

MINIMUM ENERGY AND STEEPEST DESCENT PATH ALGORITHMS FOR QM/MM APPLICATIONS

by

Steven Knox Burger

Department of Chemistry
Duke University

Date: _____

Approved:

Weitao Yang, Supervisor

David Beratan

Jie Liu

Steven Craig

Dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy
in the Department of Chemistry
in the Graduate School of
Duke University

2007

ABSTRACT

MINIMUM ENERGY AND STEEPEST DESCENT PATH
ALGORITHMS FOR QM/MM APPLICATIONS

by

Steven Knox Burger

Department of Chemistry
Duke University

Date: _____

Approved:

Weitao Yang, Supervisor

David Beratan

Jie Liu

Steven Craig

An abstract of a dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy
in the Department of Chemistry
in the Graduate School of
Duke University

2007

Copyright © 2007 by Steven Knox Burger
All rights reserved

Abstract

A number of new methods are presented to determine the reaction path both for chemical systems where the transition state(TS) is known and for the more complicated case when only the reactant and product are available. To determine the minimum energy path(MEP) without the TS two algorithms are developed.

The first MEP method is a quadratic string method (QSM) which is based on a multiobjective optimization framework. In the method, each point on the MEP is integrated in the descent direction perpendicular to path. Each local integration is done on an approximate quadratic surface with an updated Hessian allowing the algorithm to take many steps between energy and gradient calls. The integration is performed with an adaptive step size solver, which is restricted in length to the trust radius of the approximate Hessian. The full algorithm is shown to be capable of practical superlinear convergence, in contrast to the linear convergence of other methods. The method also eliminates the need for predetermining such parameters as step size and spring constants, and is applicable to reactions with multiple barriers. The method is demonstrated for the Müller Brown potential, a 7-atom Lennard-Jones cluster and the enolation of acetaldehyde to vinyl alcohol.

The second MEP method is referred to as the Sequential Quadratic Programming Method (SQPM). This method is based on minimizing the points representing the path in the subspace perpendicular to the tangent of the path while using a penalty term to prevent kinks from forming. Rather than taking one full step, the minimization is divided into a number of sequential steps on an approximate quadratic surface. The resulting method is shown to be capable of super-linear convergence. However, the emphasis of the algorithm is on its robustness and its ability to determine the reaction mechanism efficiently, from which TS can be easily identified and refined

with other methods. To improve the resolution of the path close to the TS, points are clustered close to this region with a reparametrization scheme. The usefulness of the algorithm is demonstrated for the Müller Brown potential, amide hydrolysis and an 89 atom cluster taken from the active site of 4-Oxalocrotonate tautomerase (4-OT) for the reaction which catalyzes 2-oxo-4-hexenedioate to the intermediate 2-hydroxy-2,4-hexadienedioate.

With the TS known we present two methods for integrating the steepest descent path (SDP). Also the concepts of stability and stiffness are elaborated upon. The first SDP method is an optimally combined explicit-implicit method for following the reaction path to high accuracy. Although the SDP is generally considered to be a stiff ODE, it is shown that the reaction path is not uniformly stiff and instead is only stiff near stationary points. The optimal algorithm is developed by combining the explicit and implicit methods with a simple criterion, based on the stiffness, to switch between the two. Using two different methods an algorithm is developed to efficiently integrate the SDP. This method is tested on a number of small molecules.

The final method given is based on the diagonally implicit Runge-Kutta framework, which is shown to be a general form for constructing stable, efficient steepest descent reaction path integrators, of any order. With this framework tolerance driven, adaptive step-size methods can be constructed by embedding methods to obtain error estimates of each step without additional computational cost. There are many embedded and non-embedded, diagonally implicit Runge-Kutta methods available from the numerical analysis literature and these are reviewed for orders 2,3 and 4. New embedded methods are also developed which are tailored to the application of reaction path following. All integrators are summarized and compared for three systems.

Acknowledgements

Thanks and honor go first to God who is the source of all good things. I would like to thank my advisor Prof. Weitao Yang for creating an environment to freely pursue my research and also for invaluable discussions and feedback. His ideas and insights were the beginning of every project I undertook.

I would also like to thank Dr. Hao Hu and Jerry Parks for taking time to help me understand the QM/MM methodology. Additionally I have enjoyed discussions with our many group members: Dr. Felipe Bulat, Dr. Aron Cohen, Dr. Xiangqian Hu, Dr. San-Huang Ke, Dr. Shahar Keinan, Dr. Andres Cisneros, Dr. Jiang Hong, Dr. Amy Boone, Dr. Takumi Hori, Dr. Rui Liu, Dr. Zhenyu Lu, Dr. Mingliang Wang, Dr. Qin Wu, Dr. Paula Mori-Sanchez, Jing Song and Dr. Shubin Liu.

Thanks also go to Dr. Michael Peterson and William Day for computer help and for assistance with administrative issues from Trish Mcmillan and Caroline Morris.

Financial support from the the National Institutes of Health, and Duke University have been gratefully appreciated.

Contents

Abstract	iv
Acknowledgements	vi
List of Figures	ix
List of Tables	xiii
1 Algorithms for Determining the Minimum Energy Path	1
1.1 Introduction	1
1.2 Quadratic String Method	5
1.2.1 Theory	5
1.2.2 String Method	9
1.2.3 Quadratic String Method	10
1.2.4 Nonlinear Equations	15
1.3 Sequential Quadratic Programming Method	21
1.3.1 Method	21
1.3.2 Sequential Quadratic Programming Method	24
1.4 Numerical Results	27
1.4.1 Müller Brown Potential	27
1.4.2 Lennard-Jones 7-atom Potential	33
1.4.3 Vinyl Alcohol to Acetaldehyde	36
1.4.4 Amide Hydrolysis	39
1.4.5 4-OT cluster	40
1.5 Summary	45
2 Methods for Integration of the Steepest Descent Path	46

2.1	Introduction	46
2.2	Combined Explicit-Implicit Method for Reaction Path Integration	50
2.2.1	The Steepest Descent Path	50
2.2.2	Stiffness	51
2.2.3	Stiff ODE Example	52
2.2.4	Stability Regions	55
2.2.5	Stabilized Explicit Runge-Kutta Integration	58
2.2.6	Combined Explicit-Implicit Method	60
2.3	Diagonal Implicit Runge-Kutta Methods for Reaction Path Integration . . .	62
2.3.1	Stability Measures	62
2.3.2	Solving Diagonal Implicit Runge-Kutta Equations	64
2.3.3	Survey of Non-embedded Methods	66
2.3.4	Review of Embedded Methods	69
2.3.5	New Specialized Embedded Methods for Reaction Path Integration .	73
2.3.6	Summary of methods	78
2.3.7	Algorithm for multi-stage implicit integration	79
2.4	Comparison of Integration methods on Potential Energy Surfaces	83
2.4.1	Müller Brown Potential	84
2.4.2	$SiH_2 + H_2 \rightarrow SiH_4$	85
2.4.3	$CH_3CHO \rightarrow CH_2CHOH$	92
2.4.4	$H_2ClC-C(+)H_2 + Cl^- \rightarrow ClHC=CH_2 + HCl$	94
2.5	Summary	99
	Bibliography	100
	Biography	106

List of Figures

1.1	A schematic of the clustering scheme for a 12 image path. The point TS is located at λ_{TS} with lines drawn at points $\lambda_{TS} \pm 2$. The points on the lower line are spaced a distance s_{avg} apart.	22
1.2	Müller Brown surface with the exact path (black curve). Converged paths at iteration 13 are shown for $\theta_0 = 0.9$ (small dashes), $\theta_0 = 0.8$ (larger dashes) and $\theta_0 = 0.5$ (solid lines).	28
1.3	Comparison of SQPM and QSM with the direct solution of the nonlinear equations using the LM method. A 10 image path was used and the nonlinear solvers were started at the 5 th iteration.	29
1.4	Convergence of (a) the norm of projected gradient and (b) the norm of the distance between the approximate and exact path x_e for a 20 image path. QSM converges quadratically for this example while the string methods converge linearly. For QSM the damped BFGS update was used and we set $\Delta_0 = 0.2$. In the SM $dt = 0.0004$	30
1.5	Energy profile for the MB potential with a 20 image page. Points from the different methods are compared to the closest point on the exact path. The arc length scale on x-axis is determined by exact results. Results are taken from the 10 th iteration when QSM method has fully converged.	31
1.6	QSM projected gradient convergence rates for 3 different updates. Results from the MB potential with a 20 image path.	32
1.7	Energy profile for the LJ_7 potential with a 20 images path, with the start and end structures depicted. Points are compared against the closest point on exact path. Results are taken from the 20 th iteration, when the energy profile for the QSM method no longer visibly changes.	33
1.8	LJ_7 convergence of the norm of the projected gradient. A 20 image path was used with $\Delta_0 = 0.03$ and a damped BFGS updated Hessian for QSM. For the VV SM, $dt = 0.015$	34
1.9	LJ_7 convergence of $\ \Delta TS\ $, the RMS distance between the exact transition state and closest point on the cubic spline interpolated path. A 20 image path was used with $\Delta_0 = 0.03$ and a damped BFGS updated Hessian for QSM. For the VV SM, $dt = 0.015$	35

1.10	Projected gradient convergence of nonlinear methods for LJ_7 compared to the QSM, with the same parameters as in Fig. 1.8.	36
1.11	The starting structure, vinyl alcohol, shown on left, and end structure, acetaldehyde, shown on right.	37
1.12	Energy profile for the reaction $CH_2CHOH \rightarrow CH_3CHO$ with a 10 image path. Points from string methods are compared to the closest point on exact path. Results are taken from the 45 th iteration, when the energy profile for QSM no longer visibly changes.	37
1.13	$CH_2CHOH \rightarrow CH_3CHO$ convergence of the norm of the projected gradient. A 10 image path was used with $\Delta_0 = 0.1\text{\AA}$ and a damped BFGS updated Hessian for the QSM. In the SM $dt = 0.3$, and in the NEB method $dt = 0.3$ and $k = 1$	38
1.14	Convergence of the norm of the projected gradient of nonlinear methods compared to the QSM for the system $CH_2CHOH \rightarrow CH_3CHO$. The same parameters were used as in Fig. 1.13.	39
1.15	The HF/3-21G transition state structure of amide hydrolysis.	40
1.16	SQPM convergence for amide hydrolysis. The right y axis shows $\Delta RMSD_{TS}$, the RMS all-atom distance between the exact TS and the highest point on a cubic spline path. On the left axis is the difference between exact TS energy and highest energy of the cubic spline fit of the energy. SQPM was run with $N = 20, \Delta_0 = 0.1, M=20, \mu = 0.001$ and $\eta = 0.5$	41
1.17	Energy barrier for amide hydrolysis on iteration 25.	42
1.18	4-OT cluster model with select atoms frozen, shown with stars.	43
1.19	Convergence plot for the HF/3-21G 4-OT cluster. Axis labels and parameters are the same as those in Fig. 1.16.	43
1.20	4-OT cluster barrier at the 75th iteration with SQPM. The exact TS state barrier is 13.92 kcal/mol.	44
2.1	Two solutions of an ODE which demonstrate the concept of stiffness. The rapidly varying solution 1 only contributes at the beginning of the integration while the slowly varying solution 2 is more characteristic of the correct numerical integration.	51

2.2	Comparison between the Euler and Implicit Euler methods for (a) the error per step with the arc length step fixed at 0.1. and (b) the arc length traveled with the error per step fixed at 0.01. For (a), $\ \delta q\ $ is norm of the closest deviation to the exact path, while in (b), s is the arc length. The PES is given by $V(x, y) = -\frac{1}{2}0.5x^2 + \frac{1}{2}100y^2$, with $(x_0, y_0) = (1, 1)$	54
2.3	Complex Runge-Kutta stability plots for $ R(\lambda h) \leq 1$, where $R(z)$ is the series in Eq. (2.19) truncated at order $s = p - 1$. Four regions appear from darkest (Euler method, $s=1$) to lightest (fourth order RK method, $s=4$). . .	57
2.4	Complex plot of the Chebyshev based functions for $ R(\lambda h) \leq 1$. Darker region is $s = 2$ and the lighter is $s = 3$	59
2.5	Spectral radius of Eq. (2.6) over the reaction $SiH_2 + H_2 \rightarrow SiH_4$. The Hessian is updated with BFGS from the exact Hessian given at the TS. . .	61
2.6	Stability plots for the 2 stage SDIRK method with $\gamma = \gamma_{-1}$, and 3 stage SDIRK methods with $\gamma = \gamma_1$ and $\gamma = \gamma_{-1}$, respectively. Contour lines enclose the darkest area where $ R(z) \rightarrow 0$ to the white area where $ R(z) > 1$.	67
2.7	The Müller-Brown Potential with the exact MEP connecting the TS at $(-0.82, 0.62)$ to the minimum at $(-0.56, 1.44)$	85
2.8	Energy profile for $SiH_2 + H_2 \rightarrow SiH_4$ as calculated with DUMKA3.	87
2.9	(a) Performance and (b) accuracy of explicit methods compared to the GS algorithm with $h = 0.1$ for $SiH_2 + H_2 \rightarrow SiH_4$	88
2.10	(a) Performance and (b) accuracy of explicit methods compared to the GS algorithm with $h = 0.01$ for $SiH_2 + H_2 \rightarrow SiH_4$	89
2.11	(a) Performance and (b) accuracy of the combined explicit-implicit methods compared to the GS algorithm with $h = 0.01$ for $SiH_2 + H_2 \rightarrow SiH_4$	90
2.12	Variable time step embedded method accuracies for the system $SiH_2 + H_2 \rightarrow SiH_4$ with $tol = 10^{-3}$. The total arc length of the path (t_{end}) is 8 angstroms. 'Method' is Butcher array used, 'Max/Min Error' is the Log_{10} maximum/minimum local error (left axis), 'Evaluations' is the total number of gradient evaluations and 'Steps' is the number of steps taken (right axis).	91
2.13	Energy profile for $CH_3CHO \rightarrow CH_2CHOH$	92
2.14	(a) Performance and (b) accuracy of the combined explicit-implicit methods compared to the GS algorithm for $CH_2CHOH \rightarrow CH_3CHO$	93

2.15	Energy profile for $H_2ClC-C(+H_2+Cl^- \rightarrow ClHC=CH_2+HCl$ as calculated with DUMKA3.	94
2.16	(a) Performance and (b) accuracy of the combined explicit-implicit methods compared to the GS algorithm for $H_2ClC-C(+H_2+Cl^- \rightarrow ClHC=CH_2+HCl$	95
2.17	Variable time step embedded method accuracies for the system $H_2ClC-C(+H_2+Cl^- \rightarrow ClHC=CH_2+HCl$ with $tol = 10^{-3}$. Heading are defined as in Table 2.12.	97

List of Tables

2.1	Order conditions for DIRK methods	74
2.2	Non-Embedded Methods. 'Method' is the Butcher array used, 'LE' the local error, 'p' the order, 's' the number of stages, 'IS' the number of implicit stages, 'stable' the stability of the method and 'ref' the reference in which the method was found.	79
2.3	Embedded Methods. LE, LEE1 and LEE2 are measures of the error and are described in the Summary of methods section. All other headings are the same as those in Table 2.3.6.	80
2.4	Non-embedded method results at fixed step size $h = 0.1$ on the Müller Brown surface. Ten steps in total were taken. 'Method' is Butcher array used, ' γ ' distinguishes between the (2,3) and (3,4) methods, Max/Min(LE) is the maximum/minimum local error and 'Eval' is the mean number of evaluations per step.	86
2.5	Non-embedded method accuracies for the system $H_2ClC-C(+)H_2 + Cl^- \rightarrow ClHC=CH_2 + HCl$ with $h = 0.1$ and 35 steps.Heading are defined as in Table 2.4.	96
2.6	Embedded method accuracies for the system $H_2ClC-C(+)H_2+Cl^- \rightarrow ClHC=CH_2+HCl$ with $h = 0.1$ and 35 steps. Max/Min(Δ_0 -LE) is given by $\max / \min(\log_{10}(\Delta_0) - \log_{10}(LE))$ and the other heading are defined as in Table 2.4.	98

Chapter 1

Algorithms for Determining the Minimum Energy Path

1.1 Introduction

Simulations to determine enzymatic mechanisms have become an increasing common application of computational chemistry. Often cluster models, mechanical/molecular mechanical (QM/MM)[1], or ONION methods[2] are used to approximate the system. Numerous reviews on these topics are available [3, 4, 5, 6]. Of central importance to these applications is an algorithm to determine the minimum energy path(MEP) when the reactant and product states are known.

Methods to determine the MEP are equally applicable on free energy or potential energy surfaces. While QM/MM studies have been done on free energy surfaces[7], it is more common to work on the potential energy surface and then to do sampling on the final path with methods such free energy perturbation (FEP) [8]. The algorithms developed in this work focus solely on potential energy surfaces.

At finite temperature there is no single path a system would necessarily take when moving between two points. Instead the motion of a system is best represented by an ensemble of trajectories.[9] However, methods that calculate ensembles such as transition path sampling[10] tend to be computationally expensive for large systems, and so a single representative pathway is often used instead. Examples of pathways include the least action [11], least-time [11], maxflux [12] and the MEP[13]. With any of these pathways, transition state theory [14] may then be used to approximately determine rate constants for each step.

Many methods have been developed to find the MEP. Among the more successful ones

are coordinate driving[15], the Ayala and Schlegel[16] method, nudged elastic band (NEB) method[17], and the string methods[18, 19, 20, 21, 22]. In our laboratory we have often used coordinate driving or NEB and the Ayala and Schlegel method in combination and on reduced dimensional surfaces, which has proven successful for studying large enzymatic systems.[23, 24, 25] However all these methods have their drawbacks. Coordinate driving for example, simply involves slowly changing one coordinate while minimizing the rest and is known to suffer from discontinuities in the path. The Ayala and Schlegel algorithm combines a TS search with an implicit integration from the progressively more accurate TS. Although effective at finding the exact path, the algorithm is slow to converge when far from the desired TS.

NEB is widely used because of its simplicity and robustness. With NEB the points are propagated downhill using the velocity Verlet algorithm with a spring force applied between images to keep them well separated. A switching function is also added to prevent kinks from forming in the path. The main problem with NEB is its slow convergence since it minimizes similar to a steepest descent method. Several groups [26, 27, 28] have sought to improve the convergence of NEB by using optimization methods to minimize an unstated function. They use the NEB force in place of the gradient and the norm of the gradient as a merit function. Generally, these attempts have not been stable or fail to achieve the convergence properties associated with their respective minimization algorithm. The most promising is the double nudged elastic band (dNEB) method which includes an additional “nudging” term and uses the force as a gradient that is minimized with a quasi-Newton method. The Ayala and Schlegel method minimizes the path by searching for the TS with the highest point on the path, then minimizing the rest of the path with the implicit trapezoidal integration method. This method works very well for single TS paths that can be closely approximated, but may perform poorly otherwise.

Most NEB type methods are hampered by the dual need to keep the images spaced out and to move the images downhill perpendicular to the tangent. Ren et. al proposed a string method (SM)[20] which avoids this problem by integrating the force tangent to the

path while redistributing the points on a polynomial interpolation. The method has proven effective[21] and has resulted in other methods[22, 18, 19, 29, 30] which extend the basic concept. Among these methods are some which “grow” a path from the endpoints [18, 19] and do not require an initial interpolation. Also an additional SM has been proposed which does not require a tangent definition[22].

SMs generally have the same linear rate of convergence as the NEB method, as is often demonstrated for standard two dimensional potentials. In section 1.2 we present the quadratic string method(QSM)[31] which is able to achieve super-linear convergence by adding a quasi-Newton quadratic approximation to the local surface at each point. The method then integrates the steepest descent path in the subspace tangent to the path. The integration is done with a 4/5 Runge-Kutta method which allows the step-size to be determined adaptively rather than being specified by the user. To keep the points correctly spaced out, they are redistributed evenly on a cubic spline.

The extra approximate quadratic information used by QSM yields a more accurate procedure for evolving the string downhill to the solution. This enables the QSM to take larger steps and to achieve superlinear convergence to the tangent approximated solution path. Additionally the QSM eliminates the need for the user to choose the step size required by the SM.

While QSM convergences well, for larger systems it is sometimes computational inefficient since the integrated steepest descent path often sharply zigzags toward the minimum. Also QSM, like many SMs, lack an effective way of dealing with kinks that develop in the path.

To deal with the shortcomings of QSM we present an alternative method in section 1.3 which circumvents the problems of integrating on the energy surface by instead using a pure minimization scheme. This method uses the the same quasi-Newton approximation to the surface as QSM, but does separate minimizations in a sequence of steps. When necessary the approximate energy surface is augmented with a penalty function to correct for kinks in the path. This method is referred to as the sequential quadratic programming method

(SQPM). The main strength of SQPM is its robustness and that it scales well with large system sizes. However, since SQPM does not integrate toward the solution, the gradient perpendicular to the path does not converge as easily as with QSM.

Finally in section 1.4, numerous examples are given demonstrating the convergence properties of both algorithms.

1.2 Quadratic String Method

In this section we show how the search for the MEP can be formulated as a multiobjective optimization problem.

1.2.1 Theory

The MEP is a curve in N dimensional space connecting two minima through a first order saddle point (the TS). Starting at a first order saddle point, the MEP $\mathbf{x}(t)$ is defined as the the SDP on a potential surface $V(\mathbf{x})$, where \mathbf{x} is the vector describing the physical coordinates of the system. Here t is a parameterization for which the SDP takes on the ODE form,

$$\frac{d\mathbf{x}(t)}{dt} = -\mathbf{g} \quad (1.1a)$$

where $\mathbf{g} = \nabla V(\mathbf{x}(t))$ is the vector derivative of $V(\mathbf{x}(t))$. For the case where the curve $\mathbf{x}(t)$ is parametrized by arc length s , it has been shown that,

$$\frac{ds}{dt} = \sqrt{\frac{d\mathbf{x}^T}{dt} \frac{d\mathbf{x}}{dt}} \quad (1.1b)$$

[32, 33]. This turns Eq. (1.1a) into the familiar form

$$\frac{d\mathbf{x}}{ds} = -\frac{\mathbf{g}}{\|\mathbf{g}\|}. \quad (1.1c)$$

At a stationary point equation Eq. (1.1c) is undefined and so the ODE does not have a unique solution. For the TS this problem can be circumvented with a frequency calculation.[32] For minima there is no such workaround. Since string methods start with only minima it is not clear how to solve Eq. (1.1) as an ODE. Instead we follow the approach of many authors and view Eq. (1.1) as a minimization problem.[17, 20, 34, 35] This approach is derived from Eq. (1.1c) which states that the normalized gradient of the potential energy surface is tangent to the solution of Eq. (1.1), $\mathbf{x}^*(s)$. Stated as a set of equations we have,

$$\mathbf{g}_\perp = \mathbf{g} - \left(\mathbf{g}^T \tau(\mathbf{x}) \right) \tau(\mathbf{x}) = 0 \quad (1.2)$$

where $\tau(\mathbf{x})$ is the normalized tangent to the path, $\tau(\mathbf{x}^*) = \frac{d\mathbf{x}^*}{ds}$.

To calculate the curve $\mathbf{x}^*(s)$ practically a discrete representation must be used. This limits the accuracy of the approximation of $\mathbf{x}^*(s)$ by $\mathbf{x}(s)$ to the interpolation scheme used. Minimizing this error implies adding a constraint whereby any partition of the parametrization used for N points give equal arc lengths between points. If the parametrization gives the endpoints at a and b and $a < t_1 < \dots < t_N < b$ then the constraint is simply,

$$\int_{t_{i-1}}^{t_i} ds = S_i(\mathbf{X}) = \int_{t_i}^{t_{i+1}} ds = S_{i+1}(\mathbf{X}) \quad (1.3)$$

where $S_i(\mathbf{X})$ is the arc length between points \mathbf{x}_{i-1} and \mathbf{x}_i . Here for simplicity of notation we write $\mathbf{x}(t_i)$ as \mathbf{x}_i and to avoid confusion we refer to the entire path as \mathbf{X} . Also we put the column or index number in the subscript position. For the discrete path then, \mathbf{X} is a matrix with columns \mathbf{x}_i . To put Eq. (1.3) into a form which is useful for minimization we write it as,

$$c_{1i}(\mathbf{x}_i; \mathbf{x}_{j \neq i}) = S_i(\mathbf{X}) - S_{i+1}(\mathbf{X}). \quad (1.4)$$

This is written to have a parametric dependence on the other points $\mathbf{x}_{j \neq i}$ to emphasize that they are constant with respect to a minimization in \mathbf{x}_i .

With constraint (1.4) one could minimize $\|\mathbf{g}_\perp\|$ directly. However, this function requires calculating $\frac{\partial^2 V(\mathbf{x})}{\partial x_i \partial x_j}$ to obtain the derivative and thus is not practical. Instead we note that if the tangents at the solution $\tau(\mathbf{x}(t)^*)$, are known, then each point from the initial path need only be minimized in the hyperplane defined such that its normal is parallel to the constant tangent $\tau(\mathbf{x}(t)^*)$. However, since $\tau(\mathbf{x}(t)^*)$ is not known, we take the usual assumption that the tangents are best approximated by a finite difference scheme based on the current approximate path.[17] This gives a second constraint: The minimization for a given point \mathbf{x}_i is restricted to the surface defined by $c_{2i}(\mathbf{x}_i; \mathbf{x}_{j \neq i}) = 0$, such that its normal $\nabla_{\mathbf{x}_i} c_{2i}(\mathbf{x}_i; \mathbf{x}_{j \neq i})$ always point in the direction of the tangent.

