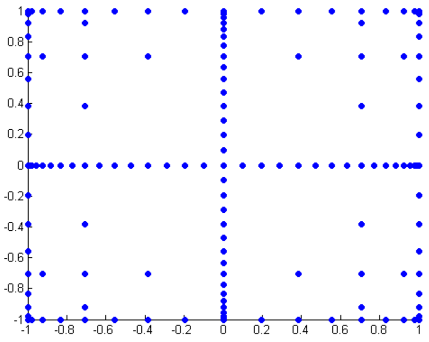
For $x = (x_1 \ldots x_d)^T$ define

$$\mathcal{U}^{\bar{\mathbf{i}}}(f)(x) = \sum_{j_1=1}^{m_{i_1}} \cdots \sum_{j_d=1}^{m_{i_d}} f(x^{i_1}_{j_1} \ldots x^{i_d}_{j_d})(l^{i_1}_{j_1}(x_1) \ldots l^{i_d}_{j_d}(x_d)). \tag{4}$$

The sparse interpolation is

$$\mathcal{A}(k, d, f) = \sum_{k+1 \le |\bar{\mathbf{i}}| \le d+k} (-1)^{d+k-|\bar{\mathbf{i}}|} \binom{d-1}{d+k-|\bar{\mathbf{i}}|} \mathcal{U}^{\bar{\mathbf{i}}}, \tag{5}$$

and the set of nodes Ω^k is implicitly defined by (4) and (5). Figure 3 are examples in $d = 2, 3$ for degree of exactness $k = 5$.


(a)

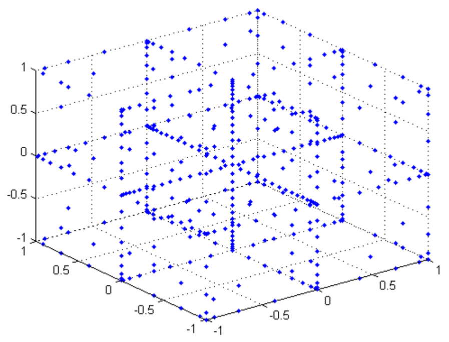

(b)

Figure 3. Sparse Grids for $k = 5$ in two and three dimensions

Our surrogates E^k are polynomial interpolations with degree of exactness k. We construct them by mapping the patch to $[-1, 1]^d$ and using (5).

The important points are

- The Smolyak grids are nested; $\Omega^k \subset \Omega^{k+1}$. So, one can estimate E^l for any $l < k$ with no additional work after the evaluation of E^k.
- E^k requires $\approx 2^k d^k / k!$ evaluations of E. This compares well with the best possible

$$\binom{d+k}{k} \approx d^k / k!$$

2.2. *Error Estimation and Control*

We control the size of the patch in two ways. The first is to keep the error below a tolerance τ.

For a patch of side length h, the error in E^k is $O(h^{k+1})$. Just as one would do in a variable step Runge Kutta approach, we estimate the error in E^k by

$$\epsilon_k = \|E^k - E\|_{RK}/\|E^k\|_{RK} \approx \|E^{k+1} - E^k\|_{RK}/\|E^k\|_{RK}$$

where the norm is the maximum over the Runge-Kutta steps in the integration. If the estimate is over τ, we shrink the current patch and recompute the path.

If the error in the current patch is acceptable, we then set

$$\epsilon_k = Ch^{k+1} = \|E^{k+1} - E^k\|_{RK}/\|E^k\|_{RK}$$

and solve for C to obtain the approximation

$$\tilde{C} = \frac{\|E^{k+1} - E^k\|_{RK}}{\|E^k\|_{RK} h^{k+1}}.$$

We estimate the error in the new patch by

$$\tilde{\epsilon} = \tilde{C} h^{k+1}$$

and set the patch size so that $\tilde{\epsilon} = X\tau$, so the new patch size is

$$h_{new} = .7 \left(\tau/\tilde{C}\right)^{1/(k+1)}.$$

Gaussian reports the number of internal optimization iterations needed to evaluate E at each of the interpolation nodes, and its internal optimizaiton terminates with failure after 20 iterations If the internal optimization fails, we must rebuild the patch with a small side length. This is very expensive and wastes the evaluations we did before the failure. To guard against failure we reduce h if the any the internal optimizations need more than 10 iterations. The safeguarding takes precedence over the error control, so we reduce the patch size if we are in danger of having the internal optimization fail, even if the error indicators would allow us to increase it.