To elucidate c_{2i} we give two examples of possible surfaces. If the tangent is defined by the central difference equation, $\tau_i(\mathbf{X}) = \frac{\mathbf{x}_{i+1} - \mathbf{x}_{i-1}}{\|\mathbf{x}_{i+1} - \mathbf{x}_{i-1}\|}$ then $c_{2i} = (\mathbf{x}_i - \mathbf{x}_i^0)^T \tau_i(\mathbf{X})$ since $\nabla_{\mathbf{x}_i} c_{2i} = \tau_i(\mathbf{X})$, where \mathbf{x}_i^0 is the i^{th} point on the initial path. Clearly c_{2i} takes this form for all tangent approximations which are constant with respect to the point \mathbf{x}_i . If the tangent

instead takes the form of the forward difference then $\tau_i(\mathbf{X}) = \frac{\mathbf{x}_{i+1} - \mathbf{x}_i}{\|\mathbf{x}_{i+1} - \mathbf{x}_i\|}$ and therefore $c_{2i} = -2\|\mathbf{x}_{i+1} - \mathbf{x}_i\| + 2\|\mathbf{x}_{i+1} - \mathbf{x}_i^0\|$ since $\nabla_{\mathbf{x}_i} c_{2i} = \frac{\mathbf{x}_{i+1} - \mathbf{x}_i}{\|\mathbf{x}_{i+1} - \mathbf{x}_i\|}$. This surface is a hypersphere of radius $\|\mathbf{x}_{i+1} - \mathbf{x}_i^0\|$ centered at \mathbf{x}_{i+1} .

Incorporating the constraint c_{2i} with Eq. (1.4) gives N minimization problems, each in the space of the point \mathbf{x}_i ,

$$\begin{aligned} & \min_{\mathbf{x}_i} V(\mathbf{x}_i), \quad i = 1 \dots N \\ \text{subject to : } & c_{1i}(\mathbf{x}_i; \mathbf{x}_{j \neq i}) = S_i(\mathbf{X}) - S_{i+1}(\mathbf{X}) = 0 \\ & c_{2i}(\mathbf{x}_i; \mathbf{x}_{j \neq i}) = 0 \end{aligned} \quad (1.5)$$

where N is the number of points on the path and c_{2i} is determined by the finite difference tangent such that,

$$\nabla_{\mathbf{x}_i} c_{2i}(\mathbf{x}_i; \mathbf{x}_{j \neq i}) = \tau_i(\mathbf{X}) \quad (1.6a)$$

$$c_{2i}(\mathbf{x}_i; \mathbf{x}_{j \neq i}) = 0 \quad (1.6b)$$

As an example of Eq. (1.5), for the case of a central difference tangent with a linear spline, the system of equations would be,

$$\begin{aligned} & \min_{x_i} V(\mathbf{x}_i), \quad i = 1 \dots N \\ \text{subject to : } & \|\mathbf{x}_{i+1} - \mathbf{x}_i\| = \|\mathbf{x}_i - \mathbf{x}_{i-1}\|. \end{aligned} \quad (1.7)$$

The single constraint clearly satisfies the equal arclength requirement Eq. (1.3) for a linear spline which requires the points to be equally spaced apart. Also if we rearrange the constraint we obtain $c_{2i}(\mathbf{x}_i; \mathbf{x}_{j \neq i}) = \frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_{i-1}\|^2 - \frac{1}{2}\|\mathbf{x}_{i+1} - \mathbf{x}_i\|^2$, without changing the equality of the original expression. Taking the gradient of this expression results in the unnormalized central difference tangent, $\nabla_{\mathbf{x}_i} c_{2i}(\mathbf{x}_i; \mathbf{x}_{j \neq i}) = \mathbf{x}_{i+1} - \mathbf{x}_{i-1}$. Solving Eq. (1.5) appears to give the correct solution $\mathbf{x}^*(t)$ to Eq. (1.1), however without a small fix this is not true. To see this we assume all other points are held fixed. The Lagrangian for the i^{th}

point of Eq. (1.5) is then

$$L_i(\mathbf{x}_i, \lambda_{1i}, \lambda_{2i}) = V(\mathbf{x}_i) + \lambda_{1i}c_{1i}(\mathbf{x}_i; \mathbf{x}_{j \neq i}) + \lambda_{2i}c_{2i}(\mathbf{x}_i; \mathbf{x}_{j \neq i}). \quad (1.8)$$

At the solution $\nabla_{\mathbf{x}_i} L_i(\mathbf{x}_i, \lambda_{1i}, \lambda_{2i}) = 0$, we have

$$-\nabla V(\mathbf{x}_i) = \lambda_{1i} \nabla c_{1i}(\mathbf{x}_i; \mathbf{x}_{j \neq i}) + \lambda_{2i} \nabla c_{2i}(\mathbf{x}_i; \mathbf{x}_{j \neq i}). \quad (1.9)$$

Since we have defined $\nabla c_{2i}(\mathbf{x}_i; \mathbf{x}_{j \neq i}) = \tau_i(\mathbf{X}) = \frac{d\mathbf{x}}{ds}$, clearly this is only a solution which satisfies Eq. (1.1c) if $\lambda_{1i} = 0$ and $\lambda_{2i} = \|\nabla V(\mathbf{x}_i)\|$. This implies that the $c_{1i}(\mathbf{x}_i; \mathbf{x}_{j \neq i})$ must be reformulated as an inequality constraint that is not active (not on the boundary of the constraint) at the solution. Since $c_{1i}(\mathbf{x}_i; \mathbf{x}_{j \neq i}) > 0$, we can choose $c_{1i}(\mathbf{x}_i; \mathbf{x}_{j \neq i}) \leq \epsilon_i$ for an $\epsilon_i > 0$, which solves the problem.

Minimization of Eq. (1.5) for an inactive $c_{1i}(\mathbf{x}_i; \mathbf{x}_{j \neq i})$ requires a descent direction, \mathbf{d}_i , that satisfies to first order, $\nabla V(\mathbf{x}_i)^T \mathbf{d}_i < 0$ and $\nabla c_{2i}(\mathbf{x}_i; \mathbf{x}_{j \neq i})^T \mathbf{d}_i = 0$. The direction,

$$\mathbf{d}_i(\mathbf{X}) = -\left(\mathbf{I} - \frac{\nabla c_{2i}(\mathbf{x}_i; \mathbf{x}_{j \neq i}) \nabla c_{2i}(\mathbf{x}_i; \mathbf{x}_{j \neq i})^T}{\|\nabla c_{2i}(\mathbf{x}_i; \mathbf{x}_{j \neq i})\|^2}\right) \nabla V(\mathbf{X}) = -\left(\mathbf{I} - \frac{\tau_i(\mathbf{X}) \tau_i(\mathbf{X})^T}{\|\tau_i(\mathbf{X})\|^2}\right) \nabla V(\mathbf{X}) \quad (1.10)$$

can be shown to satisfy both conditions, where \mathbf{I} is the identity matrix.[36] This is the same descent direction proposed in the SM. However, here the equation has been derived with optimization theory.

For the case where $c_{1i}(\mathbf{x}_i; \mathbf{x}_{j \neq i}) = \epsilon_i$, a modified direction may be used to correct the spacing of the points. The direction must satisfy $\nabla V(\mathbf{X})^T \mathbf{d}_i < 0$ and $\nabla c_{2i}(\mathbf{X})^T \mathbf{d}_i = 0$ as before, plus $\nabla c_{1i}(\mathbf{x}_i; \mathbf{x}_{j \neq i})^T \mathbf{d}_i \leq 0$. Such a direction may be obtained by choosing a vector $\mathbf{v}_i(\mathbf{X})$ in the space $\left(\mathbf{I} - \frac{\nabla c_{2i}(\mathbf{x}_i; \mathbf{x}_{j \neq i}) \nabla c_{2i}(\mathbf{x}_i; \mathbf{x}_{j \neq i})^T}{\|\nabla c_{2i}(\mathbf{x}_i; \mathbf{x}_{j \neq i})\|^2}\right)$ which maximizes the overlap $-\nabla c_{1i}(\mathbf{x}_i; \mathbf{x}_{j \neq i})^T \mathbf{v}_i$. The vector

$$\mathbf{v}_i(\mathbf{X}) = \left(\mathbf{I} - \frac{\nabla c_{2i}(\mathbf{x}_i; \mathbf{x}_{j \neq i}) \nabla c_{2i}(\mathbf{x}_i; \mathbf{x}_{j \neq i})^T}{\|\nabla c_{2i}(\mathbf{x}_i; \mathbf{x}_{j \neq i})\|^2}\right) \nabla c_{1i}(\mathbf{x}_i; \mathbf{x}_{j \neq i}) \quad (1.11)$$

satisfies these conditions. Using Eq. (1.11), the modified direction takes the form,

$$\mathbf{d}'_i(\mathbf{X}) = \left(\mathbf{I} - \frac{\tau_i(\mathbf{X}) \tau_i(\mathbf{X})^T}{\|\tau_i(\mathbf{X})\|^2} - \frac{\mathbf{v}_i(\mathbf{X}) \mathbf{v}_i(\mathbf{X})^T}{\|\mathbf{v}_i(\mathbf{X})\|^2}\right) \nabla V(\mathbf{X}). \quad (1.12)$$

However this approach is not pursued here. Instead the points are periodically spaced out on an interpolating polynomial, as done in the SM. The bulk of the following two sections will be concerned with the efficient integration of equation Eq. (1.10) along the downhill path toward to the solution, first with the SM and then with the QSM.

1.2.2 String Method

The details of the SM are laid out in Algorithm 3.2 in Ref. [21]. The two important steps are integrating Eq. (1.10) forward and reparametrizing the path by polynomial interpolation when the points become too close together. Any integration method can be used in the SM but each function evaluation of Eq. (1.10) requires a gradient and energy call to the underlying program. As a result, accurate higher-order methods such as the 4th order Runge-Kutta (RK) are prohibitively expensive. Multistep methods can be used instead but they require a higher order one-step method to start and must be restarted at every reparametrization. Fortunately it is not critical to follow the path closely at the start and the algorithm only suffers near the solution where a more accurate method or a smaller step size is needed.

The authors of the SM offer a way of circumventing this convergence problem by solving Eq. (1.2) as a nonlinear set of equations. Since evaluating the Jacobian directly requires a Hessian evaluation, they use Broyden's method.[37] The Broyden method is superlinearly convergent and involves a rank one update scheme for the Jacobian. Like all nonlinear solvers the method must be close to the solution to work, and for practical purposes often requires an exact Jacobian evaluation when the algorithm stops making progress.

One additional problem with the SM is that like NEB it requires the user to choose a step size. If the step size dt is too large, the algorithm does not behave as expected. If dt is too small, the algorithm is slow to converge.

1.2.3 Quadratic String Method

In the QSM Eq. (1.10) is integrated forward on a quadratic approximate surface within the trust radius Δ_i of each point. At the end of each integration the surface and trust radii are then updated. The outline of the algorithm follows with further details given in each section.

- (I) Use an initial path if available, otherwise the path is set to a linear interpolation between the reactant and product.
- (II) Evaluate the energy and gradient of all points.
- (III) Update each Hessian \mathbf{H}_i . For the first few updates, if the points are sufficiently close together, neighboring points can also be used in updating.
- (IV) Update each trust radius Δ_i .
- (V) Integrate Eq. (1.10) with a variable step size method over the quadratic surfaces given by \mathbf{H}_i , stopping when one of the points reaches its trust radius. Any points that did not pass over a minimum, scale their coordinates and then reintegrate Eq. (1.10).
- (VI) Redistribute the points if necessary on a polynomial interpolation such that Eq. (1.3) is satisfied.
- (VII) If the maximum of the norms of the projected gradients ($\|\mathbf{g}_\perp\|$) is less than a given tolerance stop, otherwise continue from step II.

Hessian Update

In order to integrate along the path defined by Eq. (1.10) without evaluating the energy and gradient of the potential at each step, a local approximate surface is needed at each point. If a quadratic approximation is used it will be defined by the Taylor series as $V(\mathbf{x}_0 + \mathbf{dx}) = V(\mathbf{x}_0) + \mathbf{g}(\mathbf{x}_0)^T \mathbf{dx} + \frac{1}{2} \mathbf{dx}^T \mathbf{H}(\mathbf{x}_0) \mathbf{dx}$. Calculating the Hessian $H_{ij} = \frac{\partial^2 V(\mathbf{x})}{\partial x_i \partial x_j}$

exactly, is usually prohibitively expensive. So most algorithms, as done here as well, use an approximate version constructed by a series of updates from previous steps. Among the many possible updates we examine the SR1, DFP and BFGS.[37] Starting from an initial Hessian $\mathbf{H}_0 = \|\mathbf{g}\|\mathbf{I}$, the updates take the form,

$$\mathbf{H}_{SR1}^{k+1} = \mathbf{H}^k + \frac{(\gamma^k - \mathbf{H}^k \delta^k)(\gamma^k - \mathbf{H}^k (\delta^k)^T)}{(\gamma^k - \mathbf{H}^k (\delta^k)^T \delta^k)} \quad (1.13)$$

$$\mathbf{H}_{BFGS}^{k+1} = \mathbf{H}^k - \frac{\mathbf{H}^k \delta^k (\delta^k)^T \mathbf{H}^k}{(\delta^k)^T \mathbf{H}^k \delta^k} + \frac{\gamma^k (\gamma^k)^T}{(\gamma^k)^T \delta^k} \quad (1.14)$$

$$\mathbf{H}_{DFP}^{k+1} = \left(\mathbf{I} - \frac{\gamma^k (\delta^k)^T}{(\gamma^k)^T \delta^k} \right) \mathbf{H}^k \left(\mathbf{I} - \frac{\gamma^k (\delta^k)^T}{(\gamma^k)^T \delta^k} \right) + \frac{\gamma^k (\gamma^k)^T}{(\gamma^k)^T \delta^k} \quad (1.15)$$

where,

$$\delta^k = \mathbf{x}^{k+1} - \mathbf{x}^k \quad (1.16a)$$

$$\gamma^k = \mathbf{g}^{k+1} - \mathbf{g}^k. \quad (1.16b)$$

Here we put index k in the superscript position, as we do throughout the paper, to indicate the iterative step.

It has been shown that the DFP update gives the closest possible approximation of the exact Hessian per update.[36] The BFGS update by contrast is the closest update to the exact inverse Hessian. Numerical studies have shown that the BFGS update usually gives a better estimate of the Hessian than DFP because of intrinsic self-corrective behavior in the update.[37] The SR1 update is also known to give Hessians that are as good as and often better than the BFGS method.

The DFP and BFGS updates both maintain positive definiteness while the SR1 update does not. Maintaining positive definiteness of each Hessian can be a desirable trait in this algorithm since the integration follows the constrained steepest descent path, which in a direction of negative curvature doesn't have a minimum. However, if a strict trust radius is used this is not necessarily a problem.

In order for either Eq. (1.14) or Eq. (1.15) to maintain a positive definite Hessian, the curvature condition must be satisfied,

$$(\delta^k)^T \gamma^k > 0. \quad (1.17)$$

Normally in minimization algorithms there is a line search for which the Wolfe conditions guarantee equation Eq. (1.17) holds.[37] In this case we have no such condition. However if the path of integration for a given point follows a reasonably straight path, then $-\frac{\mathbf{g}_\perp^T \delta}{\|\mathbf{g}_\perp\| \|\delta\|} \approx 1$ and $\mathbf{g}_\parallel^T \delta = 0$, where $\mathbf{g}_\perp = \mathbf{g} - (\mathbf{g}^T \tau(\mathbf{x}))\tau(\mathbf{x})$ and $\mathbf{g}_\parallel = (\mathbf{g}^T \tau(\mathbf{x}))\tau(\mathbf{x})$. Also if the energy is minimized fully along \mathbf{g}_\perp then $\mathbf{g}_\perp^{k+1} = 0$. So since $(\delta^k)^T \gamma^k = (\mathbf{g}^{k+1})^T \delta^k - (\mathbf{g}^k)^T \delta^k = (\mathbf{g}_\perp^{k+1})^T \delta^k - (\mathbf{g}_\perp^k)^T \delta^k$, $(\mathbf{g}_\perp^k)^T \delta^k < 0$, and $(\mathbf{g}_\perp^{k+1})^T \delta^k = 0$, therefore $(\delta^k)^T \gamma^k = -(\mathbf{g}_\perp^k)^T \delta^k > 0$. Usually these assumptions hold true for this algorithm. To protect against the possibility they do not, we can use a protected update,

$$\mathbf{H}^{k+1} = \begin{cases} \mathbf{H}^{k+1} & \text{if } (\delta^k)^T \gamma^k > 0, \\ \mathbf{H}^k & \text{otherwise.} \end{cases} \quad (1.18)$$

This performs quite well, but it is desirable to avoid not updating \mathbf{H}^k . Also we would like to avoid cases where $(\delta^k)^T \gamma^k$ is very small relative to $\delta^T \mathbf{H}^k \delta$. The damped BFGS update solves these problems by using the vector \mathbf{r}^k instead of γ^k in the normal BFGS update [36] where,

$$\mathbf{r}^k = \theta^k \gamma^k + (1 - \theta^k) \mathbf{H}^k \delta^k \quad (1.19a)$$

$$\theta^k = \begin{cases} 1 & \text{if } (\delta^k)^T \gamma^k > 0.2(\delta^k)^T \mathbf{H}^k \delta^k, \\ \frac{0.8(\delta^k)^T \mathbf{H}^k \delta^k}{(\delta^k)^T \mathbf{H}^k \delta^k - (\delta^k)^T \gamma^k} & \text{otherwise.} \end{cases} \quad (1.19b)$$

This always ensures that $(\gamma^k)^T \delta^k > 0.2(\gamma^k)^T (\mathbf{H}^k)^T \gamma^k$.

Trust Radius Update

The trust radius, Δ defines the region in which it is assumed the Hessian gives a valid quadratic approximation. Starting with a user supplied Δ_0 for the first calculation, each subsequent Δ is updated based on a merit function. Here we use the potential energy, $V(\mathbf{x})$ as the merit function. The approximate version of $V(\mathbf{x})$, given by $m(\mathbf{dx})$, for an approximate Hessian takes the form of the truncated Taylor series

$$m(\mathbf{dx}) = V(\mathbf{x}_0) + \mathbf{dx}^T \nabla V(\mathbf{x}_0) + \frac{1}{2} \mathbf{dx}^T \mathbf{H} \mathbf{dx}. \quad (1.20)$$

The standard update method after moving from a point \mathbf{x}_0 to $\mathbf{x}_0 + \mathbf{dx}$ then is,

$$\rho = \frac{V(\mathbf{x}_0 + \mathbf{dx}) - V(\mathbf{x}_0)}{m(\mathbf{dx}) - m(0)} \quad (1.21a)$$

$$\Delta = \begin{cases} 2\Delta & \text{if } \rho > 0.75 \text{ and } \frac{5}{4}\|\mathbf{dx}\| > \Delta, \\ \frac{1}{4}\|\mathbf{dx}\| & \text{if } \rho < 0.25 \end{cases} \quad (1.21b)$$

Here ρ is used to determine how close the merit function is to the expected value given by $m(\mathbf{dx})$. Other merit functions are also possible. Since we know beforehand that $\|\mathbf{g}_\perp(\mathbf{x}_i)\|$ must be the 0 vector at the solution, $\|\mathbf{g}_\perp(\mathbf{x}_i)\|$ can serve as a merit function rather than the energy. However using $\|\mathbf{g}_\perp(\mathbf{x}_i)\|$ for the trust radius can often lead to problems since $\|\mathbf{g}_\perp(\mathbf{x}_i)\|$ is usually a much more rugged surface than $V(\mathbf{x})$.

Integration of the Steepest Descent Path

In order to integrate Eq. (1.10), we can rewrite it as an ODE. With a quasi-Newton Hessian and a normalized tangent, Eq. (1.10) takes the form,

$$\frac{d\mathbf{x}_i}{dt} = (\mathbf{g}_i^0 + \mathbf{H}_i(\mathbf{x}_i - \mathbf{x}_i^0))^T (I - \tau(\mathbf{X})\tau(\mathbf{X})^T) \quad (1.22)$$

where \mathbf{g}_i^0 and \mathbf{x}_i^0 are the results from the last potential energy evaluation. Any ODE solver can be used to integrate Eq. (1.22). To avoid the user having to determine the step size h , a Runge-Kutta adaptive step-size method is used in the QSM. This method adjusts the step size based on the accuracy needed. So when the current path is far from the MEP large and more inaccurate steps can be taken while closer to the solution where the path needs to be followed more accurately h becomes smaller.

In order to determine the error of a given step, two separate integration steps of different orders are taken. The higher order integration may be taken as the exact answer and the lower order method compared against it. Specifically for the QSM a 5th and 4th order RK step are taken. The error, ϵ is estimated as the difference between them,

$$\epsilon = \|\mathbf{x}_{RK5}^{k+1} - \mathbf{x}_{RK4}^{k+1}\| \quad (1.23)$$

For a desired error, ϵ_0 , then the step size can be updated by the formula,

$$h = h \left| \frac{\epsilon_0}{\epsilon} \right|^{\frac{1}{5}} \quad (1.24)$$

Full details of Eq. (1.23) and Eq. (1.24) and the adaptive step-size algorithm are given in Ref. [38].

The only element left to determine for the algorithm is ϵ_0 . This can be established from the need to reduce $\max_i(\|\mathbf{g}_\perp\|)$. For a given point, $\mathbf{d}\mathbf{g}_\perp = (\mathbf{H}\mathbf{d}\mathbf{x})_\perp$, and if we use the assumption that $\|\mathbf{g}_\perp\|$ will not be reduced by more than two orders of magnitude, we obtain $\epsilon_0 = \min \frac{\|\mathbf{g}_\perp\|}{100\|\mathbf{H}\|^{\kappa_i}}$, where κ_i is the potential scaling factor (otherwise set to 1).

When Eq. (1.22) is integrated forward, each point will either get trapped in a minimum or reach its respective trust radius, Δ_i , at a different time. There are a number of ways to perform the integration so that all points remain within their trust radius. The most straightforward way is to integrate until the first point reaches its trust radius and then stop. The main problem with this approach is if one point is in a particularly nonquadratic region with a small trust radius, all points will be held up.

Another approach is to fix each point which passes over a minimum or moves past its trust radius while letting the rest of the points continue. Whether the i^{th} point passes over a minimum can be determined at integration step k by the condition $\mathbf{d}_i(\mathbf{x}^k)^T \mathbf{d}_i(\mathbf{x}^{k-1}) < \mathbf{0}$, where $\mathbf{d}_i(\mathbf{x})$ is given by Eq. (1.10). Fixing points often performs well near the beginning of the algorithm, but near the end, this often leads to sections of the path moving away from the correct solution.

To achieve the ideal situation where each point reaches its minimum or trust radius at the same time, we look to scale the coordinate system of each point by an appropriate amount, κ_i . The scaling vector κ is determined by,

$$\Delta_i = \left\| \int_0^T \kappa_i (\mathbf{g}_i + \mathbf{H}_i \mathbf{d}\mathbf{x}_i)_\perp dt \right\| \quad (1.25)$$

for each point that reaches its trust radius before its minimum, \mathbf{x}_i^* , and

$$\|\mathbf{x}_i^0 - \mathbf{x}_i^*\| = \left\| \int_0^T \kappa_i (\mathbf{g}_i + \mathbf{H}_i \mathbf{d}\mathbf{x}_i)_\perp dt \right\| \quad (1.26)$$

for those points that pass over their minimum. Here T is the total integration time.

Determining κ prior to the integration is not possible since we do not know the path through $\mathfrak{R}^{N \times m}$ space, where N is the number of points on the path and m is the dimensionality of the system. To circumvent this problem the integration is done once with $\kappa = \mathbf{1}$ until a point reaches the trust radius and then κ is adjusted to a vector which is expected to satisfy Eq. (1.25) and Eq. (1.26) for the next integration. The changes in κ are bounded to protect against points with small $\|\mathbf{g}_\perp\|$ ending up with very large values of κ_i . The form,

$$\kappa_i = \begin{cases} \kappa_i \min(\frac{t^{k+1}}{T}, \frac{1}{2}) & \text{if } d(\mathbf{x}_i^k)^T d(\mathbf{x}_i^{k+1}) < 0, \\ \kappa_i \max(\frac{\Delta_i}{\|\mathbf{x}_i - \mathbf{x}_i^0\|}, 2) & \text{otherwise} \end{cases} \quad (1.27)$$

works well in practice. This can be applied after each integration to steadily bring the points closer to their bounds. Since the error in the integration grows with the size of the multiplicative factor, Eq. (1.27) is only applied a maximum of 4 times.

Redistribution

For polynomial interpolation we use a natural cubic spline with the arc length as the parametric variable. The points are fitted to the spline and then adjusted so they are equally spaced apart. We do not obtain any noticeably better results with other higher order forms of interpolation, such as quintic fits or forms which use derivative information from the tangent approximation such as cubic Bezier curves. Redistribution can be done after every step of the ODE integrator, or less frequently as in the SM. A good criteria is to redistribute the points if $\max_i \left| \|\mathbf{x}_i - \mathbf{x}_{i+1}\| - \|\mathbf{x}_i - \mathbf{x}_{i-1}\| \right| > \frac{\sum_i \|\mathbf{x}_i - \mathbf{x}_{i+1}\|}{10(N+1)}$.

1.2.4 Nonlinear Equations

Near the solution, rather than integrating Eq. (1.22) as an ODE, we can treat Eq. (1.2) as a nonlinear system of equations,

$$\mathbf{r}_i(\mathbf{x}) = (I - \tau_i(\mathbf{X})\tau_i(\mathbf{X})^T)\mathbf{g}_i(\mathbf{x}_i) = 0 \quad (1.28)$$

Solving this set of equations requires derivative information in the form of the Jacobian, $\mathbf{J}(\mathbf{x}) = \nabla \mathbf{r}(\mathbf{x})$. This involves the Hessian which makes solving the equations in a straightforward fashion too costly. To circumvent this problem, as mentioned earlier, it has been suggested in Ref. [21] to use the Broyden update, where the Jacobian is updated on the k^{th} iteration by the formula

$$\mathbf{J}^{k+1} = \mathbf{J}^k + \frac{(\gamma^k - \mathbf{J}^k \delta^k)(\delta^k)^T}{(\delta^k)^T \delta^k}. \quad (1.29)$$

For this particular problem, rather than use the full step $\delta^k = \mathbf{x}^{k+1} - \mathbf{x}^k$ at iteration k with $\gamma^k = \mathbf{r}^{k+1} - \mathbf{r}^k$, one can decompose the update into N steps. Each update would then use $\gamma^k = (\mathbf{r}_1^k, \dots, \mathbf{r}_i^{k+1}, \dots, \mathbf{r}_N^k) - \mathbf{r}^k$ and $\delta^k = (\mathbf{x}_1^k, \dots, \mathbf{x}_i^{k+1}, \dots, \mathbf{x}_N^k) - \mathbf{x}^k$ for $i = 1 \dots N$.