3. Examples

All computations were performed on the high performance computing cluster at North Carolina State University. Our chassis has 60 quad core Xeon processors with 2GB distributed memory per core and dual gigabit ethernet interconnects. The operating system is Red Hat Linux version 2.6.18. Potential energy computations were performed using Gaussian 09. Script editing was done with Python 2.6.5. Interpolation and optimization was done with Matlab version 7.9.0.529.

In the examples we used $k = 2$ and $k = 3$ for the low and high order interpolations with $\tau = 10^{-3}$.

3.1. *Butene Revisited*

For two or more degrees of freedom plots like Figure 2 can be difficult to understand, so we illustrate the progress of the optimization with plots for each of the three degrees of freedom and the energy. The initial excitation dramatically increases the energy, after which the molecule relaxes in the new state. The computation required only five patches.

Figure 4 illustrates the design variables. These are the three angles $D1$, $D2$, and $D3$. $D1$, which was the single degree of freedom in the computation in § 1.2, is the center torsion angle $8-6-2-1$, which means one can rotate the molecule on the $6-2$ axis. Similarly $D2$ is the angle $12-8-6-2$ and $D3$ is $6-2-1-4$.

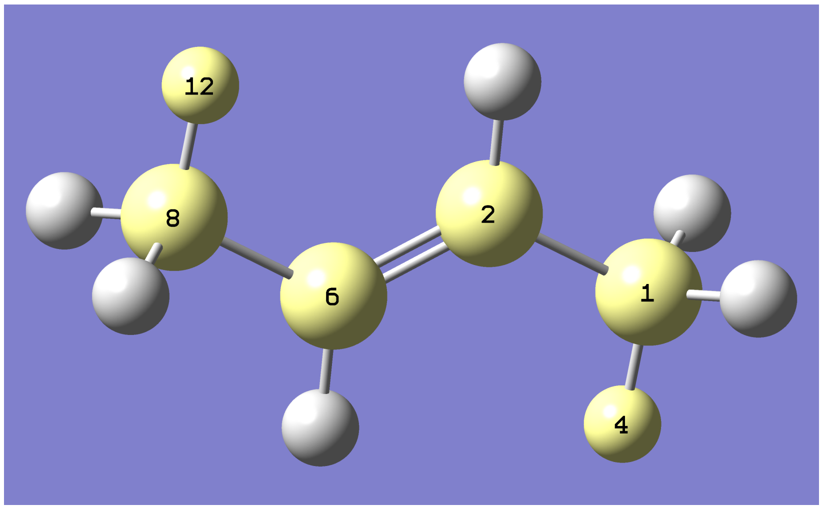


Figure 4. Three degrees of freedom

In Figure 5 we plot the changes in the three coordinates as functions of the Runge-Kutta step counter. From the figure one sees that D1 changes significantly after the simulation, which agrees with the results in § 1.2, and that D2 and D3 change little, if at all.

The vertical axis in the energy plots has a maximum energy of zero, which is an consequence of Gaussian's energy being in the interval $(-\inf, 0)$.