Alternatively, the gradient in Eq. (1.28) can be approximated with the quasi-Newton Hessian, so that $\mathbf{g}_i(\mathbf{x}_i) = \mathbf{g}_i + \mathbf{H}_i(\mathbf{x}_i - \mathbf{x}_i^0)$. Then Eq. (1.28) becomes

$$\mathbf{r}_i(\mathbf{x}) = (I - \tau_i(\mathbf{X})\tau_i(\mathbf{X})^T)(\mathbf{g}_i + \mathbf{H}_i(\mathbf{x}_i - \mathbf{x}_i^0)) = 0. \quad (1.30)$$

Taking the derivative of Eq. (1.30) gives the Jacobian,

$$\nabla_{\mathbf{x}_j} \mathbf{r}_i(\mathbf{x}) = \mathbf{H}_i(I - \tau_i \tau_i^T) \delta_{ij} + \mathbf{T}_i^j \tau_i \mathbf{g}_i + \tau_i (\mathbf{T}_i^j \mathbf{g}_i)^T, \quad (1.31)$$

where $\mathbf{T}_i^j = \nabla_{\mathbf{x}_j} \tau_i(\mathbf{X})$. At each iteration then, only the quasi-Newton Hessians are updated rather than the full Jacobian.

It is also possible to add equations, $\mathbf{es}(\mathbf{x})$, which enforce the equal spacing constraint. If we reduce the cubic spline interpolation to a linear one we obtain,

$$\mathbf{es}_i(\mathbf{x}) = \|\mathbf{x}_{i+1} - \mathbf{x}_i\|^2 - \|\mathbf{x}_i - \mathbf{x}_{i-1}\|^2. \quad (1.32)$$

However adding Eq. (1.32) to Eq. (1.28) slows the rate of convergence dramatically. Therefore we exclude it and instead redistribute when necessary.

Given a set of equations $\mathbf{r}(\mathbf{x}) = 0$ and an approximate Jacobian, there are a number of methods to find an \mathbf{x} closer to the solution.[37] If the current position is close to the solution, the expansion about $\mathbf{r}(\mathbf{x})$ gives $\mathbf{r}(\mathbf{x}_0 + \mathbf{dx}) = \mathbf{r}(\mathbf{x}_0) + \mathbf{J}(\mathbf{x}_0)\mathbf{dx} = 0$. Solving for \mathbf{dx} then gives a direction to move toward the solution. Further from the solution, trust radius or

line search methods can be used. Here we choose to use the trust radius approach. The problem to be solved is then,

$$\min_{\mathbf{dx}} \frac{1}{2} \|\mathbf{J}\mathbf{dx} + \mathbf{r}\|^2, \quad \text{subject to } \|\mathbf{dx}\| < \Delta \quad (1.33)$$

Using the Levenberg-Marquardt (LM) method [37], this can be reduced to a one dimensional root finding problem for the Lagrange multiplier $\lambda \geq 0$,

$$(\mathbf{J}^T \mathbf{J} + \lambda I) \mathbf{dx} = -\mathbf{J}^T \mathbf{r} \quad (1.34a)$$

$$\lambda(\Delta - \|\mathbf{dx}\|) = 0 \quad (1.34b)$$

This is the same form as a trust radius based minimization except with $\mathbf{H} = \mathbf{J}^T \mathbf{J}$ and $\mathbf{g} = \mathbf{J}^T \mathbf{r}$. Full details of the LM method and proof of the equivalence of Eq. (1.33) and Eq. (1.34) are found in Ref. [36].

Our approach uses the pseudobond model of the QM/MM interface developed by Zhang *et al.* [39] The reaction paths are determined by an iterative energy minimization procedure. The free energies along the reaction path are determined by free energy perturbation calculations and the harmonic approximation for the fluctuation of the QM subsystem.

All calculations were performed using QM/MM methodology [39, 15, 40] that has been implemented in a modified version of Gaussian 98 [41], which interfaces to a modified version of TINKER [42]. The AMBER94 all-atom force field parameter set [43] and the TIP3P model [44] for water were used.

A very important part of this QM/MM implementation is the use of the pseudobond model for the QM/MM boundary as developed in Ref. [39], which provides a smooth connection between the QM and the MM subsystems and an integrated expression for the potential energy of the overall system.

In the QM/MM potential energy model, the total energy of the system is

$$E_{Total} = E_{MM} + E_{QM} + E_{QM/MM}. \quad (1.35)$$

The QM/MM interactions ($E_{QM/MM}$) are taken to include bonded and non-bonded interactions. For the non-bonded interactions, the subsystems interact with each other

through Lennard–Jones and point charge interaction potentials. When the electronic structure is determined for the QM subsystem, the charges in the MM subsystem are included as a collection of fixed point charges in an effective Hamiltonian, which describes the QM subsystem. That is, in the calculation of the QM subsystem we determine the contributions from the QM subsystem (E_{QM}) and the electrostatic contributions from the interaction between the QM and MM subsystems as explained by Zhang *et al.* [15].

Geometry optimizations are carried out by an iterative minimization procedure as described by Zhang *et al.* [15] In this procedure one iteration consists of a complete optimization of the QM subsystem, followed by a complete optimization of the MM subsystem. At each point the subsystem not being optimized is held fixed at the geometry obtained from the previous iteration; QM/MM interactions are also included at each iteration. The iterations are continued until the geometries of both systems no longer change.

When the MM subsystem is being optimized, or a molecular dynamics simulation is being carried out on the MM subsystem, the QM/MM electrostatic interactions are approximated with fixed point charges on the QM atoms which are fitted to reproduce the electrostatic potential (ESP) of the QM subsystem [45].

The reaction paths were calculated using the reaction coordinate driving method (RCDM) [46]. This method introduces a harmonic restraint on the reaction coordinate, which is a linear combination of the distances between the atoms involved in the reaction to perform an optimization along a proposed reaction path. The reaction coordinate is given by the expression:

$$R = \sum_{i=1}^n a_i r_i, \quad (1.36)$$

where r_i are the distances between atoms, a_i is constant 1 for the distance that increases, -1 for the distance that decreases. The sum over i includes all the distances that change throughout the course of the reaction. R is included in the following energy expression:

$$E_{Restrain} = k(R - s)^2, \quad (1.37)$$

where R is given by Eq. 1.36, s is an adjustable parameter corresponding to the value of the reaction coordinate which is varied in a stepwise manner by 0.1 Å at each point on the PES, and k is a force constant. In this case the value of k was set to 2000 kcal/mol for all points. This energy is included in the total energy expression in the process of the optimization.

All the calculated reaction paths were determined by stepping forward (from initial state to final state for that particular step) and backward (from final state to initial state for that particular step) along the path several times until there was no change between the forward and backward paths.

Vibrational frequency calculations were performed on the structures obtained for the maxima and minima along the paths (reactant, product, intermediates and transition states) to characterize the stationary points on the PES. Stationary points with one and only one imaginary (negative) vibrational frequency were characterized as transition states. Reactant, product and stable intermediates were characterized as having no imaginary frequencies. All vibrational frequencies were calculated at the HF/3-21G level, with a scaling factor of 0.9409 [47].

After the reaction paths were determined, the free energy perturbation method (FEP) [48] was employed to determine the free energy profiles associated with the calculated reaction paths for Scheme A. These calculations were carried out in the following manner: In each molecular dynamics (MD) simulation, the QM subsystem was fixed to a given state along the reaction path. At each of these states the reaction coordinate had a particular value, and the QM subsystem had a fixed geometry and charge distribution as obtained from the reaction path calculation, this state is called a simulated state. The free energy changes associated with perturbing the simulated state “forward” and “backward” to neighboring states along the reaction path were calculated according to FEP theory [48].

The free energy perturbation calculations at the stationary points were further improved by calculating the contributions from fluctuations of the QM subsystem to the free energy difference [15, 40]. These contributions were determined by calculating the Hessian matrices

of the stationary points for the degrees of freedom involving atoms in the QM subsystem and the subsequent calculation of the vibrational frequencies. By using the quantum mechanical harmonic approximation, the change in contribution from the fluctuations of the QM subsystem for all stationary points was determined.

1.3 Sequential Quadratic Programming Method

1.3.1 Method

As for QSM a collection of points, \mathbf{x}_i are used to represent the path, and Eq. 1.1c is again treated as a multiobjective optimization problem, where each point is separately minimized in the hyperplane tangent to the path. Since the tangents of the final path are unknown, the minimization of each point is linked to that of neighboring points through the definition of the tangent. Written as a minimization problem this is,

$$\begin{aligned} & \min_{\mathbf{x}_i} V(\mathbf{x}_i), \quad i = 1 \dots N \\ \text{subject to :} & \quad \tau_i(\mathbf{x})^T (\mathbf{x}_i - \mathbf{x}_1^0) = 0 \end{aligned} \tag{1.38}$$

where N is the number of points on the path, $\tau_i(\mathbf{x})$ is the tangent, which depends on neighboring points and x_i^0 is the initial path. Quapp[30] suggested solving Eq. 1.38 with the tangent fixed to be $\tau = x_{N+1} - x_0$. For simple potentials this provides a close approximation to the MEP. In this work we propose updating the tangent with the upwind scheme[49]. This can be done at each minimization step but the convergence properties are much better if the minimization is split up into M steps. Assuming that one can calculate the gradient and an approximation to the Hessian $H_{ij} = \frac{\partial^2 V(\mathbf{x})}{\partial x_i \partial x_j}$ each minimization step then takes the form,

$$\begin{aligned} & \min_{\mathbf{dx}_{ik}} \mathbf{dx}_{ik}^T \mathbf{g}_{ik} + \frac{1}{2} \mathbf{dx}_{ik}^T \mathbf{H}_i \mathbf{dx}_{ik}, \quad i = 1 \dots N \\ \text{subject to :} & \quad \tau_{ik}^T \mathbf{dx}_{ik} = 0 \\ & \quad \|\mathbf{dx}_{ik}\| < \Delta_i / M \end{aligned} \tag{1.39}$$

where $k = 1 \dots M$, Δ_i is the trust radius for Hessian, τ_{ik} is the tangent which is updated at each step k and $\mathbf{g}_{ik} = \mathbf{g}_i + \mathbf{dx}_i \mathbf{H}_i$ with $\mathbf{dx}_i = \sum_{j=1}^k \mathbf{dx}_{ij}$. This minimization scheme works well when the points are close the MEP but when further away extra constraints need to be added to keep the points spaced out and to prevent the path from becoming kinked.

Reparametrization

To ensure correct spacing between points a constraint of the form $\|x_i - x_{i+1}\| = \|x_i - x_{i-1}\|$ may be directly added to Eq. 1.39. This can work well in some cases, but for low-dimensional system the extra constraint causes the minimization to be over constrained. and more generally the result is a less efficient minimization scheme. Instead it is better to spaced out the points equally on a cubic spline[38] at the end of each iteration k . To do this a cord-length parametrization can be used, where the path $\mathbf{x}(s)$ is parametrically given in terms of s such that, $s_i = s_{i-1} + \|x_i - x_{i-1}\|$, with $s_0 = 0$. Once the points are fitted to a cubic spline, new points can be chosen at knots $s_i = is_{N+1}/(N + 1)$. Since the motion of the path is perpendicular to the tangent, redistribution normally does not move the points significantly.

It is possible to use other reparametrization schemes for spacing out the points. For example, E et al. recommend weighting the spacing by the energy[22]. The knots can also be adaptively placed to minimize the error in the total path by placing more points in regions of high curvature[50].

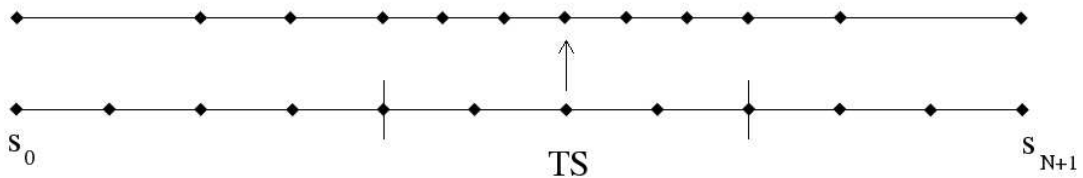


Figure 1.1: A schematic of the clustering scheme for a 12 image path. The point TS is located at λ_{TS} with lines drawn at points $\lambda_{TS} \pm 2$. The points on the lower line are spaced a distance s_{avg} apart.

For SQPM we aim to describe accurately the region of TS, so we use the following scheme to cluster points in this region. Assume that the TS is located closest to $s_{avg}\lambda_{TS}$, where λ_{TS} is an integer and $s_{avg} = s_{N+1}/(N + 1)$ is the average spacing. Then points between arc length 0 and $(\lambda_{TS} - 2)s_{avg}$ and points between $(\lambda_{TS} + 2)s_{avg}$ and s_{N+1} are

placed at knots doubly spaced apart with potentially extra points located at $(\lambda_{TS} \pm 3)s_{avg}$. The remaining points are equally spaced between $(\lambda_{TS} - 2)s_{avg}$ and $(\lambda_{TS} + 2)s_{avg}$. This scheme is shown in Fig. 1.1 for a 12 image path. Two lines are drawn at the bounding points $\lambda_{TS} \pm 2$, demonstrating the region into which the points are clustered.

There is a potential problem if λ_{TS} is located close to the middle of the spacing causing the index λ_{TS} to change frequently. Having the path regularly reparametrized is undesirable since this results in points moving significantly beyond their trust radius. To avoid this λ_{TS} is only changed if it different from the previous iteration by more than $\pm 3/4$.

Path Kinking

Kinking of the path can be a serious problem for any discrete representation of the path if a constraint force is not included. Kinking usually results if the points move at very different rates during each step of the algorithm or if the MEP has regions of high curvature and not enough points are used to represent the string. With QSM this problem is addressed by simply not including overly kinked points in the cubic spline interpolation when spacing out. This works well only when the MEP does not have regions of high curvature.

NEB deals with the kinking problem by introducing the deinking force,

$$1/2(1 + \cos(\pi(\cos\phi)))(F_s^T - (F_s^T \tau)\tau) \quad (1.40)$$

into the equations of motion for the string. In this equation F_s is the spring force between points. We propose a similar type of force for SQPM.

Using the standard definition of the angle, we define a point as being kinked if,

$$\cos(\phi) = \frac{(x_{i+1} - x_i)^T (x_i - x_{i-1})}{\|x_{i+1} - x_i\| \|x_i - x_{i-1}\|} < \eta, \quad (1.41)$$

where generally $0 < \eta < 0.5$. Eq. 1.41 can be added as an inequality constraint into Eq. 1.39. To solve this augmented minimization equation we use the active set approach[37]. In this approach, each point in Eq. 1.39 is solved without a constraint on the angle. Then for points with $\cos(\phi) < \eta$, Eq. 1.39 is resolved with the additional equality constraint $\cos(\phi) = \eta$.

1.3.2 Sequential Quadratic Programming Method

The minimization of Eq. 1.39 has three constraints. The first constraint keeps the solution in the hyperplane perpendicular to the tangent of the path. This has a linear form which can easily be eliminated by one of two ways. The most direct method is to solve $\tau^T dx = 0$ for dx_j where j is the first non-zero element of τ and then minimize with one less coordinate. This scheme usually works well, but sometimes can cause numerical instabilities. Alternatively the tangent vector can be QR factorized,

$$\tau = \begin{bmatrix} \mathbf{Q}_1 & \mathbf{Q}_2 \end{bmatrix} \begin{bmatrix} R \\ \mathbf{0} \end{bmatrix}. \quad (1.42)$$

In this simple case $R = 1$, and $\mathbf{Q}_1 = \tau$, while the $n - 1$ orthogonal directions to the tangent are the columns of $n \times n - 1$ matrix $\mathbf{Q} = \mathbf{Q}_2$. It is this later strategy which we use for the proposed method.

Eliminating the first constraint, leaves two more of the form $\|dx\| < \Delta$ and $\cos(\phi) = \eta$. If the angle constraint can be eliminated then the problem can be solved as an unconstrained minimization problem with a trust radius. We note that it is not crucial for the cosine of the angle to have the exact value η . So to eliminate the angle constraint we treat it as a penalty term. This means we can rewrite Eq. 1.39 as,

$$\min_{\mathbf{dx}_{ik}} \mathbf{dx}_{ik}^T \mathbf{Q}^T \mathbf{g}_{ik} + \frac{1}{2} \mathbf{dx}_{ik}^T \mathbf{Q}^T \mathbf{H}_i \mathbf{Q} \mathbf{dx}_{ik} + \frac{1}{2\mu} (\cos(\phi_i) - \eta)^2, \quad i = 1 \dots N$$

subject to :

$$\|\mathbf{Q} \mathbf{dx}_{ik}\| < \Delta_i / M \quad (1.43)$$

where \mathbf{Q} is from QR factorization of the tangent and \mathbf{dx}_{ik} is a $n - 1$ dimensional vector. Here we have introduced a penalty parameter μ , its value should not significantly affect unkinked points. Equation 1.43 can be solved with any unconstrained minimization scheme. To simplify the calculations we approximate $c(\mathbf{x}_i) = \cos(\phi(\mathbf{x}_i)) - \eta$ as a quadratic function in the same manner as was done going from Eq. 1.38 to Eq. 1.39. Substituting the quadratic expansion for $c(\mathbf{x})^2$ gives,

$$\min_{\mathbf{dx}_{ik}} \mathbf{dx}_{ik}^T \mathbf{Q}^T (\mathbf{g}_{ik} + \mathbf{g}_i^c) + \frac{1}{2} \mathbf{dx}_{ik}^T \mathbf{Q}^T (\mathbf{H}_i + \mathbf{H}_i^c) \mathbf{Q} \mathbf{dx}_{ik}, \quad i = 1 \dots N$$

subject to :

$$\|\mathbf{Q} \mathbf{dx}_{ik}\| < \Delta_i / M \quad (1.44)$$

where $g_i^c = \frac{c(\mathbf{x}_i)}{\mu} \nabla c(\mathbf{x}_i)$ and $H_i^c = \frac{c(\mathbf{x}_i)}{\mu} \nabla^2 c(\mathbf{x}_i) + \frac{1}{\mu} \nabla c(\mathbf{x}_i) \nabla c(\mathbf{x}_i)$. Since many small steps are taken, this as a reasonable simplification.

If a quasi-Newton Hessian is maintained at each point on the path then the algorithm for SQPM takes the following form:

- (I) Evaluate the energy, E_i and gradient \mathbf{g}_i at each point.
- (II) Update the N quasi-Hessians \mathbf{H}_i and trust radii Δ_i
- (III) FOR k = 1 ... M
- (IV) Calculate the tangents $\tau_i(\mathbf{x}_i)$.
- (V) FOR i = 1 ... N
- (VI) Solve Eq. 1.43 for $\mathbf{d}\mathbf{x}_{ik}$ without the penalty term.
- (VII) IF ($\cos(\phi_i) < \eta$) THEN solve Eq. 1.44.
- (VIII) END FOR
- (IX) IF (k > L) THEN redistribute the points with clustering.
- (X) ELSE equally distribute the points with knot spacing s_{avg} .
- (XI) END FOR
- (XII) IF $\|\mathbf{g}_\perp\| > tol$ THEN GOTO (I)

The user provides the tolerance for the calculation tol , the penalty term μ , kinking angle η , initial trust radius Δ_0 , the number of divisions M and the iteration L , at which the points are to be clustered near TS. The convergence results of the algorithm are usually only sensitive to L and Δ_0 , so one can set $tol = 0.001$, $\mu = 0.001$, $\eta = 0.5$ and $M = 20$ to reduce the number of parameters.

For SQPM, we use the same damped BFGS update and trust radius update that was used in QSM [31]. The damped update is used because, as in QSM, with the frequent

redistribution of points there is no guarantee that the energy of a point will necessarily decrease.

The trust-radius constrained minimization is a well known optimization problem and can be solved any number of ways. For this implementation we chose to use the Steihaug conjugate gradient method, which can be found in standard numerical optimization texts[37, 36]. This is the regular conjugate gradient algorithm applied to inexactly solve the linear system of equations $\mathbf{H}\mathbf{dx} = -\mathbf{g}$ with a stopping criterion to maintain the trust radius.

Spectator Atoms

Often it is preferable not to include certain floppy coordinates or atoms in the definition of the reaction path since they are irrelevant to the description of the path. Xie et al.[24] used NEB to show that including soft degrees of freedom in the definition of the path can prevent the algorithm from converging to the MEP. To avoid this problem atoms were weighted differently to create a “chemical metric” for the path description. With SQPM, the problem is less severe, but including irrelevant atoms can cause kinks to appear in the path that otherwise would not be present. Instead it is preferable to simply minimize the position of these atoms.

Excluding certain atoms from the path is quite easy with the previous formalism. If X_{ignore} is the set of ignorable coordinates, then, in the same way as was done for the chemical metric[24], we can create a reduced set $X_{reduced} = X - X_{ignore}$. This set can be used to calculate the tangent and the chord parametrization for the spline. The QR factorization is then done with the reduced tangent, $\tau(X_{reduced})$ and the other coordinates do not have to be transformed since they are not constrained to the tangent hyperplane. Alternatively the elements of X_{ignore} in the tangent vector may be set equal to zero and the QR factorization may be done in the full space with the same result. The end result is that the coordinates of X_{ignore} are minimized with only the trust radius constraint.

1.4 Numerical Results

1.4.1 Müller Brown Potential

The Müller Brown Potential (MB) potential shown in Fig. 1.2 provides a good test model for developing a method.[51] The potential is 2D, which makes it easy to visualize and quick to evaluate. Unfortunately the MB potential is often deceptively easy compared to higher dimensional problems. This can be especially true if updates are used to approximate the Hessian, because the difference between the exact and approximate Hessian often becomes unrealistically small after a small number of updates.

Figure 1.2 shows the converged MEP for different values of η for SQPM. With high values of η the path cannot converge to the MEP and cuts the corner around the TS. For $\eta = 0.5$ the minimization of the path is unconstrained and converges to the exact path. This is the same path obtained with QSM.

The convergence results of $\|g_{\perp}\|$ for SQPM and QSM for a 10 image path are shown in Fig. 1.3. This figure demonstrates the possible super-linear convergence of these methods. However for SQPM without a large M it can be quite difficult to achieve these results for a molecular system. For this example the definition of minimization for SQPM is changed slightly so that Eq. 1.43 uses $\|\mathbf{dx}_{ik}\| < \Delta - \|\mathbf{dx}_i\|$ as the constraint and then scales the step with $\mathbf{dx}_{ik} = \mathbf{dx}_{ik} \max(1/M, \|\mathbf{dx}_{i0}\|/\|\mathbf{dx}_{ik}\|)$, and $\mathbf{dx}_{i0} = \mathbf{dx}_{i0} \min(\Delta_i/(M\|\mathbf{dx}_{i0}\|), 1)$. With the normal definition the path converges very close to the MEP but then oscillates around it because of the small value of M used. QSM does not have this problem. In both methods the difference to the exact path is limited by the accuracy of tangent approximation.

Also shown in Fig. 1.3 are the results of applying the LM method close to the solution. The Broyden method does best, while the SR1 updated Hessian method has results similar to the string methods. Both methods are started close to the solution because directly solving equations (23) and (24) from the linearly interpolated path gives unphysical answers. This is a result of not having constraints on the angles between images, so that the

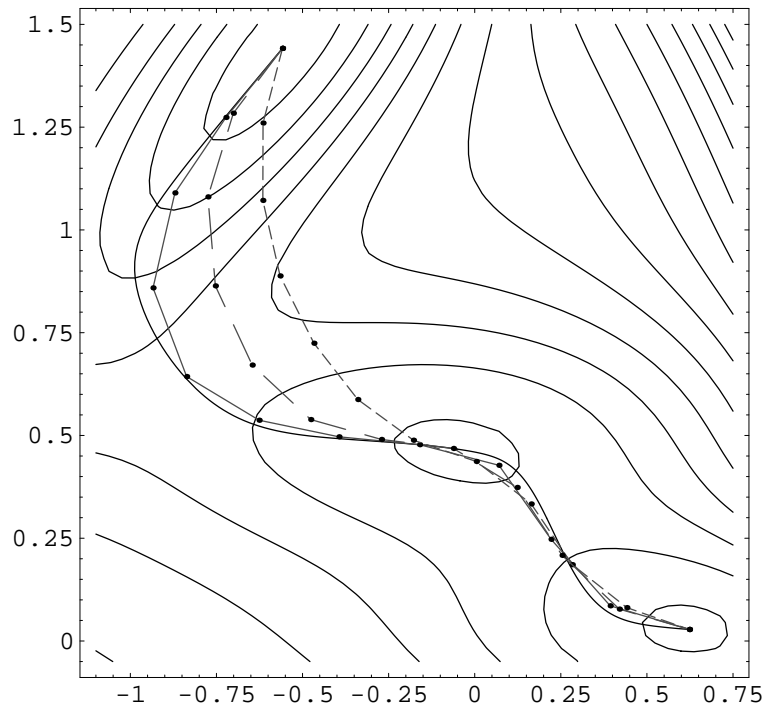


Figure 1.2: Müller Brown surface with the exact path (black curve). Converged paths at iteration 13 are shown for $\theta_0 = 0.9$ (small dashes), $\theta_0 = 0.8$ (larger dashes) and $\theta_0 = 0.5$ (solid lines).

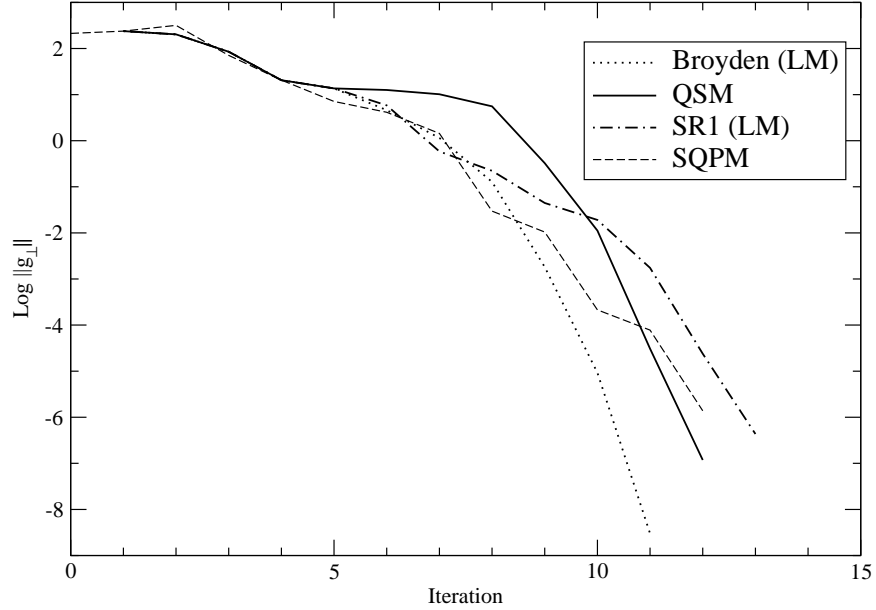


Figure 1.3: Comparison of SQPM and QSM with the direct solution of the nonlinear equations using the LM method. A 10 image path was used and the nonlinear solvers were started at the 5th iteration.

path quickly develops unrealistic kinks. Both LM methods applied in this case converge quadratically which is the same rate of convergence as when the Jacobian is calculated exactly.

A comparison between the SM and QSM for a 20 image path are shown in Fig. 1.4, with the converged energy profile for QSM appearing in Fig. 1.5. QSM is able to converge very rapidly with more points for this simple example. After 4 iterations, with 1 gradient and energy evaluation per iteration, the QSM path has converged within the limit of the tangent approximation. Reduction of the merit function $\|\mathbf{g}_\perp\|$ to within 10^{-5} takes about twice as long. From Fig. 1.4(b) we see that at convergence, the difference to the exact path is about 0.03.

The SM as presented in Ref. [21] does not include the Velocity Verlet (VV) algorithm. A quenched version was added in the Mathematica code. As Fig. 1.4 shows, adding the VV algorithm does not significantly improve the convergence for this case. However, for

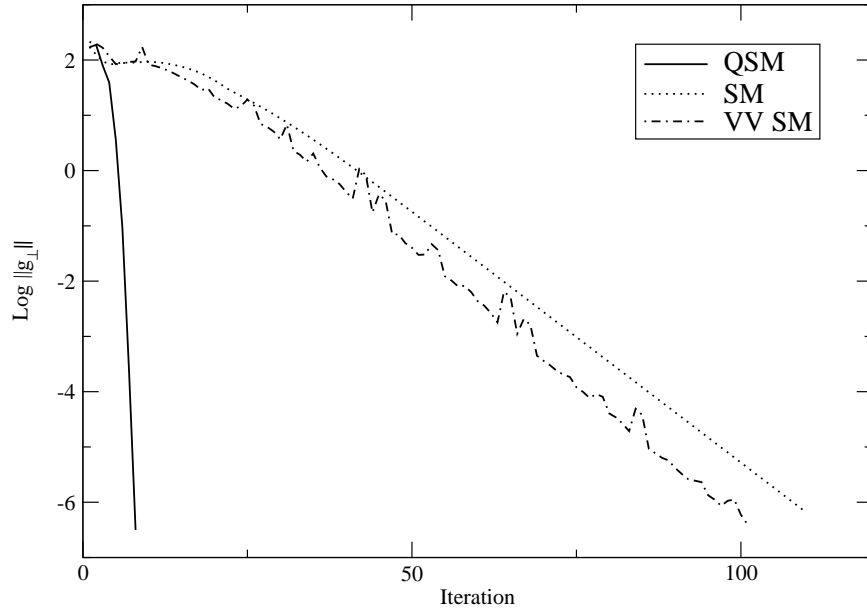


Figure 1.4: Convergence of (a) the norm of projected gradient and (b) the norm of the distance between the approximate and exact path x_e for a 20 image path. QSM converges quadratically for this example while the string methods converge linearly. For QSM the damped BFGS update was used and we set $\Delta_0 = 0.2$. In the SM $dt = 0.0004$.

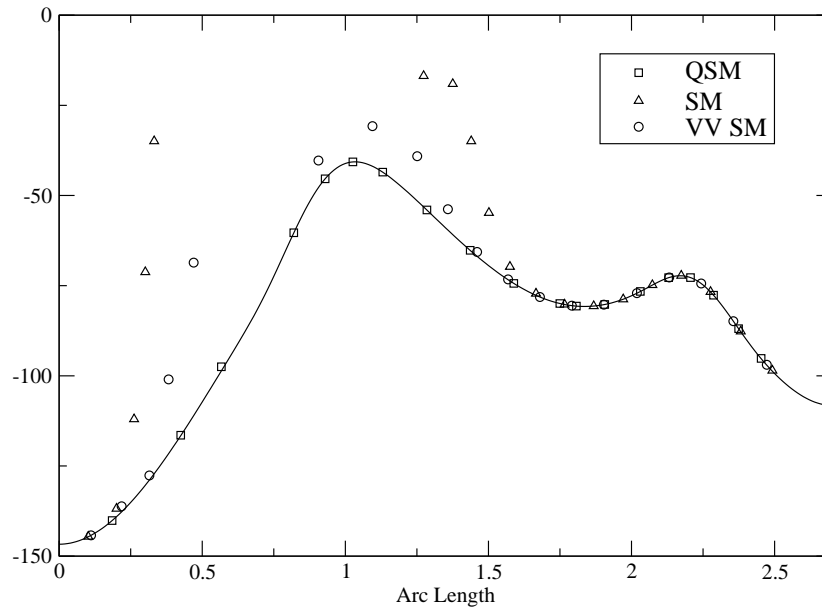


Figure 1.5: Energy profile for the MB potential with a 20 image page. Points from the different methods are compared to the closest point on the exact path. The arc length scale on x-axis is determined by exact results. Results are taken from the 10th iteration when QSM method has fully converged.

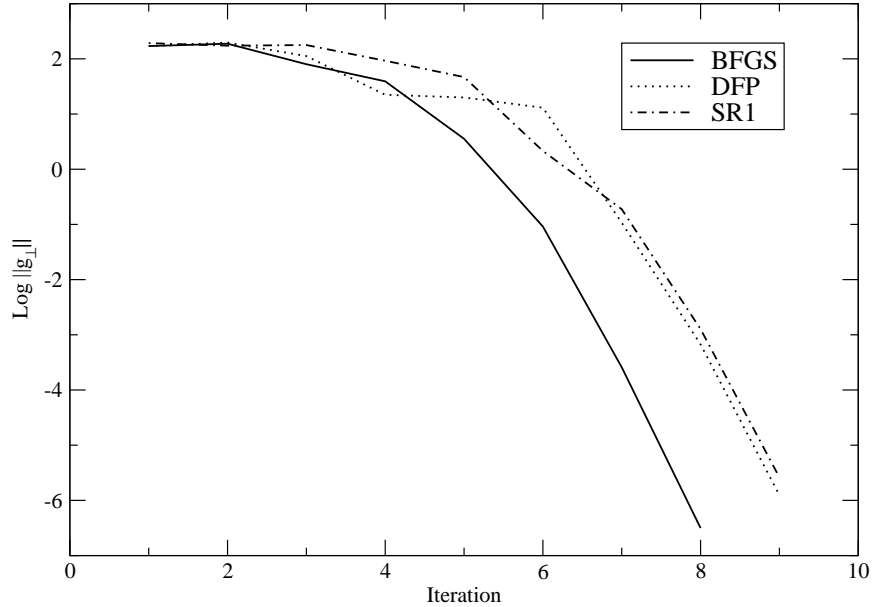


Figure 1.6: QSM projected gradient convergence rates for 3 different updates. Results from the MB potential with a 20 image path.

other examples and for less optimal choices of dt , the VV algorithm is much more effective than fixed step size Euler integration.

Figure 1.6, gives a comparison between different updates. Although the BFGS does best, the improvement over the other updates is not significant. For more complex systems the effects are much more pronounced; the DFP update is slow and will not work without some form of Hessian resetting, while the SR1 update usually requires a more stringent trust radius update that slows down its convergence. For all other calculations only the BFGS update was used.

The results for the MB potential are encouraging for the SQPM and QSM, but no algorithm can regularly give quadratic convergence without calculating second order terms. Higher dimensional systems must be used to demonstrate the true behavior of these methods.

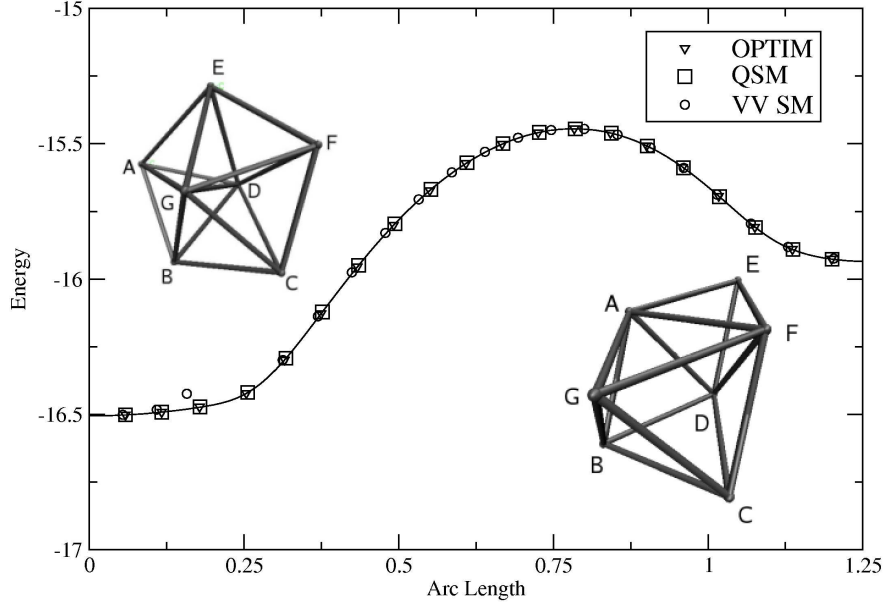


Figure 1.7: Energy profile for the LJ_7 potential with a 20 images path, with the start and end structures depicted. Points are compared against the closest point on exact path. Results are taken from the 20th iteration, when the energy profile for the QSM method no longer visibly changes.

1.4.2 Lennard-Jones 7-atom Potential

Lennard-Jones clusters are given by the simple pair-wise potential, $V_{LJ}(r) = -4\epsilon[\frac{\sigma}{r^6} - \frac{\sigma}{r^{12}}]$, and have been studied extensively.[52] Their global minima are cataloged for many different sizes in the Cambridge Cluster Database.[53] Here we focus on the 7-atom cluster for the reduced unit system given by $\epsilon = 1$ and $\sigma = 1$. We examine the rearrangement from the global minimum to a nearby local minimum shown in Fig. 1.7. These two structures differ by a root mean square (RMS) distance of 0.39, with one transition state in between, and were obtained from the website referred to in Ref. [27].

In Fig. 1.8 and Fig. 1.9 QSM is compared against the SM and the dNEB method from OPTIM. To compare against the output of OPTIM the RMS projected gradient is used as the measure of convergence,

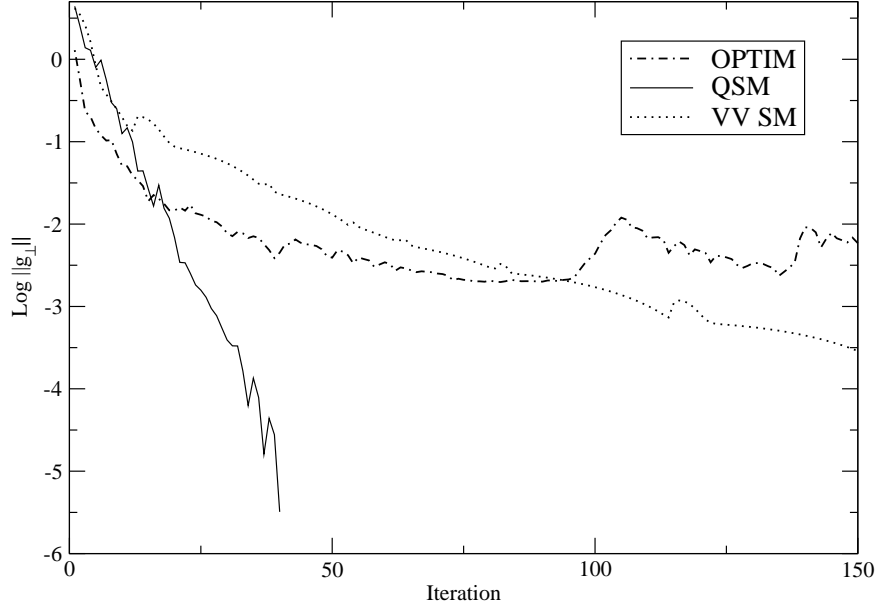


Figure 1.8: LJ_7 convergence of the norm of the projected gradient. A 20 image path was used with $\Delta_0 = 0.03$ and a damped BFGS updated Hessian for QSM. For the VV SM, $dt = 0.015$.

$$\mathbf{g}_\perp^{RMS} = \sqrt{\frac{\sum_{i=1}^N \|(\mathbf{g}_\perp)_i\|^2}{N\eta}} \quad (1.45)$$

where η is the number of degrees of freedom and N the number of images.

OPTIM is a hybrid path and transition state optimizer. As such, it is difficult to make a comparison of its convergence to the exact path and the results are excluded from Fig. 1.9. OPTIM is able to reduce $\|\mathbf{g}_\perp^{RMS}\|$ at almost twice the rate of QSM with the help of a TS search for the first 10 iterations. However after the TS is found, the algorithm is unable to fully converge to the rest of the path.

QSM converges at approximately twice the rate of the SM. The SM here again exhibits linear convergence. QSM is able to converge to the approximate path in about 20 iterations, with full convergence to a tolerance of 10^{-5} taking about twice as long. This was similar to the convergence results for the MB potential.

The results for nonlinear methods are displayed in Fig. 1.10. For this example the

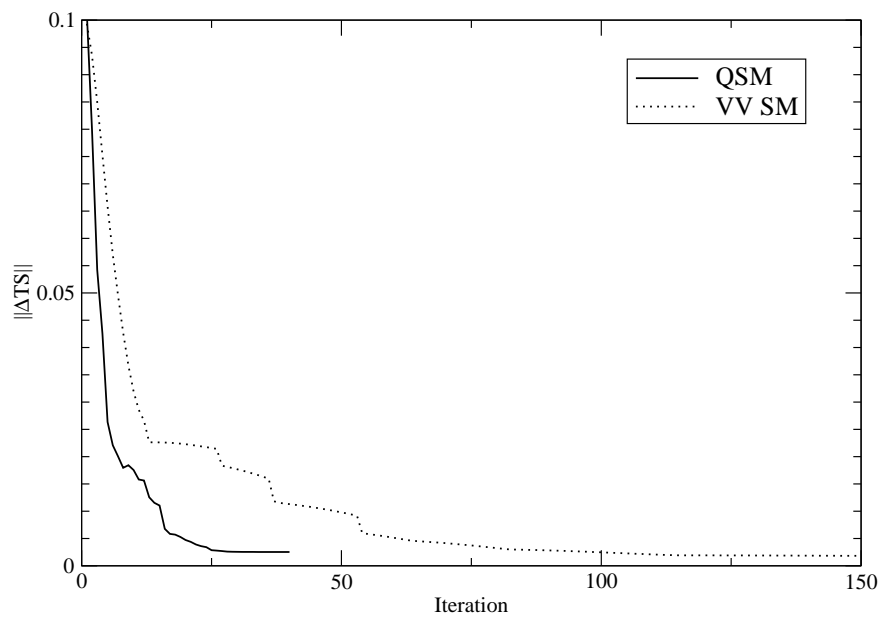


Figure 1.9: LJ_7 convergence of $\|\Delta TS\|$, the RMS distance between the exact transition state and closest point on the cubic spline interpolated path. A 20 image path was used with $\Delta_0 = 0.03$ and a damped BFGS updated Hessian for QSM. For the VV SM, $dt = 0.015$.

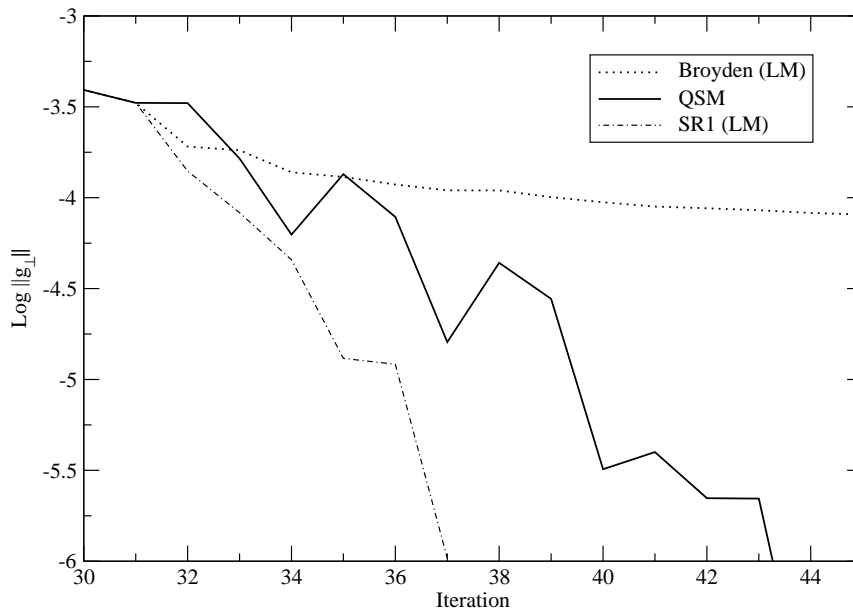


Figure 1.10: Projected gradient convergence of nonlinear methods for LJ_7 compared to the QSM, with the same parameters as in Fig. 1.8.

results are opposite those of the MB example. The Broyden method does not improve the convergence of QSM while the SR1 substituted Jacobian gives significantly better results. These favorable results can probably be attributed to the fact that the tangent derivative in Eq. (1.31), is given exactly.

1.4.3 Vinyl Alcohol to Acetaldehyde

Here we examine a simple reaction with the aid of Gaussian 98. The enol start and carbonyl end structures are shown in Fig. 1.11. All calculations were done using HF/STO-3G with Cartesian coordinates.

As seen in Fig. 1.12 and Fig. 1.13, QSM again compares favorably against other string methods. The NEB path is somewhat difficult to compare against QSM and SM since the code used does not remove overall rotation translation (ORT). This causes NEB to converge to a different path, further from the true transition state. This type of behavior was also noted in Ref. [27].

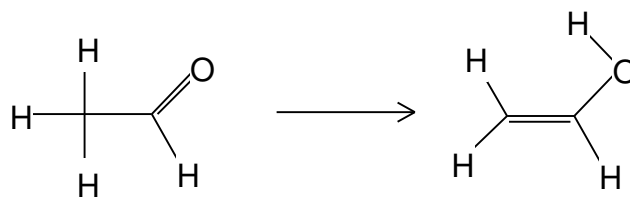


Figure 1.11: The starting structure, vinyl alcohol, shown on left, and end structure, acetaldehyde, shown on right.

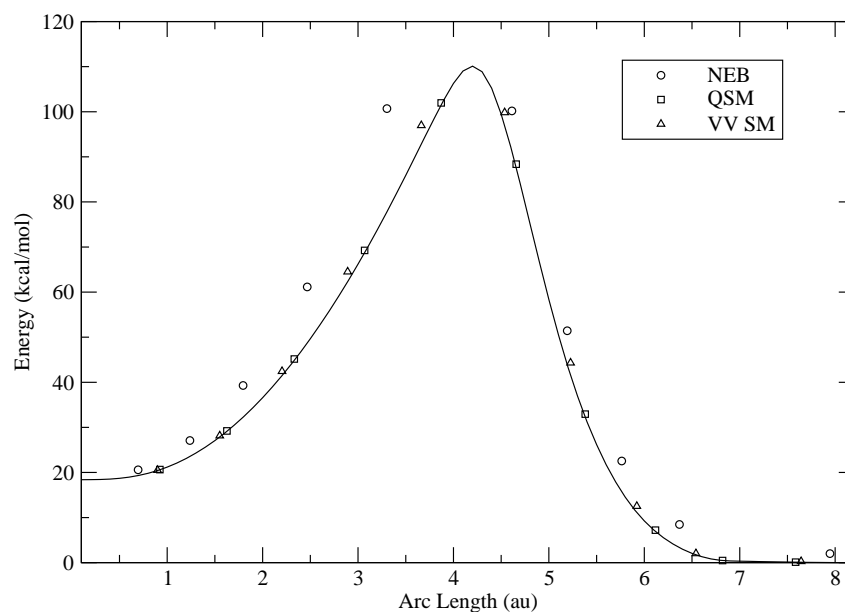


Figure 1.12: Energy profile for the reaction $CH_2CHOH \rightarrow CH_3CHO$ with a 10 image path. Points from string methods are compared to the closest point on exact path. Results are taken from the 45th iteration, when the energy profile for QSM no longer visibly changes.

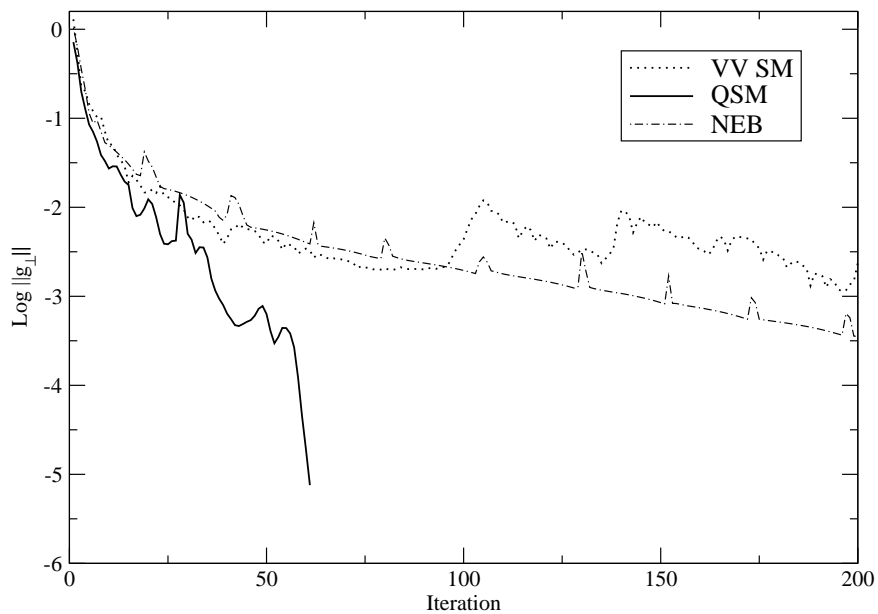


Figure 1.13: $CH_2CHOH \rightarrow CH_3CHO$ convergence of the norm of the projected gradient. A 10 image path was used with $\Delta_0 = 0.1\text{\AA}$ and a damped BFGS updated Hessian for the QSM. In the SM $dt = 0.3$, and in the NEB method $dt = 0.3$ and $k = 1$.

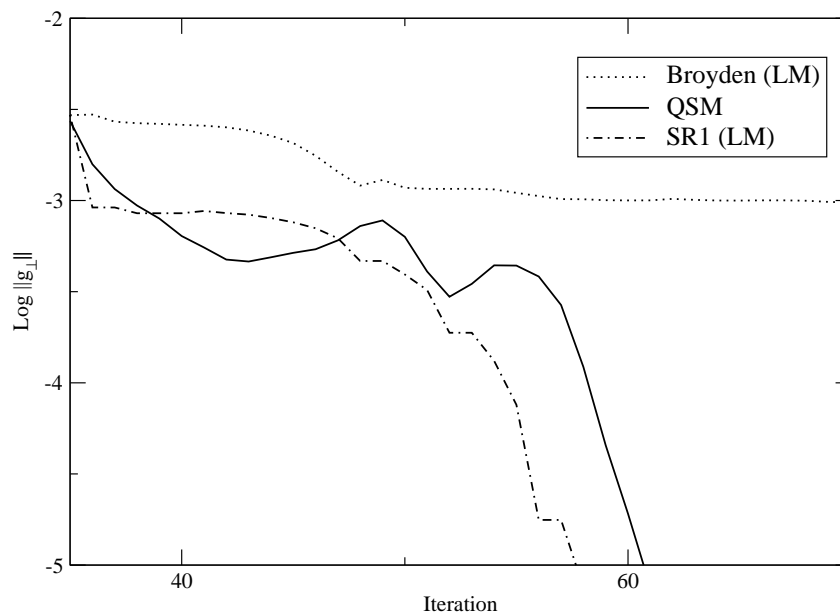


Figure 1.14: Convergence of the norm of the projected gradient of nonlinear methods compared to the QSM for the system $CH_2CHOH \rightarrow CH_3CHO$. The same parameters were used as in Fig. 1.13.

The VV SM converges at a similar rate to NEB, which is not surprising, since they both use the same quenched VV algorithm. QSM is able to converge to a tolerance of 10^{-5} very rapidly once it gets close the solution. Notably both the VV SM and the NEB method appear to give a much smoother convergence than QSM. This is because QSM minimizes the energy of each point path. As a result the sum of the energies of all the points decreases uniformly, but $\|\mathbf{g}_\perp\|$ and $\|\Delta TS\|$ do not.

The results of using nonlinear solvers appear in Fig. 1.14, and are similar to the LJ_7 example. The SR1 substituted Jacobian performs slightly better than QSM while the Broyden method converges very slowly.

1.4.4 Amide Hydrolysis

To test SQPM with a small molecule, we studied amide hydrolysis[54] at the HF/3-21G level of theory with $N = 20$. The TS for this reaction in the presence of two waters is

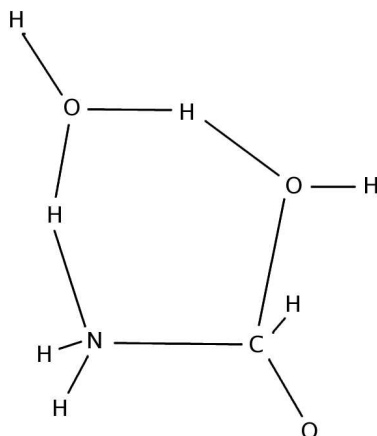


Figure 1.15: The HF/3-21G transition state structure of amide hydrolysis.

shown in Fig. 1.15. Given two minimized endpoints Fig. 1.16 demonstrates how SQPM can converge to within 1 kcal/mol of the true barrier after about 15 iterations. After 20 iterations the barrier is correctly predicted within 0.1 kcal/mol. At the 10th iteration the points are clustered near the TS and there is no discontinuity in either plot. The all-atom RMS difference to the TS monotonically decreases, but does not converge as well as the energy.