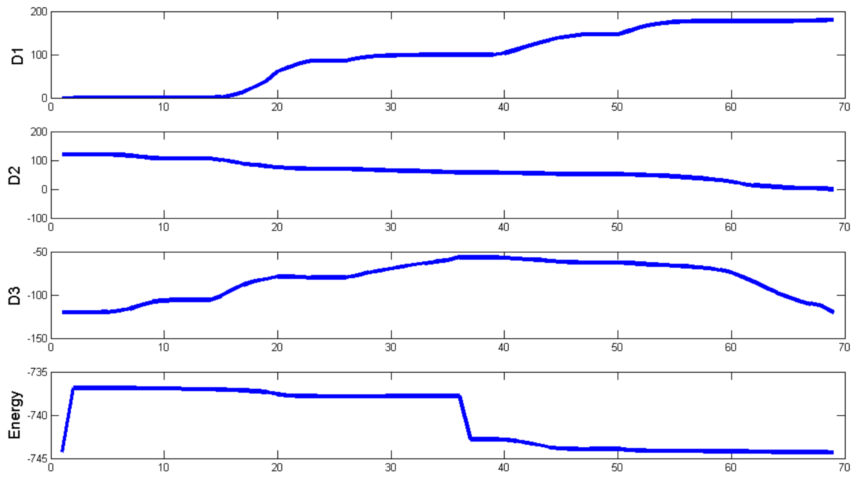


Figure 5. Simulation with three degrees of freedom

In Figure 6 we compare the value of $E(x)$ as computed with a call to Gaussian, with the solution of

$$\dot{x} = -\nabla E^3(x),$$

on each of the five patches, again as a function of the Runge-Kutta step on the patch. This figure is very expensive to create, as we have to do a Gaussian computation at every Runge-Kutta point and can only exploit parallel Gaussian calls in a limited way because there are only a few $(10 - 20)$ Runge-Kutta points in each patch. Contrast this with the parallel execution of Gaussian on the 69 patch nodes for $k = 3$.

The error estimate is quite good and the error control keeps the error well below the tolerance of 10^{-3}.

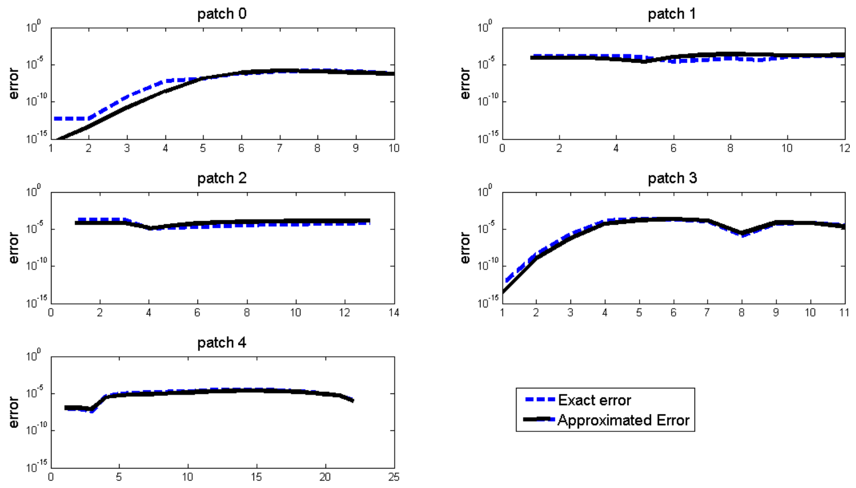


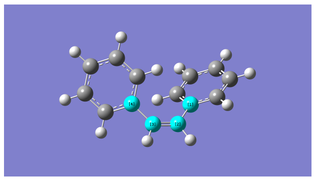
Figure 6. Testing the error indicator

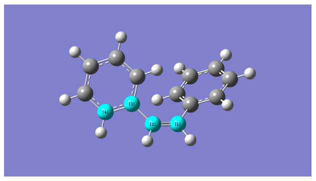
3.2. *Stilbene $C_{14}H_{12}$*

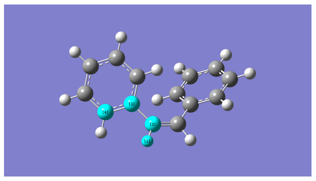
Our final example is stilbene $(C_{14}H_{12})$ a larger molecule. We include this final example to show how our approach can handle a molecule with more atoms in the context of a simulation with more degrees of freedom. The simulation for stilbene took over 95,000 seconds, more than 50 times the 1800 seconds we needed for butene.