The results of clustering the points near the TS can be seen from the energy profile of Fig. 1.17. Because the cluster favors the TS region, the rest of the energy profile is not as well spaced out.

Using the TS structure from the 15th iteration, the exact TS was obtained using the “Opt(TS,CalcAll)” keyword in Gaussian 03. With the force constants calculated at each step only 4 iterations were required to converge the calculation.

1.4.5 4-OT cluster

To see how SQPM performs for an enzyme system we examine a constrained cluster of 4-OT which is shown in Fig. 1.18. For this cluster we have clipped out the substrate and the main contributing residues from a QM/MM optimized structure of the wild type 4-OT enzyme.[55] Part of backbone connecting Leu-8 and Ile-7 is included, as well as all of Pro-1

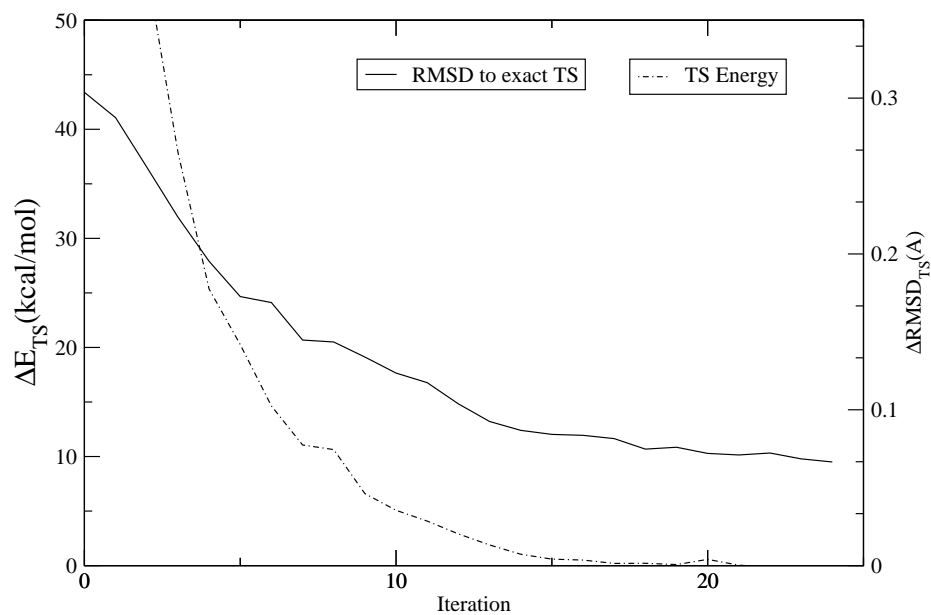


Figure 1.16: SQPM convergence for amide hydrolysis. The right y axis shows $\Delta RMSD_{TS}$, the RMS all-atom distance between the exact TS and the highest point on a cubic spline path. On the left axis is the difference between exact TS energy and highest energy of the cubic spline fit of the energy. SQPM was run with $N = 20, \Delta_0 = 0.1, M=20, \mu = 0.001$ and $\eta = 0.5$.

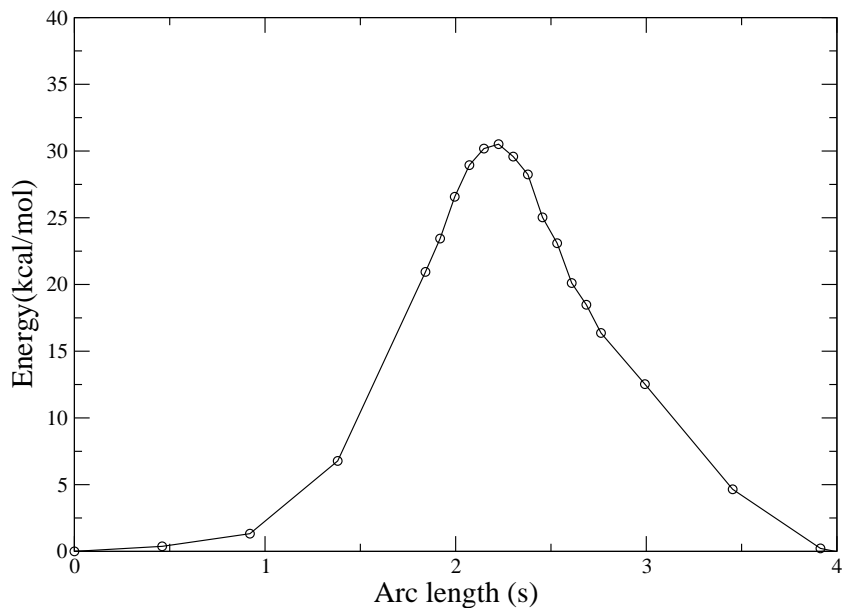


Figure 1.17: Energy barrier for amide hydrolysis on iteration 25.

and the guanidino group from Arg-61' and Arg-39''. Some waters were also left in, giving a total of 89 atoms in the system. To maintain a stable structure a total of 16 atoms were fixed. Since only the proline and substrate participate in the reaction, all the atoms on the water and residues Leu-8, Ile-7, Arg-61' and Arg-39'' were set as spectator atoms and not included in the path.

To represent the path 22 points were used and the path was reparametrized to cluster near the TS at the 25th iteration. The convergence results are shown in Fig. 1.19. With such a large system it takes significantly longer to converge to the correct path as compared to the previous examples. The barrier converges within 2 kcal/mol of the exact barrier of 13.92 kcal/mol within 35 iterations. About 25 more iterations are required to fully refine the barrier to within 1 kcal/mol. The RMSD to the exact TS declines monotonically as in the previous example.

Figure 1.20 shows the energy barrier of the path at the 75th iteration. Clustering of the points gives a good resolution at the region near the TS. In this case, unlike amide hydrolysis, the energy changes significantly over a small RMSD. This is typical of large

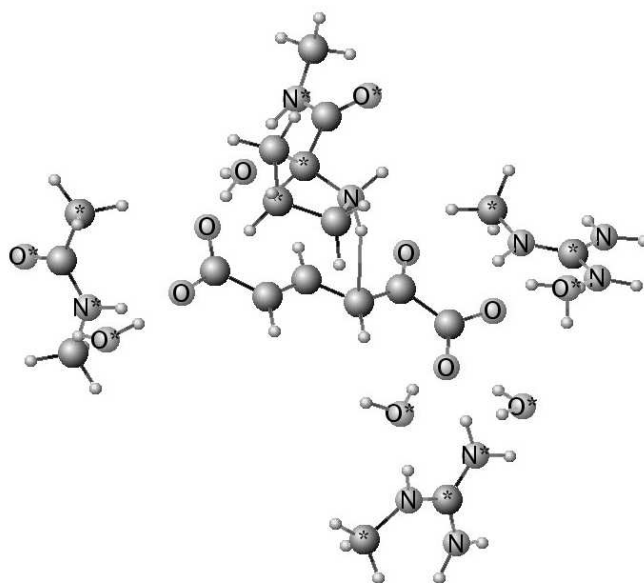


Figure 1.18: 4-OT cluster model with select atoms frozen, shown with stars.

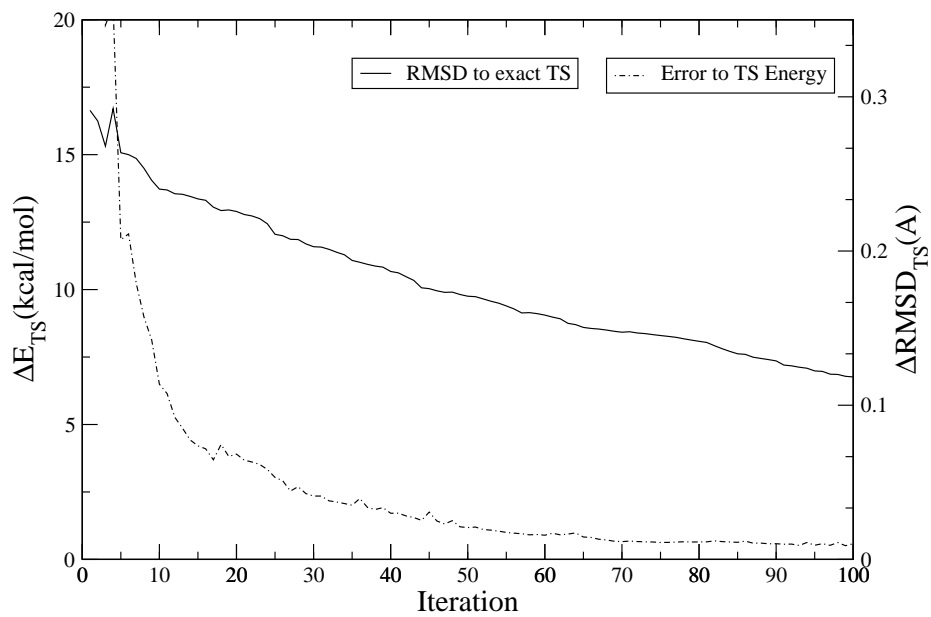


Figure 1.19: Convergence plot for the HF/3-21G 4-OT cluster. Axis labels and parameters are the same as those in Fig. 1.16.

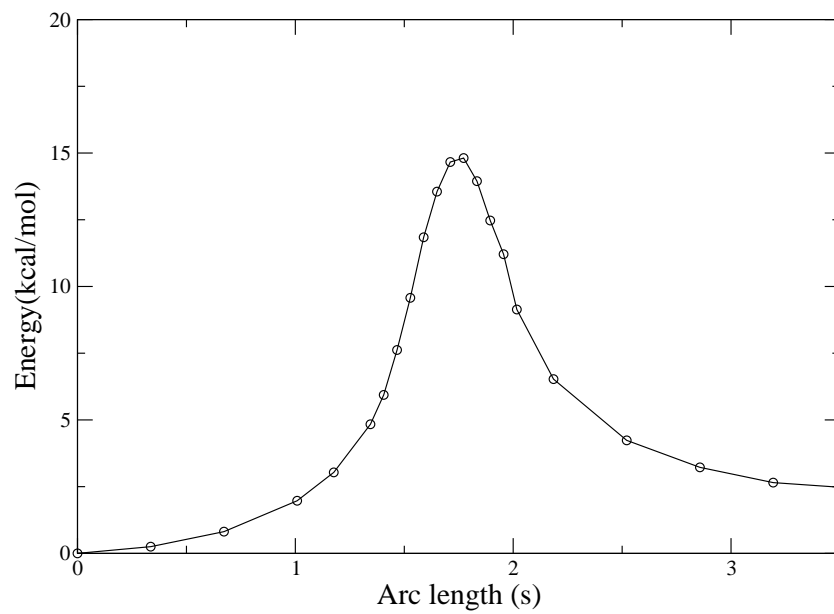


Figure 1.20: 4-OT cluster barrier at the 75th iteration with SQPM. The exact TS state barrier is 13.92 kcal/mol.

systems when Cartesian coordinates are used.

1.5 Summary

A number of conclusions can be drawn from success of QSM and SQPM. Most importantly we demonstrated that using a quasi-Newton approximation to the local surface allows for the development of algorithms that are significantly better than existing methods.

Both methods were shown to be capable of superlinear convergence by recasting the problem as a multi-objective minimization. We also showed that spacing out the points on a cubic spline kept the points correctly separated without interfering with the minimization. Also by adding a penalty term in SQPM kinks could be handled in a way that didn't interfere with the minimization of the rest of the path.

The use of a variable time step ODE method removed the need for the user to specify parameters such as dt and the spring constant k . This prevented the results from being dependent on choices by the user. For NEB and the SM often calculations had to be rerun with different values if progress was unduly slow toward the solution.

The damped BFGS update was found to be the best way to update each Hessian, rather than the the DFP and SR1 updates. However, when solving the nonlinear SDP equations, only the SR1 substituted Hessian consistently worked well. The Broyden method proved too slow compared to other methods.

The improved convergent behavior of SQPM and QSM was demonstrated for the MB surface which gave unrealistic quadratic convergence. The LJ_7 and vinyl alcohol systems demonstrated the true superlinear convergence properties of QSM. The amide hydrolysis and the 4-OT cluster reactions demonstrated that SQPM was capable of approximating paths for more realistic problems and that the TS could be closely approximated by clustering points in the region of interest. Using either method gave an approximate path from which the TS could easily be refined further with well established methods.[56]

Chapter 2

Methods for Integration of the Steepest Descent Path

2.1 Introduction

When characterizing the reaction mechanism or rate constants, once the minima and transition states (TS) are calculated, the next step usually involves a calculation of the reaction path. To estimate rate constants, for instance, an algorithm for following the reaction path is used with variational transition state theory or the reaction path Hamiltonian [57, 58]. The reaction path can also be used to fully elucidate the mechanism for a reaction and to ensure that the known TSs connect to the expected minima.

The reaction path is coordinate dependent, and at zero temperature is defined on a potential energy surface (PES) as the steepest descent path (SDP) from a first order saddle point (the TS). At finite temperatures the reaction does not take a single path and the various dynamic paths may differ considerably from the SDP. Besides the SDP, other representative paths of the dynamic ensemble may be obtained through formulations that maximize the diffusive particle flux [12], or minimize the least-action or least-time of the classical Hamilton-Jacobi equation [59]. In this chapter we focus only on the SDP.

In a mass-weighted Cartesian coordinate system the SDP is known as the intrinsic reaction coordinate (IRC) [13]. This path can be calculated from just the end points, as is done with such algorithms as the nudged elastic band method[17], the string method[21], and the quadratic string method[31]. These methods have been recently modified for large biological systems where QM/MM methods are used and require a significant amount of computational effort per step [23, 24]. While such methods have the advantage of being

parallel, they are less efficient than integration when the TS is known.

The equation for the SDP has the form of an ordinary differential equation (ODE), and is best solved with a numerical ODE method. The many ODE solvers available can be categorized as explicit or implicit, single step or multistep. Multistep solvers [60] use information from past steps and are not studied here. Instead we focus on single-step methods, which only use information available from the current point on the integration path.

Explicit solvers are the most commonly used methods for integration. These methods integrate forward from the current point taking a series of steps that can be combined to approximate the next point to some known order. Implicit ODE solvers, in contrast, use information from the unknown next point and therefore must be solved iteratively starting with an initial guess [61]. As a result, implicit methods are much more computationally costly per step than explicit methods. However, this extra cost allows for a much larger stability region [61]. We give a rigorous definition of the stability region in the next section, but one can think of the stability region as defining how large the step size can be without the integration becoming unstable. When unstable, the result is the well-known behavior of zigzagging about the true path [62, 63].

Whether an ODE remains within a stability region is related to the stiffness of the ODE [64]. That is, the stiffer the problem the larger the stability region must be to ensure the integration does not become unstable. The defining equation for the IRC is known to be a stiff ODE[62], so implicit methods have often been the method of choice [65, 60, 51]. Of the single step methods, the implicit trapezoidal method applied to this problem by Gonzalez-Schlegel (GS) works very well and does not require second order information used in other methods [66, 63, 60]. However, the GS algorithm is only accurate to $2nd$ order and has not been formulated with an error bound. As a result, the error is not known and if higher accuracy is desired, a large number of steps must be taken.

The traits lacking in the GS algorithm are available in explicit methods such as the traditional Runge-Kutta (RK) method. But for stiff equations such as the SDP, explicit

methods perform poorly. Stabilized explicit methods seek to excel in the middle ground between explicit and implicit methods for problems that are mildly stiff. The idea is to yield some of the efficiency of traditional RK methods to extend the stability region by an appropriate amount based on the stiffness of the problem. The net result can be a solver which is able to integrate more efficiently than either implicit or non-stabilized explicit methods. Also these stabilized methods can achieve the high order accuracy and error bounding of RK methods.

Of the many stabilized explicit methods we focus on the one realized in the code DUMKA3 [67]. This will be compared to the well-known fourth order RK method from the code RKSUITE and the GS algorithm. It will be shown that the GS algorithm performs best for low accuracy solutions especially in regions that are highly stiff, such as the area near the minima. DUMKA3 and RKSUITE, by contrast, can perform better for higher accuracy, in the mildly stiff region in the middle of the reaction path. To formulate the optimal algorithm, which performs well in both stiff and non-stiff regions, a simple criterion is developed to switch between an explicit and implicit method. We show in section 2.2 that the final combined algorithm for high accuracy can be almost twice as efficient as either algorithm alone.

In section 2.3 we seek to develop new methods which solve some of the problems associated with the GS algorithm. We are able to develop adaptive, implicit single step methods beyond second order with the diagonally implicit Runge-Kutta (DIRK) framework. It is shown that the stages involved in DIRK methods can each be reduced to the same constrained minimization used in Refs. [51, 65]. These stages can then be combined to give the full step which allows for a higher order result than is possible with single stage methods.

With multiple stages efficient error estimation is possible with the process of embedding. Error estimation allows the step-size to be adaptively adjusted to match the specified tolerance, leading to automatic integration of the reaction path. This can greatly reduce the total number of steps taken by using large steps in smooth, easy-to-follow, parts of the path while ensuring that more rugged parts of the path are followed more closely with

smaller steps. This is in contrast to many implicit ODE solvers [65, 51, 60, 68, 63] for integrating the IRC which do not suggest a form of error control, which means a small step-size is required to ensure good accuracy over the entire path. There are other methods based on following a dynamic reaction pathway[69, 70, 71, 72]. This pathway is close to the IRC and ODE methods are used with error estimation and step-size control[70, 72], however the methods are not implicit.

2.2 Combined Explicit-Implicit Method for Reaction Path Integration

To understand why some methods work best in certain regions of the reaction path and not others we look to expand upon the ideas of stiffness and stability. We will first describe the specific ODE we are interested in and then give a precise definition of stiffness and how it can be measured for the SDP. To make the concepts clear we will look at a simple example which demonstrates how implicit and explicit solvers differ in solving stiff ODEs. Then we will define the stability region for any integration method and give the specific functions for the RK method and the GS algorithm. Next we will examine the general concepts behind stabilized explicit methods and discuss the specific implementation of a stabilized explicit method in DUMKA3. Finally we develop a simple criterion for switching between explicit and implicit methods and give the combined explicit-implicit algorithm.

2.2.1 The Steepest Descent Path

The SDP is described by the ODE,

$$\frac{d\mathbf{x}(s)}{ds} = -\frac{\mathbf{g}(\mathbf{x})}{\|\mathbf{g}(\mathbf{x})\|} \quad (2.1)$$

where $\mathbf{g}(\mathbf{x}) = \nabla V(\mathbf{x})$ is the gradient of the PES, $V(\mathbf{x})$, and the solution $\mathbf{x}(s)$ is parametrized by the arc length, s . Equation (2.1) can be reparameterized to a simpler form[32],

$$\frac{d\mathbf{x}(t)}{dt} = -\mathbf{g}(\mathbf{x}) \quad (2.2)$$

where t and s are related by an additional ODE,

$$\frac{ds}{dt} = \sqrt{\frac{d\mathbf{x}^T}{dt} \frac{d\mathbf{x}}{dt}} \quad (2.3)$$

Integrating Eq. (2.1) with a constant step size gives points equally spaced along the path. Equation (2.3) is a simpler form but often requires significant changes in the step size for regions where the magnitude of $\|\mathbf{g}(\mathbf{x})\|$ changes dramatically.

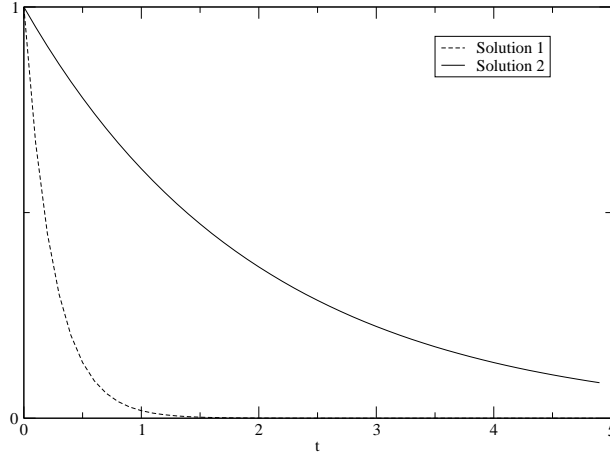


Figure 2.1: Two solutions of an ODE which demonstrate the concept of stiffness. The rapidly varying solution 1 only contributes at the beginning of the integration while the slowly varying solution 2 is more characteristic of the correct numerical integration.

2.2.2 Stiffness

The concept of a stiff ODE is stated by Dekker and Verwer[64] as: “The essence of stiffness is that the solution to be computed is slowly varying but that perturbations exist which are rapidly damped.” This is demonstrated in Fig. 2.1 which is similar to Figure 15.6.1 in Ref. [38]. The two solutions of the ODE are graphed over the integration time variable, t . At the beginning of the integration small steps must be taken to capture the rapidly decaying behavior of solution 1 and the problem is not considered stiff. However, for $t > 1$, solution 1 is merely a rapidly damped perturbation and should not affect the behavior of the overall slowly varying solution which is almost solely made up of solution 2.

A more precise definition of stiffness is given in terms of the linear ODE $\mathbf{y}'(t) = \mathbf{A}\mathbf{y}(t)$, where A is an $n \times n$ matrix with eigenvalues $\lambda_1 \dots \lambda_n$. For real eigenvalues an equation is referred to as stiff by Dekker and Verwer[64] if:

1. there exist $\lambda_k \ll 0$.

2. λ_i exist such that $\lambda_k \ll \lambda_i < 0$.

3. no $\lambda_j \gg 0$ exist.

This definition can also be applied to a nonlinear equation such as Eq. (2.2), by linearizing the right-hand side. Expanding the gradient in Eq. (2.2) about the point \mathbf{x}_0 to first order we have

$$\frac{d\mathbf{x}}{dt} = -\mathbf{g}(\mathbf{x}) = -\mathbf{g}(\mathbf{x}_0) - \mathbf{H}(\mathbf{x} - \mathbf{x}_0) \quad (2.4)$$

where the Hessian is given by $H_{ij} = \frac{\partial^2 V(\mathbf{x})}{\partial x_i \partial x_j}$. If we let $\mathbf{y} = \mathbf{x} - \mathbf{x}_0$ then we obtain,

$$\mathbf{y}' = \mathbf{H}\mathbf{y} \quad (2.5)$$

This is the desired linear equation with $\mathbf{H} = \mathbf{A}$. Therefore, we know that Eq. (2.2) will be locally stiff if the frequencies of the normal modes satisfy the stiff definition above.

The same analysis can also be done for Eq. (2.1). Linearizing Eq. (2.1) requires the derivative,

$$\nabla_{\mathbf{x}} \frac{\mathbf{g}}{\|\mathbf{g}\|} = \frac{\mathbf{H}}{\|\mathbf{g}\|} - \frac{\mathbf{g}(\mathbf{H}\mathbf{g})^T}{\|\mathbf{g}\|^3} \quad (2.6)$$

which again is the linear matrix A that determines the stiffness. This is similar in form to Eq. (2.5) except that the norm of gradient appears in the denominator of the expression. This implies that the stiffness will be exaggerated by the magnitude of the gradient. That is, when the norm of the gradient is small the stiffness is increased; and when the norm of the gradient is large the stiffness is decreased.

2.2.3 Stiff ODE Example

To examine the concept of stiffness and how it relates to the integration scheme used, we focus on a simple two dimensional PES. We assume the PES has a steep valley structure given by,

$$V(x, y) = -\frac{1}{2}ax^2 + \frac{1}{2}by^2 \quad (2.7)$$

where $a, b > 0$ and $b \gg a$. The gradient of Eq. (2.7) is simply, $\nabla V(x, y) = (-ax, by)^T$. Then if given an initial starting point (x_0, y_0) , and if the independent variables are written

in vector form $\mathbf{q}(t) = (x(t), y(t))^T$, the SDP ODE of Eq. (2.7) takes the form

$$\mathbf{q}'(t) = \begin{pmatrix} a & 0 \\ 0 & -b \end{pmatrix} \mathbf{q}(t), \quad \mathbf{q}(0) = \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} \quad (2.8)$$

This is a first order uncoupled homogeneous linear system of equations with constant coefficients. The solution is simply,

$$\mathbf{q}(t) = \begin{pmatrix} e^{at} \\ e^{-bt} \end{pmatrix} \quad (2.9)$$

which is exponentially increasing in x and exponentially decreasing in y . Since we have stated that $b \gg a$, we know the rate of decrease in y is much more rapid than the rate of increase in the x component. This behavior defines Eq. (2.8) as a stiff ODE.