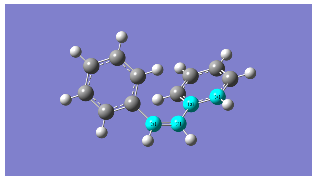
These results were first reported in [4]. The simulation has five degrees of freedom, and is so large that we could not make an error plot like the one in Figure 6 in a reasonable amount of time. In Figure 7 we illustrate the five angular degrees of freedom.

Figure 8 has plots of the changes in the degrees of freedom and the energy as a function of the Runge-Kutta step. The excitation at the start of the simulation is indicated by a vertical line from the ground state (marked with a $*$) to the initial data for the simulation. D5 changes significantly in the simulation, and Figure 9 illustrates that.


(a)


(b)


(c)


(d)

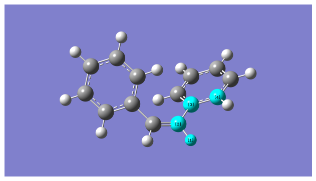

(e)

Figure 7. The five design angles D1(a), D2(b), D3(c), D4(d), D5(e)

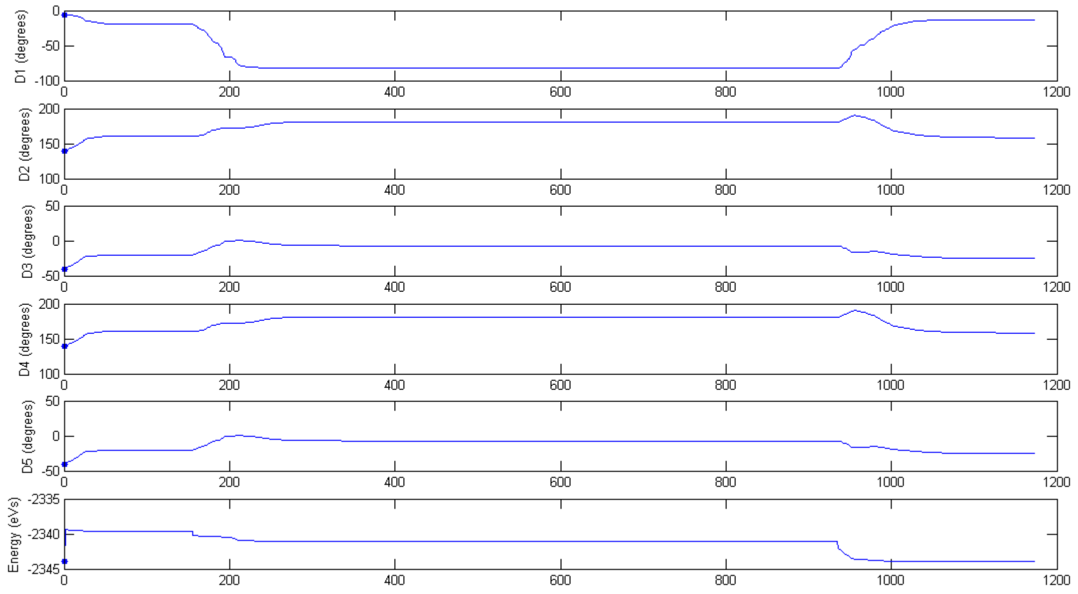
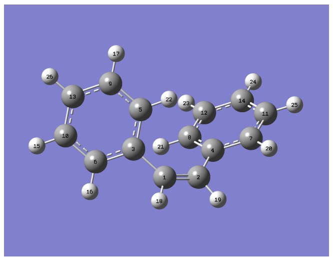


Figure 8. Stilbene simulation with five degrees of freedom


(a)

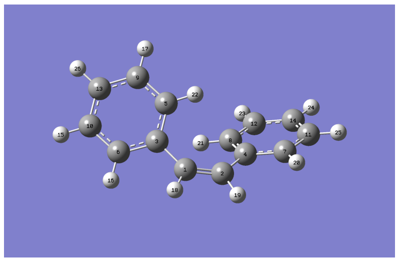

(b)

Figure 9. initial (a) and final (b) states

Acknowledgments

One of the authors (CTK) spoke about this work at the 8th International Conference on Numerical Optimization and Numerical Linear Algebra, held in Xiamen China in November of 2011. The author is very grateful to the organizers of that conference for their hospitality.