We examine two different methods for solving Eq. (2.8). For the general ODE $y'(t) = f(t, y(t))$ stepping by $\Delta t = h$ the most simple first order integrators are the Euler and implicit Euler methods. These methods are given by,

$$y_{n+1} = y_n + hf(t_n, y_n) \quad (2.10a)$$

and

$$y_{n+1} = y_n + hf(t_{n+1}, y_{n+1}) \quad (2.10b)$$

Applying Eqs. (2.10) to Eq. (2.8) gives,

$$\mathbf{q}(t_{n+1}) = \begin{pmatrix} 1 + ha & 0 \\ 0 & 1 - hb \end{pmatrix} \mathbf{q}(t_n) \quad (2.11a)$$

and

$$\mathbf{q}(t_{n+1}) = \begin{pmatrix} \frac{1}{1-ha} & 0 \\ 0 & \frac{1}{1+hb} \end{pmatrix} \mathbf{q}(t_n), \quad (2.11b)$$

There are two distinct phases for this problem as t increases. In the first phase, when t is small, the e^{-bt} term decays rapidly and must be approximated with a small h in both methods. During this phase Eq. (2.8) is not a stiff equation and similar behavior is expected of the explicit and implicit methods. In the longer time scale, however, e^{-bt} diminishes to

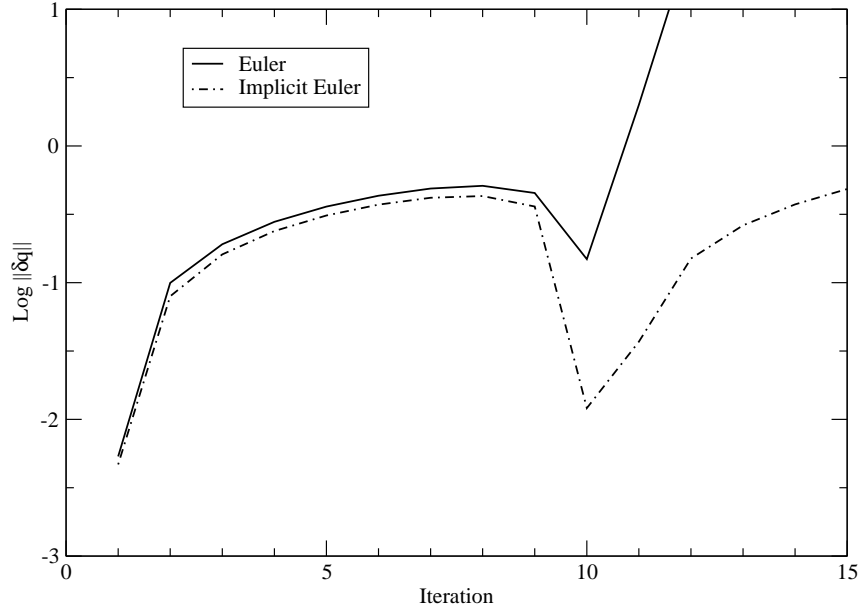


Figure 2.2: Comparison between the Euler and Implicit Euler methods for (a) the error per step with the arc length step fixed at 0.1. and (b) the arc length traveled with the error per step fixed at 0.01. For (a), $\|\delta q\|$ is norm of the closest deviation to the exact path, while in (b), s is the arc length. The PES is given by $V(x, y) = -\frac{1}{2}0.5x^2 + \frac{1}{2}100y^2$, with $(x_0, y_0) = (1, 1)$.

zero and should have no impact on the path. This term is expected to be rapidly damped in this second phase. This is true for the implicit method where the term $\frac{1}{1+hb} < 1$, so that $\lim_{t \rightarrow \infty} y(t) = \lim_{n \rightarrow \infty} (\frac{1}{1+hb})^n \rightarrow 0$. The explicit Euler method, by contrast, only gives stable results for the term $1 - hb$ if $h < \frac{2}{b}$. For large b this can require a smaller h than is necessary for the requested accuracy.

To graphically demonstrate this stiff behavior we choose $a = 0.5$ and $b = 100$ for the PES. On this surface the only stationary point is at $\mathbf{q}(t) = (0, 0)$ which is the TS between two infinitely deep wells. The SDP in this case would be the straight line located on the y -axis. We choose a starting point well off the SDP at $\mathbf{q}(0) = (1, 1)$. The results of applying the two different Euler methods from $\mathbf{q}(0)$ are given in Fig. 2.2. During the first 9 iterations

the implicit and explicit methods give similar performance, since the problem is not stiff and they are of the same order. Once both paths are close to the reaction path on y-axis, about 1 unit away from the start, Eq. (2.8) exhibits its stiff behavior. For the fixed step size 0.1 chosen in Fig. 2.2 (a) the Euler method is unstable and the error grows exponentially due to the term, $1 + hb$. If stability is enforced by bounding the error at 0.01 per step as shown in Fig. 2.2 (b), the Euler method makes very little progress compared to the implicit method. As discussed in the next section, the reason for the stark contrast in performance between these methods rests in the size of their stability regions.

2.2.4 Stability Regions

To examine stability regions we again look to the linear ODE, Eq. (2.5). When used to solve Eq. (2.5) any ODE method has a unique stability function represented by $R(z)$ [73], where in the multidimensional case we write this as $\mathbf{R}(\mathbf{Z})$. Given a stability function, the k^{th} iteration can be written as,

$$\mathbf{y}_{k+1} = \mathbf{R}(h\mathbf{H})\mathbf{y}_k \quad (2.12)$$

In the one dimensional case for the Euler method, Eq. (2.10a), $R(z) = 1 + z$, while for the implicit Euler method, Eq. (2.10b) which uses the derivative at the $k + 1$ step, has $R(z) = \frac{1}{1-z}$.

For the case of the SDP, $\mathbf{R}(\mathbf{Z})$ is a function of the Hessian and therefore a symmetric matrix, which can be diagonalized: $\mathbf{R}(h\mathbf{H}) = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$, where $\mathbf{\Lambda}$ is a diagonal matrix of eigenvalues λ_i and \mathbf{U} is the eigenvector basis. We can then represent the current point at iteration k in terms of the initial point,

$$\mathbf{y}_{k+1} = \mathbf{U}\mathbf{\Lambda}^k\mathbf{U}^T\mathbf{y}_0. \quad (2.13)$$

This can be simplified by rewriting \mathbf{y} in terms of the eigenvector basis, $\bar{\mathbf{y}} = \mathbf{U}^T\mathbf{y}$. Using the identity $\mathbf{U}^T\mathbf{U} = I$, we have

$$\bar{\mathbf{y}}_{k+1} = \mathbf{\Lambda}^k\bar{\mathbf{y}}_0 \quad (2.14)$$

which is clearly only stable if $|\lambda_i| \leq 1$. Given $R(z)$ then, the relation $|\lambda_i| \leq 1$ provides the defining condition for the stability region.

To show how $R(z)$ is obtained we examine the special case of an explicit RK method, which has the form

$$\begin{aligned} Y_i &= y_k + h \sum_{j=1}^{i-1} a_{ij} f(t_k + c_j h, Y_j) \\ y_{k+1} &= y_k + h \sum_{j=1}^s b_j f(t_k + c_j h, Y_j), \end{aligned} \quad (2.15)$$

where the matrix A is strictly lower triangular. As an example, the original fourth order RK method is defined by

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1/2 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \mathbf{b} = \begin{pmatrix} 1/6 \\ 2/6 \\ 2/6 \\ 1/6 \end{pmatrix}, \mathbf{c} = \begin{pmatrix} 0 \\ 1/2 \\ 1/2 \\ 1 \end{pmatrix}. \quad (2.16)$$

If we apply Eq. (2.15) to the linear 1D case, $y' = \lambda y$, then,

$$\begin{aligned} Y_i &= y_k + h\lambda \sum_{j=1}^{i-1} a_{ij} Y_j \\ y_{k+1} &= y_k + h\lambda \sum_{j=1}^s b_j Y_j. \end{aligned} \quad (2.17)$$

Substituting Y_i into y_{k+1} we obtain the desired form $y_{k+1} = R(h\lambda)y_k$ such that,

$$R(z) = 1 + z \sum_{j=1}^s b_j + z^2 \sum_{j=1}^s \sum_{i=1}^s b_j a_{ij} + \dots \quad (2.18)$$

Since the solution of $y' = \lambda y$ is simply $y(t) = e^{\lambda t}$ and that since e^z has the well-known series expansion $1 + z + \frac{z^2}{2!} + \dots$, it is easily proved [73], that in order for the RK polynomial $R(z)$ to be of order $p - 1$ it must also have a form such that,

$$R(z) = 1 + z + \frac{z^2}{2!} + \dots + \mathcal{O}(z^p). \quad (2.19)$$

What we refer to as traditional RK methods all have coefficients such that $p = s + 1$, where s is given by Eq. (2.15) [73]. The stability regions, which are truncated forms of Eq. (2.19),

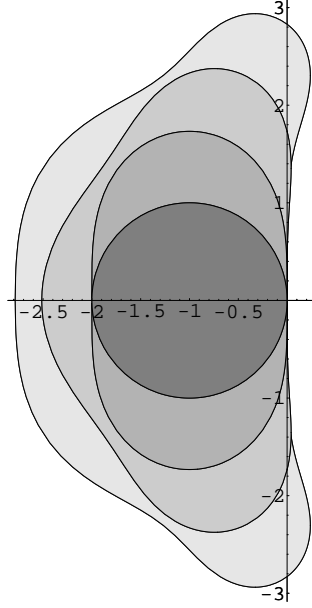


Figure 2.3: Complex Runge-Kutta stability plots for $|R(\lambda h)| \leq 1$, where $R(z)$ is the series in Eq. (2.19) truncated at order $s = p - 1$. Four regions appear from darkest (Euler method, $s=1$) to lightest (fourth order RK method, $s=4$).

are defined by $|R(z)| \leq 1$, and are graphed in the complex plane $h\lambda$ in Fig. 2.3. Since we only deal with the PES here, all the eigenvalues of the Hessian will be real. Our interest then is only in how far the stability region extends along the negative real axis. From Fig. 2.3 and the results of the previous section it is clear that the stability regions of traditional RK methods do not extend very far along the real axis and so are generally inadequate for stiff equations.

Implicit methods have much better stability properties. The implicit Euler method was given in Eq. (2.10b), and the implicit trapezoidal method has a similar form given by,

$$y_{n+1} = y_n + \frac{1}{2}hf(t_n, y_n) + \frac{1}{2}hf(t_{n+1}, y_{n+1}). \quad (2.20)$$

Both Eq. (2.10b) and Eq. (2.20) have the next point y_{n+1} on the right hand side of the equation and thus for a general problem must be solved as a nonlinear set of equations. However, for the SDP, these implicit equations can be simplified to a constrained

minimization of the energy[51, 65].

The stability plots for the implicit Euler and trapezoidal methods are less interesting than the explicit methods since they are stable for all step sizes. Their stability functions are rational functions given by $R(z) = \frac{1}{1-z}$ and $R(z) = \frac{1+z/2}{1-z/2}$ respectively [73]. Plotted for the values which satisfy $|R(z)| \leq 1$, their regions fill up the left half of the complex plane.

2.2.5 Stabilized Explicit Runge-Kutta Integration

The aim of stabilized explicit integration is to extend the stability region along the real negative axis of the complex plane $h\lambda$, while still retaining efficiency and order. Here we only explain the basic idea for first order as done by Hairer and Wanner[73]. See Medovikov[67] for a discussion of higher order schemes.

For a first order algorithm, the main idea is to represent $R(z)$ by the Chebyshev polynomials,

$$R_s(z) = T_s(1 + z/s^2) \quad (2.21)$$

such that,

$$\begin{aligned} T_s(x) &= 2xT_{s-1}(x) - T_{s-2}(x) \\ T_0(x) &= 1 \quad T_1(x) = x. \end{aligned} \quad (2.22)$$

If $R(z)$ takes this form then the region which satisfies $|R(z)| \leq 1$ will extend along the negative real axis so that $-2s^2 \leq z \leq 0$ [73]. The region $|R(z)| \leq 1$ is plotted in Fig. 2.4 for $s = 2$ and $s = 3$. So depending on the stiffness of the problem the stability region can be extended by increasing the order of the Chebyshev polynomial used.

In order to find an RK method corresponding to the stability region given by Eq. (2.21) the polynomial $R_s(x)$ is factored into its roots $z_{i=1\dots s}$ so that,

$$R_s(z) = (1 - z/z_1)(1 - z/z_2) \dots (1 - z/z_s). \quad (2.23)$$

Since an Euler step has a stability region $R(h\frac{1}{z_i}) = 1 + h\frac{1}{z_i}$, the region given by Eq. (2.23) can be formed by taking the series of Euler steps $Y_i = Y_{i-1} + h\frac{1}{z_i}f(Y_{i-1})$ for $i = 1 \dots s$. So

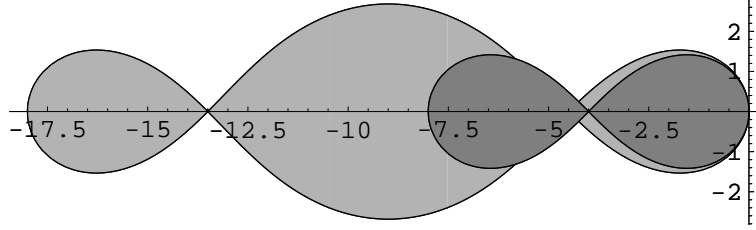


Figure 2.4: Complex plot of the Chebyshev based functions for $|R(\lambda h)| \leq 1$. Darker region is $s = 2$ and the lighter is $s = 3$.

although the stability region can be expanded to any desired length, this comes at the cost of increasing the number of required Euler steps.

While stabilized methods can be effective, there are a number of problems with this simple case of taking Euler steps. These problems are dealt with in the literature [73] and are summarized here along with their solution. First, there are $s - 1$ points on the negative real axis where $|R(z)| = 1$. This is undesirable since there is no damping at these points. This problem can be remedied by altering the polynomial into a rational function. Second, it is not clear how to order the roots for the series of Euler steps. Deciding how to do this has been addressed by minimizing round off error. Third, some of the roots can be quite small leading to large inaccurate Euler steps. This problem has been solved by grouping roots together in a recursion formula.

The last and most obvious problem with this method is that Eq. (2.23) only matches Eq. (2.19) up to first order. It is this issue that has been addressed and implemented in the codes RKC (2^{nd} order) and DUMKA3 (3^{rd} order) [74, 67]. Since our interest is in high accuracy, we examine only the code DUMKA3 which uses a modified set of polynomials that agree with Eq. (2.19) up to 3^{rd} order.

DUMKA3 solves the general ODE, $\frac{d\mathbf{y}}{dt} = \mathbf{f}(t, \mathbf{y})$. As such, it requires the function $\mathbf{f}(t, \mathbf{y})$ and an estimate of the spectral radius, $\rho(\mathbf{J}) = \max_i \lambda_i$, where $\mathbf{J} = \nabla_{\mathbf{y}} \mathbf{f}(t, \mathbf{y})$ is the Jacobian and λ_i are the eigenvalues of \mathbf{J} .

The spectral radius is needed to determine the order of the polynomial s , which as

discussed earlier is related to the extent of the stability region. In the case of the SDP $\mathbf{f}(t, \mathbf{y}) = -\mathbf{g}/\|\mathbf{g}\|$, and $\nabla_{\mathbf{y}}\mathbf{f}(t, \mathbf{y})$ is given by Eq. (2.6). Calculating Eq. (2.6) exactly, when needed by DUMKA3, is impractical since this would imply at least one evaluation of the Hessian per time step. However, since we are only interested in the spectral radius of the Hessian we can obtain a good estimate by approximating \mathbf{H} in Eq. (2.6) and including a safety factor $\epsilon > 1$ so that $\rho_{DUMKA} = \epsilon\rho(\nabla_{\mathbf{y}}\mathbf{f}(t, \mathbf{y}))$.

The exact Hessian is available at the TS since it is required to calculate the initial direction. This Hessian can either be updated by a Quasi-Newton method during the integration, such as the Broyden-Fletcher-Goldfarb-Shanno (BFGS) [37] update, or it can be assumed constant.

2.2.6 Combined Explicit-Implicit Method

To combine methods, it is important to note that the spectral radius of Eq. (2.6) is inversely proportional to $\|\mathbf{g}\|$. This implies the problem becomes much more stiff in areas where $\|\mathbf{g}\|$ is small. For the traditional RK method this means using smaller step sizes while for DUMKA more stages must be used to expand the stability region. In both cases this implies a higher computational cost. In these highly stiff areas it therefore makes sense to use the GS algorithm instead of explicit methods. As is shown later in Fig. 2.5, these stiff regions occur at the beginning and end of the reaction path.

Determining when to switch between an implicit and explicit method can be done practically by comparing the ratio $\frac{h_{expl}}{N_{expl}}$ to $\frac{h_{impl}}{N_{impl}}$, where N is the number of explicit (expl) or implicit (impl) evaluations per step of size h . Usually N is known in advance for methods, but for a given tolerance the allowed step size can vary based on the size of the error term.

For a p th order method the exact integration step from \mathbf{y}_n to $\mathbf{y}(x+h)$ is approximated by

$$\mathbf{y}(x+h) = \mathbf{y}_{n+1} + Ch^{p+1}\phi + \mathcal{O}(h^{p+2}) \quad (2.24)$$

where C is the error constant related to the integration method and ϕ is a consolidated term [61] dependent on the derivatives of the function $\mathbf{y}(\mathbf{x})$. For a desired tolerance tol

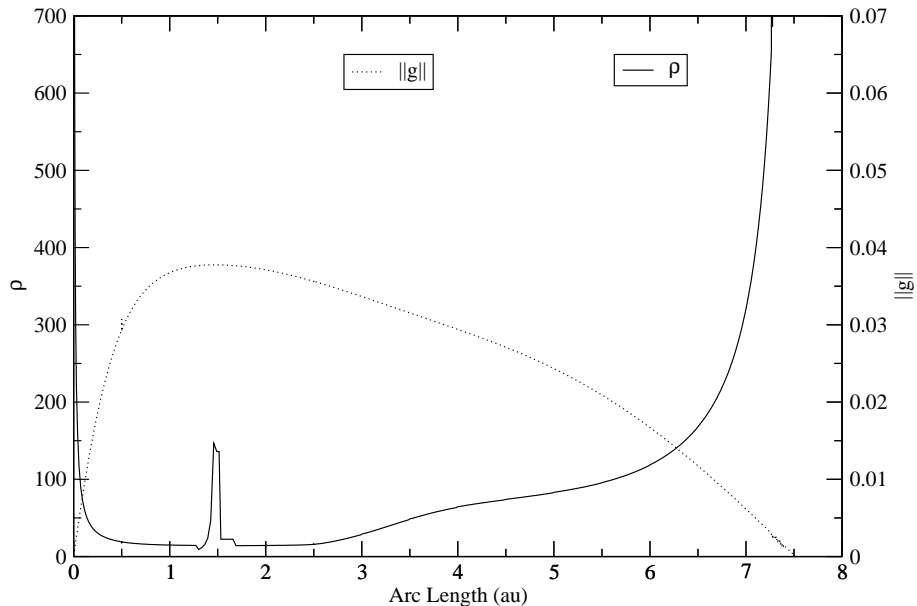


Figure 2.5: Spectral radius of Eq. (2.6) over the reaction $SiH_2 + H_2 \rightarrow SiH_4$. The Hessian is updated with BFGS from the exact Hessian given at the TS.

then, the next step size would be calculated as $h = (\frac{tol}{C\phi})^{1/(p+1)}$. This ignores stability considerations though. If the extent of the stability region along the negative x-axis is α , then to stay within the stability region and to satisfy the desired tolerance we must have,

$$h = \min \left(\frac{\alpha}{\rho(\nabla_{\mathbf{y}} \mathbf{f}(t, \mathbf{y}))}, \left(\frac{tol}{C\phi} \right)^{1/(p+1)} \right) \quad (2.25)$$

Unfortunately, the error ϕ is not calculated in the GS algorithm. Also, to compare two different methods the order would have to be the same, which is not the case for the methods used here. To circumvent this problem, at the beginning of the integration we do an explicit calculation every k steps of the implicit method. Rather than compare ratios in this case we can simply choose $h_{expl} = N_{expl} h_{impl} / N_{impl}$, and then switch to the Explicit method if the integration does not fail, i.e. the estimated error is less than the set tolerance. For the end region, the ratio h_{impl} / N_{impl} is assumed to be constant and can be set equal to the average value from the starting region. For the explicit method, h_{expl} / N_{expl} can simply be calculated after each step.

2.3 Diagonal Implicit Runge-Kutta Methods for Reaction Path Integration

In this section we first examine the basic SDP ODE which is the focus of the methods presented here. Next we give the definitions of three common stability properties: A,L, and strongly-S(SS). Then we examine the general theory of DIRK methods and survey various methods which do not have intrinsic error estimation. This is followed by a review of embedded methods that are constructed to allow for error estimation without additional calculations. Finally we derive further embedded methods and summarize all the methods presented.

2.3.1 Stability Measures

Although a method may be able to achieve a high order, it is of little use for stiff problems such as the SDP if it does not have good stability properties[75]. In this subsection we examine the requirements necessary for a method to achieve the range of possible stabilities.

To examine the stability of a method the linear model is used: $y'(t) = \lambda y(t)$, where λ is a complex number. For this equation all Runge-Kutta methods can be written in the form [64],

$$y_{n+1} = R(h\lambda)y_n, \tag{2.26}$$

where h is the step size and y_n is the numerical solution at step $t = hn$. The function here, $R(h\lambda)$ uniquely defines the Runge-Kutta method and is referred to as the stability function.

The exact solution of the linear model at $t_n = nh$ is of course, $y_n = e^{\lambda nh}y(0)$. Equation (2.26) can also be written in this form as $y_n = R(h\lambda)^n y(0)$. If the Runge-Kutta(RK) method is to be of order p then the Taylor expansion of $R(z)$, where $z = h\lambda$, must match the Taylor series of e^z up to the p th order term. Also since e^z is the exact answer we expect $R(z)$ to decay on the region $z < 0$. This implies that for $z < 0$, $\|R(z)\| \leq 1$. If this is not true then for a large enough step size any numerical solution will grow without bound.

In Ref. [75] plots of the region $\|R(z)\| \leq 1$ were given for the traditional Runge-Kutta methods and the method DUMKA. These were explicit methods and the regions did not extend very far along the negative real axis.

So far we have assumed that $y(t)$ is a scalar function. For a multidimensional system \mathbf{y}_n is a vector and $\mathbf{R}(z)$ becomes a matrix function. In this case we define the relation $\|\mathbf{R}(z)\| \leq 1$ as: $\forall u_i$ such that u_i is an eigenvalue of $R(z)$, $|u_i| \leq 1$. [75] Given this more general definition of the norm we have the following definition [73]:

Definition 1. *The stability function $R(z)$ is referred to as A-stable if $\|R(z)\| \leq 1 \forall \text{Re}(z) \leq 0$, where $z \in \mathbb{C}$.*

This is actually a more restrictive definition than we need to use, since $R(z)$ will be a polynomial function of \mathbf{H} , where $\mathbf{H}_{ij} = \frac{\partial^2 V(\mathbf{x})}{\partial \mathbf{x}_i \partial \mathbf{x}_j}$. For a classical PES all the eigenvalues of \mathbf{H} are real and thus so are those of $R(z)$. If we restrict the A-stable definition to $\text{Re}(z) < 0$ we then have [64]:

Definition 2. *The stability function $R(z)$ is referred to as A-acceptable if $\|R(z)\| \leq 1 \forall z \leq 0$, where $z \in \mathbb{R}$.*

All A-stable methods are obviously also A-acceptable.

Although A-stable methods give the correct decaying behavior, nothing is stated about their rate of decay. So while it may be true for a method that $R(z) \leq 1$, for all z . At large z , $R(z)$ may remain very close to 1. To avoid this undesirable property a stricter definition is often used forcing $R(z)$ to zero. AN-stable and L-stable methods use this stricter definition. AN-stability is defined in Ref. [76] and L-stability is defined as,

Definition 3. *The stability function $R(z)$ is referred to as L-stable if $\lim_{z \rightarrow \infty} R(z) = 0$*

Another even stricter version can be used which sets the rate at which $R(z)$ goes to zero [77].

Definition 4. *A Runge-Kutta method which solves the equation $y'(t) = g'(t) + \lambda(y - g(t))$ for any bounded function $g(t)$ and its bounded derivative $g'(t)$ is referred to as strongly S-*

stable if $\frac{y_{n+1}-g(t_{n+1})}{y_n-g(t_n)} \rightarrow 0$ as $Re(-\lambda) \rightarrow \infty$ for all h such that $y_n = y(t_n)$ and $t_{n+1} = t_n + h$.

Enforcing stricter stability conditions usually requires more stages and thus is significantly more expensive. In the next two subsections we examine specific methods which have these stability properties.

2.3.2 Solving Diagonal Implicit Runge-Kutta Equations

For the general initial value problem,

$$\frac{d\mathbf{y}}{dt} = \mathbf{f}(t, \mathbf{y}), \quad \mathbf{y}(0) = \mathbf{y}_0, \quad (2.27)$$

all Runge-Kutta methods with s intermediate stages take the form,

$$\mathbf{y}_{n+1} = \mathbf{y}_n + h \sum_{i=1}^s b_i \mathbf{f}(t_n + hc_i, \mathbf{Y}_i) \quad (2.28a)$$

such that,

$$\mathbf{Y}_i = \mathbf{y}_n + h \sum_{j=1}^s a_{ij} \mathbf{f}(t_n + hc_i, \mathbf{Y}_j). \quad (2.28b)$$

For a given defining \mathbf{A} and \mathbf{b} , a method is referred to as consistent if

$$c_i = \sum_j^s a_{ij}. \quad (2.28c)$$

To write this in a compact way, numerical ODE texts[64, 73] use the Butcher array,

$$\begin{array}{c|c} \mathbf{c} & \mathbf{A} \\ \hline & \mathbf{b}^T \end{array} \quad (2.29)$$

to represent a Runge-Kutta method. For explicit methods the $s \times s$ matrix \mathbf{A} is strictly lower triangular. As a result, each stage is calculated from a previous stage and the number of function evaluations is known in advance. For implicit methods there is at least one non-zero element of \mathbf{A} , on or above the diagonal. In this case the s equations of (2.28b) must be solved as a nonlinear set of equations.

Using a nonlinear solver on Eq. (2.28b) is costly. However, solving the entire system of nonlinear equations may be avoided if each stage can be solved independently. This implies that the i th stage takes the form,

$$\mathbf{Y}_i = \mathbf{Y}_i^* + \alpha f(\mathbf{Y}_i), \quad (2.30)$$

where α and \mathbf{Y}_i^* are constant and for the SDP, $f(\mathbf{Y}_i) = g(\mathbf{Y}_i)/\|g(\mathbf{Y}_i)\|$. Equation (2.30) can be rearranged to, $\mathbf{Y}_i - \mathbf{Y}_i^* = \alpha f(\mathbf{Y}_i)$ and then by taking the norm of both sides we obtain,

$$\|\mathbf{Y}_i - \mathbf{Y}_i^*\| = \alpha, \quad (2.31)$$

since $\|f(\mathbf{Y}_i)\| = \|g(\mathbf{Y}_i)/\|g(\mathbf{Y}_i)\|\| = 1$. From Eq. (2.31) it is evident that the solution \mathbf{Y}_i , must lie on the hypersphere centered at \mathbf{Y}_i^* with a radius of α . If we minimize the energy on this hypersphere then we obtain the optimization problem,

$$\begin{aligned} \min_{\mathbf{Y}_i} V(\mathbf{Y}_i) \\ \|\mathbf{Y}_i - \mathbf{Y}_i^*\|^2 = \alpha^2. \end{aligned} \quad (2.32)$$

At the solution of this optimization problem, $2(\mathbf{Y}_i - \mathbf{Y}_i^*) = \lambda g(\mathbf{Y}_i)$, where λ is the lagrange multiplier. Comparing this solution to Eq. (2.30) we see $\lambda = 2\alpha/\|g(\mathbf{Y}_i)\|$.

Having shown that Eq. (2.30) is equivalent to the simple constrained minimization in Eq. (2.32) it is desirable to find RK methods where the stages, given by Eq. (2.28a), have this form. Such forms have in fact already been used but only for one implicit stage. Müller and Brown[51] used the first order implicit Euler equation given by $\mathbf{y}_{n+1} = \mathbf{y}_n + h\mathbf{f}(\mathbf{y}_{n+1})$ where h is the step size. This is represented by the Butcher array,

$$\begin{array}{c|c} 1 & 1 \\ \hline & 1 \end{array} \quad (2.33)$$

In this case the minimization would be done with $\alpha = h$ and $\mathbf{Y}_1^* = \mathbf{y}_n$. Similarly Gonzalez and Schlegel[65] used the second order implicit trapezoidal rule of which has the form, $\mathbf{y}_{n+1} = \mathbf{y}_n + 1/2h\mathbf{f}(\mathbf{y}_n) + 1/2h\mathbf{f}(\mathbf{y}_{n+1})$. The Butcher array for this method is,

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & \frac{1}{2} & \frac{1}{2} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array} \quad (2.34)$$

Here, $\alpha = 1/2h$ and $\mathbf{Y}_1^* = \mathbf{y}_n + 1/2h\mathbf{f}(\mathbf{y}_n)$.

Both the implicit Euler and trapezoidal method can be reduced to a single stage and consequently the order is limited to two [77]. More stages can be added though, so long as each stage has the form,

$$\mathbf{Y}_i = \mathbf{y}_n + h \sum_{j=1}^{i-1} a_{ij} \mathbf{f}(t_j, \mathbf{Y}_j) + ha_{ii} \mathbf{f}(t_i, \mathbf{Y}_i). \quad (2.35)$$

In this form, each stage can be solved with Eq. (2.32) using $\alpha = ha_{ii}$ and $\mathbf{Y}_i^* = \mathbf{y}_n + h \sum_{j=1}^{i-1} a_{ij} \mathbf{f}(t_j, \mathbf{Y}_j)$. ODE solvers with stages given by Eq. (2.35) are known as diagonally implicit Runge-Kutta (DIRK) methods. If $a_{ii} = \gamma$ for constant γ in Eq. (2.35), then the method is referred to as a singly diagonally implicit Runge-Kutta (SDIRK) method. Furthermore if a SDIRK method has $a_{11} = 0$ and $a_{i \neq 1, i \neq 1} = \gamma$, so that the first stage is explicit, then it is known as an explicit SDIRK (ESDIRK) method.[78] With these labels the implicit Euler method is both a DIRK and a SDIRK method, while the trapezoidal method is an ESDIRK method.

2.3.3 Survey of Non-embedded Methods

For the purposes of reaction path integration we are interested in surveying DIRK methods without any constraints on the diagonal elements of Eq. (2.35). However SDIRK methods offer a significant computational advantage in many applications because the LU-factorization of the linear set of equations formed at each iteration may be reused. This allows for rapid evaluation when the function is trivial to evaluate. Generally function evaluations of chemical systems are non-trivial and so this restriction is not desirable. Nevertheless, in the literature[79, 80, 81, 82, 77, 78, 83, 84, 85, 86] SDIRK methods are largely the only ones considered because of the factorization benefits and so we review only this subclass of methods.

To label methods the pair (s,p) is used to denote the stages and order respectively. For the (1,2) case there is only one method possible,[77]. This is the implicit midpoint (IM)

rule,

$$\begin{array}{c|c} 1/2 & 1/2 \\ \hline & 1 \end{array} \quad (2.36)$$

which has the same order as the trapezoidal rule, Eq. (2.34). It also requires the same number of evaluations since the the gradient evaluation on the last stage of Eq. (2.34) is the same required by the first stage of the next step.

There are two (2,3) SDIRK methods developed by Nørsett [87] which can be found in Ref. [64]. These are given by the Butcher array,

$$\begin{array}{c|cc} \gamma & \gamma & 0 \\ 1-\gamma & 1-2\gamma & \gamma \\ \hline & 1/2 & 1/2 \end{array} \quad (2.37)$$

where $\gamma = \gamma_{\pm 1} = 1/2 \pm \sqrt{3}/6$. However the method is A-stable only if $\gamma = \gamma_1$. The stability plot[75] for $\gamma = \gamma_{-1}$ is shown in Fig. 2.3.3. We refer to Eq. 2.37 as method N1 using the convention of the first letter of last name of each author followed by an incrementing number since many authors have more than one method. The exception is the implicit trapezoidal rule used by Gonzalez and Schlegel[65] which we refer to as GS2 to match the notation found elsewhere in the literature.

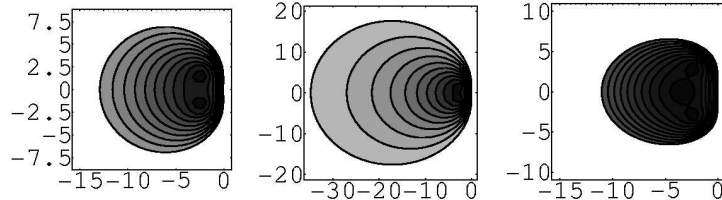


Figure 2.6: Stability plots for the 2 stage SDIRK method with $\gamma = \gamma_{-1}$, and 3 stage SDIRK methods with $\gamma = \gamma_1$ and $\gamma = \gamma_{-1}$, respectively. Contour lines enclose the darkest area where $|R(z)| \rightarrow 0$ to the white area where $|R(z)| > 1$.

For the (3,4) case Nørsett [87, 64] showed there are three possible methods which have

the form,

$$\begin{array}{c|ccc}
 \gamma & \gamma & 0 & 0 \\
 1/2 & 1/2 - \gamma & \gamma & 0 \\
 1 - \gamma & 2\gamma & 1 - 4\gamma & \gamma \\
 \hline
 & 1/(24(1/2 - \gamma)^2) & 1 - 1/(12(1/2 - \gamma)^2) & 1/(24(1/2 - \gamma)^2)
 \end{array} \tag{2.38}$$

where γ is $\gamma_0 = 2 \cos(\pi/18)/\sqrt{3}$ or $\gamma_{\pm 1} = 1/2 - 1/6\sqrt{3} \cos(\pi/18) \pm 1/2 \sin(\pi/18)$. This will be referred to as the N2 method. Equation (2.38) is only stable for $\gamma = \gamma_0$. The stability plots for $\gamma_{\pm 1}$ are shown in Fig. 2.3.3. There are no (4,5) SDIRK methods[77], but A-stable (5,5) and (6,6) methods are given by Cooper and Sayfy[83].

To obtain extra stability beyond A-stability, further stages must be added for a given order. For example Alexander[77] has two SS-stable methods. His (2,2) method A1 is,

$$\begin{array}{c|cc}
 \gamma & \gamma & 0 \\
 1 & 1 - \gamma & \gamma \\
 \hline
 & 1 - \gamma & \gamma
 \end{array} \tag{2.39}$$

where $\gamma = 1 - 1/\sqrt{2}$ and his (3,3) method A2 is,

$$\begin{array}{c|ccc}
 \gamma & \gamma & 0 & 0 \\
 (1 + \gamma)/2 & (1 - \gamma)/2 & \gamma & 0 \\
 1 & b_2 & b_2 & \gamma \\
 \hline
 & b_2 & b_2 & \gamma
 \end{array} \tag{2.40}$$

where $\gamma = 0.435866521508$, $b_1 = -(6\gamma^2 - 16\gamma + 1)/4$ and $b_2 = (6\gamma^2 - 20\gamma + 5)/4$. Similar to the A-stable case there is no (4,4) SS-stable method.[77]

To obtain higher order methods it appears advantageous to add additional explicit stages to form an ESDIRK method. In particular for the last stage if $a_{sj} = b_j$ an extra explicit step can be added at the beginning without the need for any further function eval-

uation. Such methods have been constructed by Cooper and Sayfy[83] with the structure,

$$\begin{array}{c|ccc}
 0 & 0 & 0 & 0 \\
 c_2 & a_{2,1} & a_{2,2} & 0 \\
 1 & b_1 & b_2 & b_3 \\
 \hline
 & b_2 & b_2 & \gamma
 \end{array} \tag{2.41}$$

With A-stability and $p = 3$ the only difference between this method and the method N1 is the deviation from the correct fourth order Taylor series coefficients. If, for method N1, we examine the norm of difference to the correct coefficients we find it is 0.126967. For the ESDIRK method (2.41), with the added constraint $a_{2,2} = b_3$ the norm of the difference was calculated to be 0.283907. If this constraint is removed the difference drops to 0.266524. Clearly this form does not offer any advantage in terms of reducing the local error. Later though we will see that the extra stage can be useful in embedded methods because it allows for more flexibility in constructing the error term.

2.3.4 Review of Embedded Methods

While the methods given in the last section can be used without error estimates, it is usually best to use some form of error estimation so the step size can be adjusted based on the accuracy desired. To this end one can use step-halving[77, 88], or the difference between methods of order p and $p \pm 1$. Although these forms of error estimation work well, and may be used by any Runge-Kutta method, in this case they more than double the number of minimizations. To circumvent this problem we look to methods which use embedding to achieve automatic integration[89, 88].

Embedding is the process of using two different combinations of RK stages to obtain methods which differ in order by one. So if we have a p th order method with vector \mathbf{b} ,

$$\mathbf{y}_{n+1}^p = \mathbf{y}_n + h \sum_i^s b_i \mathbf{f}(t_i, \mathbf{Y}_i), \tag{2.42a}$$

we would seek to find a $\hat{\mathbf{b}}$ such that

$$\mathbf{y}_{n+1}^{p-1} = \mathbf{y}_n + h \sum_i^s \hat{b}_i \mathbf{f}(t_i, \mathbf{Y}_i) \tag{2.42b}$$

is of order $p - 1$. With \mathbf{y}_{n+1}^{p-1} the error would be estimated as,

$$\Delta_0 = \mathbf{y}_{n+1}^p - \mathbf{y}_{n+1}^{p-1} = \sum_{i=1}^s (b_i - \hat{b}_i) \mathbf{f}(t_i, \mathbf{Y}_i), \quad (2.43)$$

with no additional calculations necessary. Here the error only applies to the lower order step, this is in contrast to step doubling where the error is of the same order as the method itself. For some methods it is possible to find $\hat{\mathbf{b}}$ such that the error is of order p . However this may lead to an underestimate of the error.

The Butcher array for embedded methods is augmented to,

$$\begin{array}{c|c} \mathbf{c} & \mathbf{A} \\ \hline & \mathbf{b}^T \\ \hline & \hat{\mathbf{b}}^T \end{array} \quad (2.44)$$

The first embedded methods we examine are reported by Cash in Ref. [86], where the author gives two SS-stable embedded methods. The (3,3) method C1 is,

$$\begin{array}{c|ccc} 0.43586652 & 0.43586652 & 0 & 0 \\ 0.71793326 & 0.28206674 & 0.43586652 & 0 \\ 1 & 1.2084966 & -0.64436317 & 0.43586652 \\ \hline & 1.2084966 & -0.64436317 & 0.43586652 \\ \hline & 0.77263013 & 0.22736987 & 0 \end{array} \quad (2.45)$$

and (5,4) method C2 is,

$$\begin{array}{c|ccccc} 0.43586652 & 0.43586652 & 0 & 0 & 0 & 0 \\ 1.571733 & 1.1358665 & 0.43586652 & 0 & 0 & 0 \\ 0.8 & 1.0854333 & -0.72129983 & 0.43586652 & 0 & 0 \\ 0.92455676 & 0.4163495 & 0.190984 & -0.11864327 & 0.43586652 & 0 \\ 1 & 0.89686965 & 0.018272527 & -0.084590031 & -0.26641867 & 0.43586652 \\ \hline & 0.89686965 & 0.018272527 & -0.084590031 & -0.26641867 & 0.43586652 \\ \hline & 0.77669193 & 0.029747279 & -0.026744024 & 0.22030481 & 0 \end{array} \quad (2.46)$$

The lower order step in these embedded methods is constructed by setting an element of the vector $\hat{\mathbf{b}}$ equal to zero and then minimizing its error term. This technique is also used by Haire and Wanner in Ref. [73] to construct a different (5, 4) method HW1 by enforcing L-stability rather than the SS-stability requirement. This L-stable (5, 4) method reported with $\gamma = 1/4$ gives “especially nice rational coefficients”, but the authors suggest using a better value of $\gamma = 4/15$. Following this advise the Butcher array is rederived as,

4/15	4/15	0	0	0	0	
23/30	1/2	4/15	0	0	0	
17/30	0.35415395	-0.054153953	4/15	0	0	
0.36613154	0.085154941	-0.064843323	0.079153253	4/15	0	(2.47)
1	2.1001157	-0.76778003	2.3998164	-2.9988187	4/15	
	2.1001157	-0.76778003	2.3998164	-2.9988187	4/15	
	2.8852642	-0.14587935	2.3900087	-4.1293935	0	

In Ref. [85] Nørsett, Syvert and Thomsen derived methods with AN-stability. One of these methods uses an explicit last stage to extend the order to four with four implicit stages. This (5,4) NST1 method is,

5/6	5/6	0	0	0	0	
29/108	-61/108	5/6	0	0	0	
1/6	-23/183	-33/61	5/6	0	0	
1/24	3464669/4002576	-1857345/917257	88445/240592	5/6	0	
7/8	457/488	-15/244	0	0	0	
	-398/1159	9258300/4307149	-73/33	10936/13965	408/655	
	26/61	324/671	1/11	0	0	(2.48)

There are also two other methods given in Ref. [85] which are only A-stable. The (4,4)

NST2 method is,

$$\begin{array}{c|cccc}
 5/6 & 5/6 & 0 & 0 & 0 \\
 29/108 & -15/26 & 5/6 & 0 & 0 \\
 0 & 215/54 & -130/27 & 5/6 & 0 \\
 1/6 & 4007/6075 & -31031/24300 & -133/2700 & 5/6 \\
 \hline
 & 32/75 & 169/300 & 1/100 & 0 \\
 \hline
 & 61/150 & 2197/2100 & 19/100 & -9/14
 \end{array} \tag{2.49}$$

and the lower order (3,3) NST3 method is,

$$\begin{array}{c|ccc}
 5/6 & 5/6 & 0 & 0 \\
 29/108 & -61/108 & 5/6 & 0 \\
 1/6 & -23/183 & -33/61 & 5/6 \\
 \hline
 & 26/61 & 324/671 & 1/11 \\
 \hline
 & 25/61 & 36/61 & 0
 \end{array} \tag{2.50}$$

ESDIRK methods are given in Refs. [84, 78, 82]. However in Ref. [82] the methods are not embedded and also lack A-stability, while in the methods in Ref. [78] require different γ for the lower order step and thus are not fully embedded. In Ref. [84] Butcher and Chen gave a useful (6, 4) L-stable method. This method, referred to BC1, does not use a lower order step to approximate the error and instead relies on the difference between the fifth order terms. The Butcher array is,

$$\begin{array}{c|cccccc}
 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 1/2 & 1/4 & 1/4 & 0 & 0 & 0 & 0 \\
 1/4 & 1/16 & -1/16 & 1/4 & 0 & 0 & 0 \\
 1/2 & -7/36 & -4/9 & 8/9 & 1/4 & 0 & 0 \\
 3/4 & -5/48 & -257/768 & 5/6 & 27/256 & 1/4 & 0 \\
 1 & 1/4 & 2/3 & -1/3 & 1/2 & -1/3 & 1/4 \\
 \hline
 & 1/4 & 2/3 & -1/3 & 1/2 & -1/3 & 1/4 \\
 \hline
 & 7/90 & 3/20 & 16/45 & -1/60 & 16/45 & 7/90
 \end{array} \tag{2.51}$$

2.3.5 New Specialized Embedded Methods for Reaction Path Integration

In this section we seek to construct embedded A-stable Runge-Kutta methods which are tailored to solving the SDP ODE. With these methods we can release the SDIRK constraint on diagonal elements and experiment with the spacing of stages. This can potentially result in deriving more efficient solvers for the SDP problem, than are available in the literature. Here we limit ourselves to developing methods which have no more than three implicit stages.

For any Runge-Kutta method which is of order p , by definition it must match the Taylor expansion of the next point up to the p order term. The Taylor series expanded about point y_n is given by,

$$y_{n+1} = y_n + hy'_n + \frac{h^2}{2}y''_n + \dots \quad (2.52)$$

such that each derivative can be expanded as[61]

$$\begin{aligned} y'_n &= f \\ y''_n &= f_t + f_y f \\ y'''_n &= f_{tt} + 2f_{ty}f + f_y f_t + f_{yy}f^2 + f_y^2 f \end{aligned} \quad (2.53)$$

where $f = f(t, y)$ and partial derivatives take the form $f_{ty} = \frac{\partial^2 f(t, y)}{\partial t \partial y}$. When Eq. (2.28) is expanded as a Taylor series the coefficients can be matched up to the exact series in Eq. (2.52). The result is the order conditions given in Table 2.3.5. [73] For a given number of stages s and a fixed pattern of A these polynomial equations can be solved up to a given order to obtain an integration method. If there are any leftover terms these can be used to either improve the stability of the method or minimize the error of higher order terms in the Taylor series.

The conditions for A-stability can be derived from the stability function, which for a DIRK method is given by,

$$R(z) = \frac{\det(\mathbf{I} - z\mathbf{A} + z\mathbf{e}\mathbf{b}^T)}{\det(\mathbf{I} - z\mathbf{A})} \quad (2.54)$$

where $e^T = (1, 1, \dots, 1)$. Since $R(z)$ is clearly a rational function it can be written as $R(z) = P(z)/Q(z)$. As stated in Definition 1, A-stability requires that $|R(y)| \leq 1$ for all complex y . It is difficult to derive methods from this condition, so the following result from Ref. [83] is used,

Theorem 1. *An $s - 1$ stage DIRK method of order p is A-stable if and only if $b_{rr} \geq 0$, $r = 1, 2, \dots, s$, and*

$$\sum_{r=\lceil(p+2)/2\rceil}^s y^r \sum_{j=0}^r *(\beta_{2r-j}\beta_j - \alpha_{2r-j}\alpha_j)(-1)^{r+j} \geq 0 \quad \forall y \geq 0. \quad (2.55)$$

In this theorem ' $\lceil \cdot \rceil$ ' is the integer part of the expression and the '*' symbol means that any term with $2j = r$ is divided by two. The expressions for β_j and α_j are defined by,

$$P(z) = 1 - \alpha_1 z + \alpha_2 z^2 \dots + (-1)^{s-1} \alpha_{s-1} z^{s-1} \quad (2.56a)$$

and,

$$Q(z) = 1 - \beta_1 z + \beta_2 z^2 \dots + (-1)^s \beta_s z^s. \quad (2.56b)$$

order	condition
1	$\sum_j b_j = 1$
2	$\sum_{jk} b_j a_{jk} = \frac{1}{2}$
3	$\sum_{jkl} b_j a_{jk} a_{jl} = \frac{1}{3}$
3	$\sum_{jkl} b_j a_{jk} a_{kl} = \frac{1}{6}$
4	$\sum_{jklm} b_j a_{jk} a_{jl} a_{jm} = \frac{1}{4}$
4	$\sum_{jklm} b_j a_{jk} a_{kl} a_{jm} = \frac{1}{8}$
4	$\sum_{jklm} b_j a_{jk} a_{kl} a_{km} = \frac{1}{12}$
4	$\sum_{jklm} b_j a_{jk} a_{kl} a_{lm} = \frac{1}{24}$

Table 2.1: Order conditions for DIRK methods

Here the convention of Ref. [83] is followed where it is assumed that vector \mathbf{b} is the last stage. The following result is also used, which can be obtained by matching the p orders terms from $R(z)$ to the exact linear expansion,

$$\alpha_r = \beta_r + \beta_{r-1} + \frac{1}{2}\beta_{r-2} \dots (-1)^r \frac{1}{r!}, \quad r = 1, 2, \dots, p. \quad (2.57)$$

With these additional conditions we can solve for the necessary Butcher array coefficients.

In the following, we focus only on methods with an explicit first stage since the extra explicit stage can be used for the error estimate. Also, as noted earlier, if $c_s = 1$ then the explicit first stage can be calculated without any additional function evaluations. For the simplest form, with only one implicit stage, the optimal method is the implicit trapezoidal rule. To this we can easily add a first order error term,

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & \frac{1}{2} & \frac{1}{2} \\ \hline & \frac{1}{2} & \frac{1}{2} \\ \hline & 0 & 1 \end{array} \quad (2.58)$$

to arrive at method BY1. Since the SDP ODE has no dependence on t we write the local error estimate for method (2.34) as $\hat{\mathbf{l}} = \mathbf{y}_{n+1} - \hat{\mathbf{y}}_{n+1} = 1/2 f f_y$. and the local error as $\mathbf{l} = \mathbf{y}(h(n+1)) - \mathbf{y}_{n+1} = 1/12 \mathbf{f} \mathbf{f}_y^2 + 1/12 \mathbf{f}^2 \mathbf{f}_{yy}$.

Ideally a method would have $\hat{\mathbf{l}} \approx \mathbf{l}$. However to make the error estimate of BY1 second order requires an additional stage. Since an additional implicit stage would double the computational cost an explicit stage is added instead. With the additional stage the third order error terms can be reduced and we obtain method BY2,

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ 1 & 1/2 & 1/2 & 0 \\ 1/2 & 1/4 & 1/4 & 0 \\ \hline & 1/6 & 1/6 & 2/3 \\ \hline & 1/2 & 1/2 & 0 \end{array} \quad (2.59)$$

where $\mathbf{l} = 1/12 \mathbf{f} \mathbf{f}_{yy}^2$ and $\hat{\mathbf{l}} = 1/12 \mathbf{f}^2 \mathbf{f}_{yy}$. In both this method and the previous one the error estimate is A-stable. This cannot be easily achieved for higher order methods.

To construct a third order embedded method two implicit stages are used. Using Eqs. (2.55) and (2.57) the A-stable condition for this case is $\beta_1 > 1/2$ [83]. Adding this constraint to the order conditions and minimizing the $p = 4$ terms, the resulting method BY3 is,

$$\begin{array}{c|ccc}
0 & 0 & 0 & 0 \\
c_2 & a_{21} & a_{22} & 0 \\
c_3 & a_{31} & a_{32} & a_{33} \\
\hline
& b_1 & b_2 & b_3 \\
\hline
& \hat{b}_1 & \hat{b}_2 & \hat{b}_3
\end{array} \tag{2.60}$$

where,

$$\begin{aligned}
a_{21} &= 0.01863465511 & a_{22} &= 0.7886751346 \\
a_{31} &= 0.3214171387 & a_{32} &= -0.5335197972 \\
a_{33} &= 0.7886751346 & b_1 &= 0.2295834467 \\
b_2 &= 0.2418292011 & b_3 &= 0.5285873522 \\
\hat{b}_1 &= 0.1918145421 & \hat{b}_2 &= 0.1474512676 \\
\hat{b}_3 &= 0.6607341902
\end{aligned} \tag{2.61}$$

so that $\mathbf{c} = (0, 0.807, 0.577)$, $\mathbf{l} = -0.0897792\mathbf{f}\mathbf{f}_y^3 + 0.00450165\mathbf{f}^2\mathbf{f}_y\mathbf{f}_{yy} - 0.0035737\mathbf{f}^3\mathbf{f}_{yyy}$ and $\hat{\mathbf{l}} = 0.0569177\mathbf{f}\mathbf{f}_y^2 + 0.0087902\mathbf{f}^2\mathbf{f}_{yy}$.