References

[1] U. M. Ascher and L. R. Petzold, *Computer Methods for Ordinary Differential Equations and Differential Algebraic Equations*, SIAM, Philadelphia, 1998.

[2] V. Barthelmann, E. Novak, and K. Ritter, *High dimensional polynomial interpolation on sparse grids*, Advances in Computational Mathematics, 12 (2000), pp. 273–288.

[3] C. G. Broyden, *A new double-rank minimization algorithm*, AMS Notices, 16 (1969), p. 670.

[4] A. Byhkovski and D. Woolard, *Physics and modeling of DNA-derivative architectures for long-wavelength bio-sensing*, 2011. to appear in Proceedings of CMOS Emerging Technologies 2011, Whistler, BC, Canada.

[5] J. E. Dennis and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, no. 16 in Classics in Applied Mathematics, SIAM, Philadelphia, 1996.

[6] R. Fletcher, *A new approach to variable metric methods*, Comput. J., 13 (1970), pp. 317–322.

[7] M. J. Frisch, A. Frisch, F. R. Clemente, and G. W. Trucks, *Gaussian 09 User's Reference*, Gaussian inc., Wallingford, CT, 2009.

[8] D. Goldfarb, *A family of variable metric methods derived by variational means*, Math. Comp., 24 (1970), pp. 23–26.

[9] C. T. Kelley, *Iterative Methods for Optimization*, no. 18 in Frontiers in Applied Mathematics, SIAM, Philadelphia, 1999.

[10] Y. Luo, B. Gelmont, and D. Woolard, *Bio-molecular devices for terahertz frequency sensing*, in Molecular and Nano Electronics: Analysis, Design, and Simulation, J. Seminario, ed., 2007, pp. 55–81.

[11] D. Mokrauer, C. T. Kelley, and A. Bykhovski, *Parallel computation of surrogate models for potential energy surfaces*, in 2010 International Symposium on Distributed Computing and Applications to Business, Engineering and Science, Q. Qingping and G. Yucheng, eds., Los Alamitos, CA, 2010, IEEE, pp. 1–4.

[12] ———, *Efficient parallel computation of molecular potential surfaces for the study of light-induced transition dynamics in multiple coordinates*, IEEE Transactions on Nanotechnology, 10 (2011), pp. 70–74.

[13] ———, *Simulations of light-induced molecular transformations in multiple dimensions with incremental sparse surrogates.* to appear in J. Algorithms and Computational Technology, 2011.

[14] E. Novak and K. Ritter, *Simple cubature formulas with high polynomial exactness*, Constr. Approx., 15 (1999), pp. 499–522.

[15] I. J. Palmer, I. N. Ragazos, F. Bemardi, M. Olivucci, and M. A. Robb, *An mc-scf study of the s1 and s2 photochemical reactions of benzene*, J. Am. Chem. Soc., (1993), pp. 673–682.

[16] L. F. Shampine, *Numerical Solution of Ordinary Differential Equations*, Chapman and Hall, New York, 1994.

[17] D. F. Shanno, *Conditioning of quasi-Newton methods for function minimization*, Math. Comp., 24 (1970), pp. 647–657.

[18] A. Simeonov, M. Matsushita, E. A. Juban, E. H. Z. Thompson, T. Z. Hoffman, A. E. B. IV, M. J. Taylor, P. Wirsching, W. Rettig, J. K. McCusker, R. C. Stevens, D. P. Millar, P. G. Schultz, R. A. Lerner, and K. D. Janda, *Blue-fluorescent antibodies*, Science, (2000), pp. 307–313.

[19] S. Smolyak, *Quadrature and interpolation formulas for tensor products of certain classes of functions*, Soviet Math. Dokl., 4 (1963), pp. 240–243.

[20] G. Wasilkowski and H. Wozniakowski, *Explicit cost bounds of algorithms for multivariate tensor product problems*, J. Complexity, 11 (1995), pp. 1–56.