Although the previous method minimizes the number of implicit stages for its order, the method has the undesirable property that the values of \mathbf{c} are unevenly spaced apart. This makes it difficult to estimate the next step. Also it is difficult to minimize certain stages since the Quasi-Newton Hessian is often significantly different from the previous stage. This problem can be avoided by adding an additional implicit stage, as well as constraints so

that $\mathbf{c} = (0, 1/3, 2/3, 1)$. The resulting method BY4 is,

$$\begin{array}{c|ccccc}
 0 & 0 & 0 & 0 & 0 \\
 1/3 & 1/6 & 1/6 & 0 & 0 \\
 2/3 & 1/6 & 1/3 & 1/6 & 0 \\
 1 & 1/12 & 1/2 & 1/4 & 1/6 \\
 \hline
 & 1/12 & 1/2 & 1/4 & 1/6 \\
 \hline
 & -1/12 & 1 & -1/4 & 1/3
 \end{array} \tag{2.62}$$

with $\mathbf{l} = 1/216\mathbf{f}\mathbf{f}_y^3 + 1/108\mathbf{f}^2\mathbf{f}_y\mathbf{f}_{yy} + 1/648\mathbf{f}^3\mathbf{f}_{yyy}$ and $\hat{\mathbf{l}} = 1/216\mathbf{f}\mathbf{f}_y^3 + 5/216\mathbf{f}^2\mathbf{f}_y\mathbf{f}_{yy} + 1/162\mathbf{f}^3\mathbf{f}_{yyy}$. As shown in the results section this method works particularly well, so for clarity we also give the full form of the step to obtain y_{n+1} ,

$$\begin{aligned}
 Y_1 &= y_n + h(1/6f(y_n) + 1/6f(Y_1)) \\
 Y_2 &= y_n + h(1/6f(y_n) + 1/3f(Y_1) + 1/6f(Y_2)) \\
 y_{n+1} &= y_n + h(1/12f(y_n) + 1/2f(Y_1) + 1/4f(Y_2) + 1/6f(y_{n+1})).
 \end{aligned} \tag{2.63}$$

If the vector \mathbf{c} is not spaced out, a fourth order method can be constructed with the same number of implicit stages. The resulting method BY5 is,

$$\begin{array}{c|ccccc}
 0 & 0 & 0 & 0 & 0 & 0 \\
 c_2 & a_{21} & a_{22} & 0 & 0 & 0 \\
 c_3 & a_{31} & a_{32} & a_{33} & 0 & 0 \\
 c_4 & a_{41} & a_{42} & a_{43} & a_{44} & 0 \\
 1 & b_1 & b_2 & b_3 & b_4 & 0 \\
 \hline
 & b_1 & b_2 & b_3 & b_4 & 0 \\
 \hline
 & \hat{b}_1 & \hat{b}_2 & \hat{b}_3 & \hat{b}_4 & \hat{b}_5
 \end{array} \tag{2.64}$$

where,

$$\begin{aligned}
a_{21} &= -0.2492773031 & a_{22} &= 1.111951692 \\
a_{31} &= 0.2925884303 & a_{32} &= -0.6570244861 \\
a_{33} &= 1.111984065 & a_{41} &= -0.9715595928 \\
a_{42} &= 0.04319206389 & a_{43} &= -0.05841439430 \\
a_{44} &= 0.9933909624 & b_1 &= -8.108748860 \\
b_2 &= 0.01275682451 & b_3 &= 0.5788320810 \\
b_4 &= 8.517159954 & \hat{b}_1 &= -9.384121599 \\
\hat{b}_2 &= 0.01133076785 & \hat{b}_3 &= 0.5346659509 \\
\hat{b}_4 &= 9.812438904 & \hat{b}_5 &= 0.02568597579
\end{aligned} \tag{2.65}$$

and $\mathbf{c} = (0, 0.863, 0.758, 0.007, 1)$. The final linear combination of the stages is included as an extra stage for the benefit of the error estimate. In this case, $\mathbf{l} = 0.165587\mathbf{f}\mathbf{f}_y^4 - 0.15899\mathbf{f}^2\mathbf{f}_y^2\mathbf{f}_{yy} - 0.000137813\mathbf{f}^3\mathbf{f}_{yy}^2 - 0.000645215\mathbf{f}^3\mathbf{f}_y\mathbf{f}_{yyy} - 0.000507163\mathbf{f}^4\mathbf{f}_{yyyy}$ and $\hat{\mathbf{l}} = -0.0514673\mathbf{f}\mathbf{f}_y^3 - 0.00340323\mathbf{f}^2\mathbf{f}_y\mathbf{f}_{yy} - 0.00105339\mathbf{f}^3\mathbf{f}_{yyy}$.

2.3.6 Summary of methods

To summarize the many methods outlined in the previous three sections, the local errors, stability properties, and other basic properties are reported in Tables 2.3.6 and 2.3.6. From these tables we can determine the possible reasons for the differences between them. For the two tables the local error is a measure of the absolute value of the error coefficients which multiply the next highest order derivatives. For example if a method is third order then,

$$\mathbf{y}(t_1) - \mathbf{y}_1 = \phi h^4 + \mathcal{O}(h^5) \tag{2.66}$$

where $\mathbf{y}(t)$ is the exact solution and ϕh^4 is the local error. In this third order case without time dependence the ϕ term is,

$$\phi = v_1\mathbf{f}(\mathbf{y})\mathbf{f}'(\mathbf{y})^3 + v_2\mathbf{f}(\mathbf{y})^2\mathbf{f}'(\mathbf{y})\mathbf{f}''(\mathbf{y}) + v_3\mathbf{f}(\mathbf{y})^3\mathbf{f}'''(\mathbf{y}), \tag{2.67}$$

so that the local error(LE) is reported as,

$$LE = \log_{10} \left[\frac{(1/24, 1/6, 1/24) - (v_1, v_2, v_3)}{\|(1/24, 1/6, 1/24)\|} \right], \quad (2.68)$$

where $\mathbf{v} = (1/24, 1/6, 1/24)$ are the exact coefficients. We also define here other local errors LEE1 and LEE2 which use the same measure to show the local error of the embedded error estimate, Δ_0 . Here LEE1 is a measure of the lowest order non-zero error estimate and LEE2 is a measure of the next lowest order error estimate.

Method	γ	LE	p	s	IS	Stable	Ref
GS2		-0.30	2	2	1	A	[65]
IM		-0.40	2	1	1	A	[77]
N1	γ_1	-0.14	3	2	2	A	[64]
N1	γ_{-1}	-1.29	3	2	2		[64]
N2	γ_0	0.246	4	3	3	A	[64]
N2	γ_1	-1.55	4	3	3		[64]
N2	γ_{-1}	-2.21	4	3	3		[64]
A1		-0.75	2	2	2	SS	[77]
A2		-0.74	3	3	3	SS	[77]

Table 2.2: Non-Embedded Methods. 'Method' is the Butcher array used, 'LE' the local error, ' p ' the order, ' s ' the number of stages, ' IS ' the number of implicit stages, 'stable' the stability of the method and 'ref' the reference in which the method was found.

2.3.7 Algorithm for multi-stage implicit integration

For a given Butcher array with $(\mathbf{A}, \mathbf{b}, \mathbf{c})$, the basic algorithm follows for obtaining the path from $y(t = 0)$ to $y(t_{end})$ with an estimated difference to the exact path $TOL > \|y_i - y(t_i)\|$,

Method	LE	LEE1	LEE2	p	s	IS	Stable	Ref
HW1	-1.4	0.0774	0.249	4	5	5	L	[73]
BC1	-1.3	0	0.0469	4	6	5	L	[84]
C1	-0.74	0.367	0.664	3	3	3	SS	[86]
C2	-0.45	0.354	0.986	4	5	5	SS	[86]
NST3	-0.16	0.119	0.434	3	3	3	A	[85]
NST2	0.45	0.672	2.97	4	4	4	A	[85]
NST1	0.013	0.691	2.9	4	5	4	AN	[85]
BY1	-0.3	1.5	2.29	2	2	1	A	
BY2	-0.45	0	0.354	2	3	1	A	
BY3	-0.29	0.244	0.668	3	3	2	A	
BY4	-1.2	0	0.138	3	4	3	A	
BY5	0.3	0.292	1.65	4	5	3	A	

Table 2.3: Embedded Methods. LE, LEE1 and LEE2 are measures of the error and are described in the Summary of methods section. All other headings are the same as those in Table 2.3.6.

a local tolerance for the convergence of each minimization $TOL2$ and an initial trust radius TR_0 :

- (I) Calculate the Hessian and initial direction at the TS.
- (II) Set $t = 0$, $TR_i = TR_0$ and $h = (0.1 * TOL)^{\frac{1}{p}}$.
- (III) For stage i IF $a_{ii} = 0$ THEN take explicit step and GOTO XI.
- (IV) ELSE IF $a_{ii} \neq 0$ THEN guess initial \mathbf{Y}_i .
- (V) Evaluate the energy and gradient, (E, \mathbf{g}_i) .
- (VI) IF \mathbf{H}_i does not exist use \mathbf{H}_j and TR_i from the closest point.
- (VII) Update \mathbf{H}_i with E and \mathbf{g}_i .
- (VIII) Update TR_i using Eq. (2.70).
- (IX) Solve for \mathbf{Y}_i on the quadratic version of Eq. (2.32) using $\mathbf{H}_i, \mathbf{g}_i$ and E .
- (X) IF $\Delta g_i / \|\mathbf{g}_i\| > TOL2/a_{ii}$ THEN GOTO V.
- (XI) IF $i \neq s$ then GOTO III with $i = i + 1$.
- (XII) Calculate error, Δ_0 by Eq. (2.43).
- (XIII) Adjust h by Eq. (2.69).
- (XIV) IF $\Delta_0 < TOL$ THEN $t = t + h$ and save y_t .
- (XV) IF $(t \geq t_{end})$ THEN stop ELSE GOTO III with $i = 1$.

To update the step size h the equation from Ref. [88] is used,

$$h = h \left| \frac{TOL}{\Delta_0} \right|^{\frac{1}{p}} \quad (2.69)$$

where Δ_0 is given by Eq. (2.43).

To update the trust radius the standard algorithm[36] is used, with

$$\rho = \frac{V(y_i) - V(y_i + \Delta y_i)}{g_i^T \Delta y_i + \frac{1}{2} \Delta y_i^T H_i \Delta y_i} \quad (2.70)$$

where y_i is the current set of coordinates and Δy_i is the change resulting from a minimization step. Here ρ is a measure of the quality of the quadratic approximation to the surface using H_i . The closer ρ is to unity the more confidence one can have in increasing the trust radius. The algorithm used for changing trust radius, TR_i is simply: if ($\rho < \frac{1}{4}$) then set $TR_i = \frac{1}{4} \|\Delta y_i\|$, else if ($\rho > 0.8$ and $\|\Delta y_i\|/TR_i > 0.8$) then set $TR_i = 2TR_i$, otherwise TR_i is not changed.

To guess the next point a Euler step is taken as the initial guess for the first stage. For each additional stage the gradients at previous stages are used to extrapolate a guess for the current stage. The initial guess for the next point is then calculated as,

$$\mathbf{Y}_i = \mathbf{y}_n + h \sum_{j=1}^{i-1} a_{ij} \mathbf{f}(t + c_j h, \mathbf{Y}_j) + h a_{ii} \mathbf{f}_{extrp}(t_i, \mathbf{Y}_i). \quad (2.71)$$

where each $\mathbf{f}(t + c_j h, \mathbf{Y}_j)$ has been calculated at a previous stage and $\mathbf{f}_{extrp}(t_i, \mathbf{Y}_i)$ is the extrapolated value.

2.4 Comparison of Integration methods on Potential Energy Surfaces

Three molecular systems and the Müller-Brown potential are compared in this section. All calculations were done with Gaussian 03 [41] at the HF/STO-3G level of theory, with Cartesian coordinates. The DIRK algorithm was implemented in MatlabTM with the Optimization ToolboxTM. For the GS algorithm we followed the outline in Ref. [60]. The constrained minimization, was performed with the Matlab function “fmincon” to solve Eq. (2.32), while the extrapolation in the DIRK methods was done with ‘interp1’. The exact Hessian was calculated for the TS and then updated at each point with the BFGS update for the combined method and the Symmetric Rank-1 (SR1) update [36] for the DIRK methods. Also we set $TOL2 = 0.01TOL$.

Normally the convergence of each minimization step was very rapid, requiring only 2 to 3 energy and gradient evaluations. For the combined method the GS algorithm was run for each system, first with a step size of $h = 0.1$ for the lower accuracy results and then with $h = 0.01$ for the higher accuracy results. Since the GS algorithm is a second order method, when $h = 0.1$ we expect the error to be of order $\mathcal{O}(h^3) = \mathcal{O}(10^{-3})$, while at $h = 0.01$ the error should be of order $\mathcal{O}(10^{-6})$.

A Fortran 77 version of the 3rd order DUMKA algorithm is available[90]. The code required the functions FUN and RHO which are $\mathbf{f}(t, \mathbf{y})$ and $\rho(\nabla_{\mathbf{y}}\mathbf{f}(t, \mathbf{y}))$, respectively. For the SDP these are given by Eqs. (2.1) and (2.6) respectively. The Hessian required by Eq. (2.6) was obtained with the same BFGS updating scheme as the GS algorithm and a safety factor of $\epsilon = 1.3$ was used to multiply the spectral radius. Although not shown here, similar results were obtained by using the Hessian from the TS without updating and by using a larger safety factor $\epsilon = 1.5$.

The traditional fourth order RK code was available from Netlib [91] in the Fortran 77 code RKSUITE. This code only required the function $\mathbf{f}(t, \mathbf{y})$ given by Eq. (2.1). We set the parameter $METHOD = 2$ so the error would be estimated as the difference between

the fourth and fifth order integration.

For the combined method, in all cases, we examined the reaction path at intervals of arc length 0.5 au. If the last point fell close to the minimum, the calculation was omitted. This is because the integration became very stiff and the extra point did not add significant information to the the path. In the explicit methods, the tolerance was set to 10^{-3} for low accuracy calculations and 10^{-6} for high accuracy calculations in order to match the GS results. The exact answer was computed with RKSUITE at a tolerance of 10^{-7} . To switch between methods, at the start we checked to see if the explicit method was more efficient every 10 iterations when $h = 0.01$ for the GS algorithm. Toward the end of the integration we followed the outline given in the above section with $N_{impl} = 2$, switching to the GS algorithm when $\frac{h_{impl}}{N_{impl}} > \frac{h_{expl}}{N_{expl}}$.

We did not include any form of error estimation for the GS algorithm since this was not suggested in Ref. [65]. Error estimation could be achieved with step doubling [38], but this would more than double the number of calculations involved.

2.4.1 Müller Brown Potential

This small two dimensional system shown in Fig. 2.7 provides an excellent sample system for testing methods without having to worry about precision or coordinate choice issues associated with chemical systems. The results of applying non-embedded SDIRK methods to this system for a fixed step size of $h = 0.1$, appear in Table 2.4. The second order IM and GS2 methods give similar results, which is to be expected since they are the same order and both have only one implicit stage. The N1 and N2 methods perform poorly at this step size for the A-stable γ . Interestingly the γ corresponding to non A-stable methods give much better results. These results are perhaps best understood in the context section 2.2 where it was shown that the entire path is not necessarily stiff and that only near the endpoints does the SDP become very stiff. As a result stability may not be critically important for some parts of reaction paths, as is clearly the case for this system. Table 2.3.6 shows that the local error for these methods is significantly less than their A-stable versions. As may

be predicted then, the more stable and more costly (in terms of stages), SS-stable, A1 and A2 methods do not do well compared to other methods.

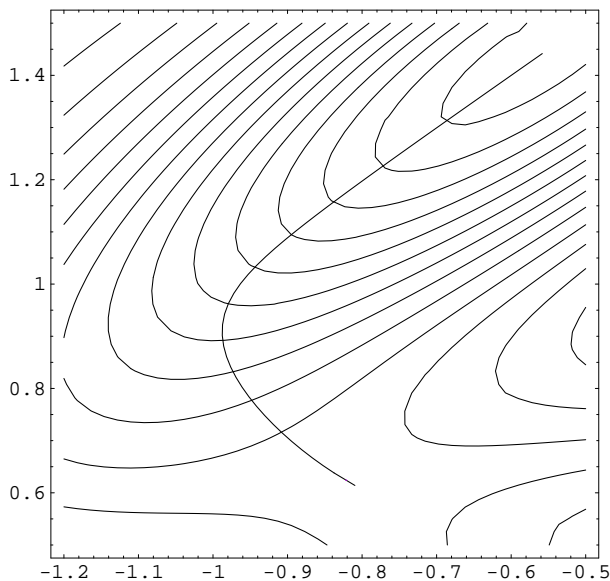


Figure 2.7: The Müller-Brown Potential with the exact MEP connecting the TS at $(-0.82, 0.62)$ to the minimum at $(-0.56, 1.44)$.

Given the results of this system then we can conclude that for non-stiff systems, non A-stable methods can do quite well, especially if high accuracy is desired. For low accuracy paths the GS2 or IM method are clearly the method of choice.

2.4.2 $SiH_2 + H_2 \rightarrow SiH_4$

The first molecular system we examine is the association of H_2 with SiH_2 . The energy profile is detailed in Fig. 2.8. This reaction is quick to run and turns out to be a reasonably non-stiff problem. For a DUMKA3 run the values of ρ and $\|\mathbf{g}\|$ are shown in Fig. 2.5. The inverse relationship between ρ and $\|\mathbf{g}\|$ is clear from the figure. We would expect that the explicit solvers would do best in the middle region where ρ is smallest.

At the lower tolerance of 10^{-3} , shown in Fig. 2.9 (a) DUMKA3 does much better

Method	γ	min(LE)	max(LE)	Eval
GS2		-2.5	-3.53	3
IM		-2.24	-3	2.9
N1	γ_1	-2.75	-3.3	6.2
N1	γ_{-1}	-3.25	-3.88	5.9
N2	γ_0	-2.96	-3.96	10.3
N2	γ_1	-3.68	-4.02	9.1
N2	γ_{-1}	-4.07	-4.62	8.9
A1		-2.69	-3.34	5.2
A2		-2.9	-3.36	8.4

Table 2.4: Non-embedded method results at fixed step size $h = 0.1$ on the Müller Brown surface. Ten steps in total were taken. 'Method' is Butcher array used, ' γ ' distinguishes between the (2,3) and (3,4) methods, Max/Min(LE) is the maximum/minimum local error and 'Eval' is the mean number of evaluations per step.

than RKSUITE, but neither method performs better than the GS algorithm over any part of the path. Figure 2.9 (b) shows the accuracy compared to the exact path for all three methods. While DUMKA3 and the GS algorithm give the correct accuracy, RKSUITE significantly deviates from the requested tolerance. This is most likely due to the stiff behavior near the TS where the gradient is small. It is possible that neither the fourth nor fifth order integration step is able to follow the path correctly and the estimated error does not represent the true error well.

At the higher tolerance of 10^{-6} , shown in Fig. 2.10 (a), RKSUITE does somewhat better than DUMKA3 because the problem is so non-stiff. As expected though, near the beginning and end of the integration where $\|\mathbf{g}\|$ is smallest, the explicit methods perform poorly. This is especially true for the final point which is close to the minimum.

Overall, for high accuracy the RK method performs slightly better than DUMKA3, but

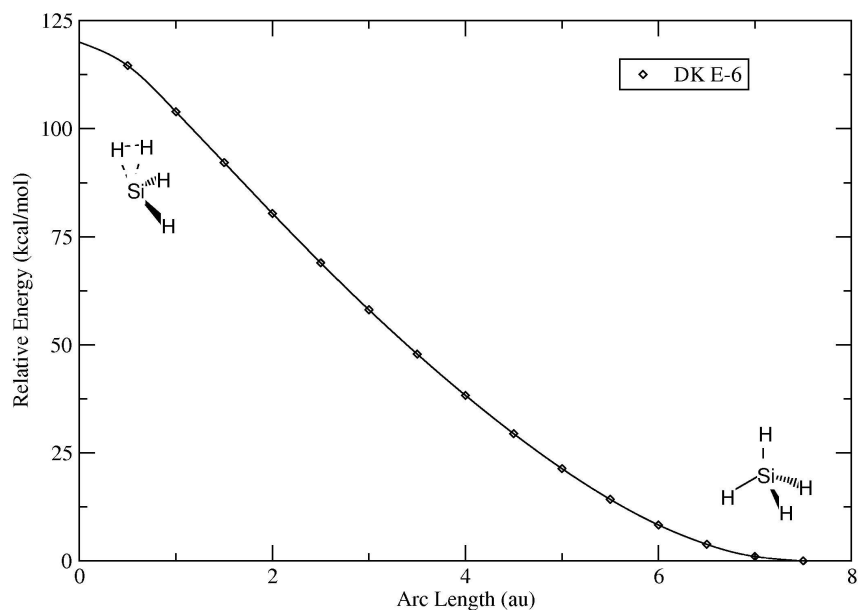
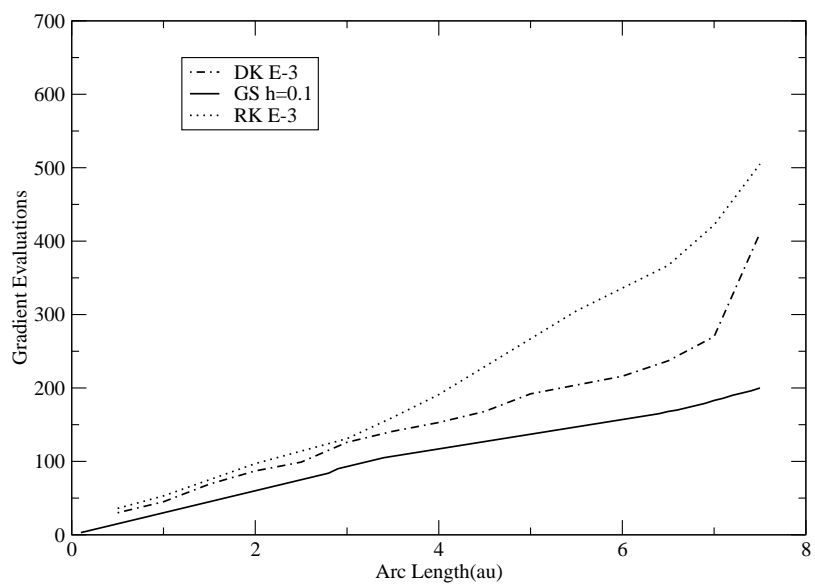


Figure 2.8: Energy profile for $SiH_2 + H_2 \rightarrow SiH_4$ as calculated with DUMKA3.

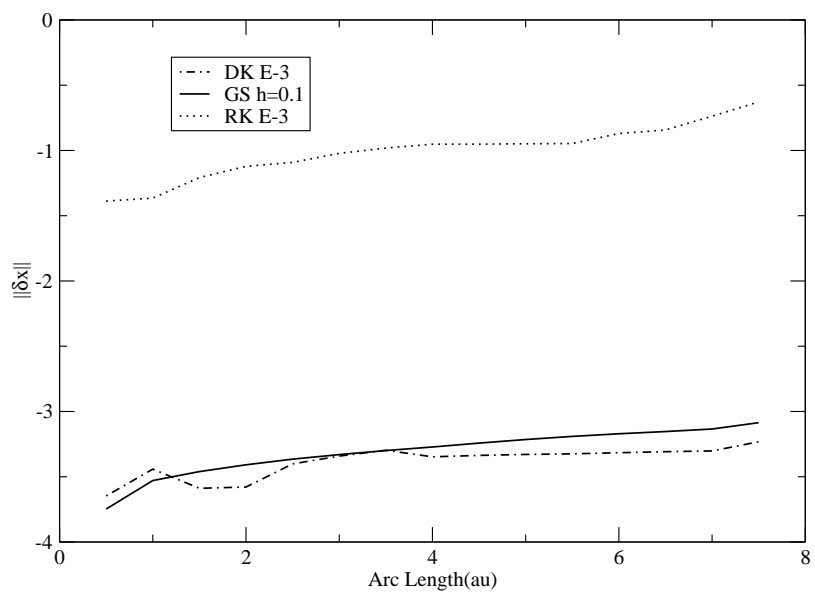
as Fig. 2.10 (b) shows, the RK method has trouble keeping the error less than the tolerance for the final step. The GS algorithm for this system gives a much more accurate path than would be expected.

When the GS algorithm is combined with the explicit methods, as seen in Fig. 2.11, the unfavorable performance on the last step is dramatically improved. Also, the accuracy problems with the final RK step are avoided.

For the DIRK embedded methods the reaction path was followed at a tolerance of 10^{-3} . These results are summarized in Figure 2.12. The fourth order, L-stable methods HW1 and BC1 give almost an order of magnitude more accurate paths than are required by the tolerance. Method BC1 is able to achieve this with a very small number of steps. The A-stable method NST3 is also able to integrate the path with a small number of evaluations but the error is not estimated well and some steps have a larger error than the tolerance. The NST1 and NST2 methods are able to bound the error better but require many more integration steps. Similar to the Müller Brown results the more stable SS-stable methods C1 and C2 require a large number of evaluations without significantly improving the accuracy.

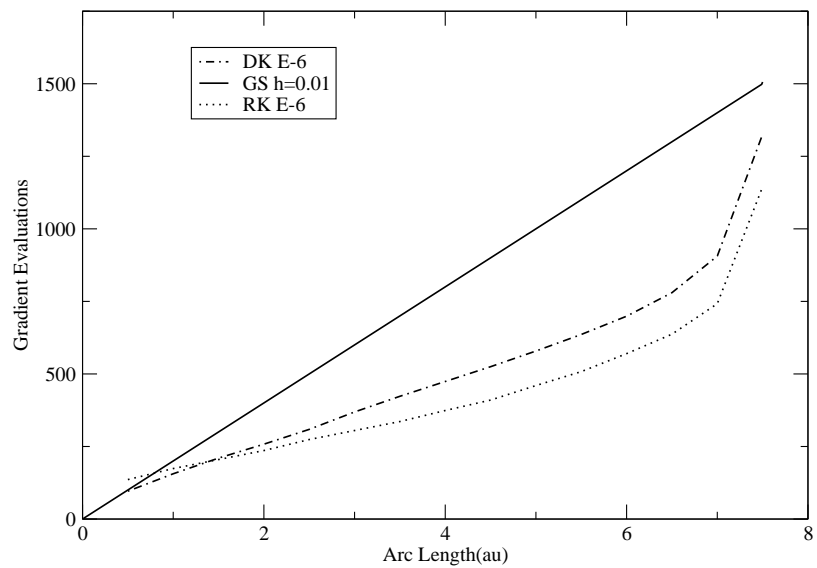


(a)

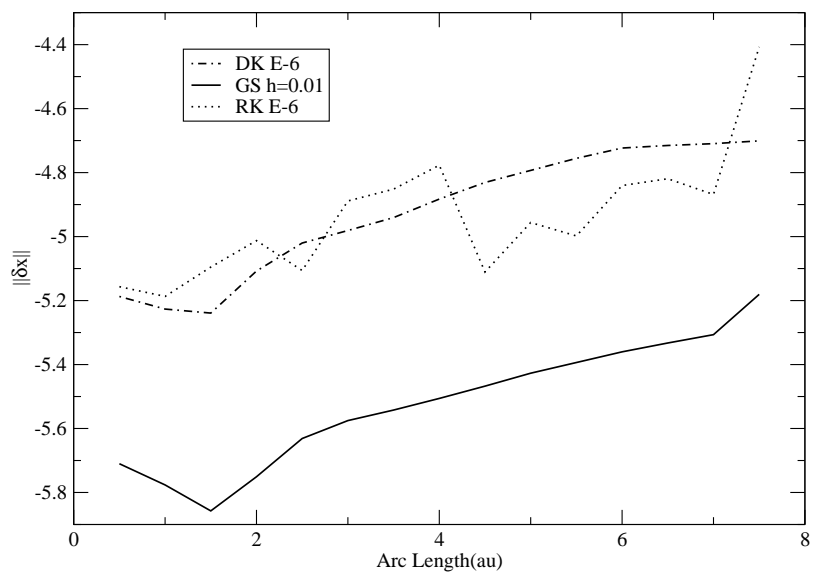


(b)

Figure 2.9: (a) Performance and (b) accuracy of explicit methods compared to the GS algorithm with $h = 0.1$ for $SiH_2 + H_2 \rightarrow SiH_4$.

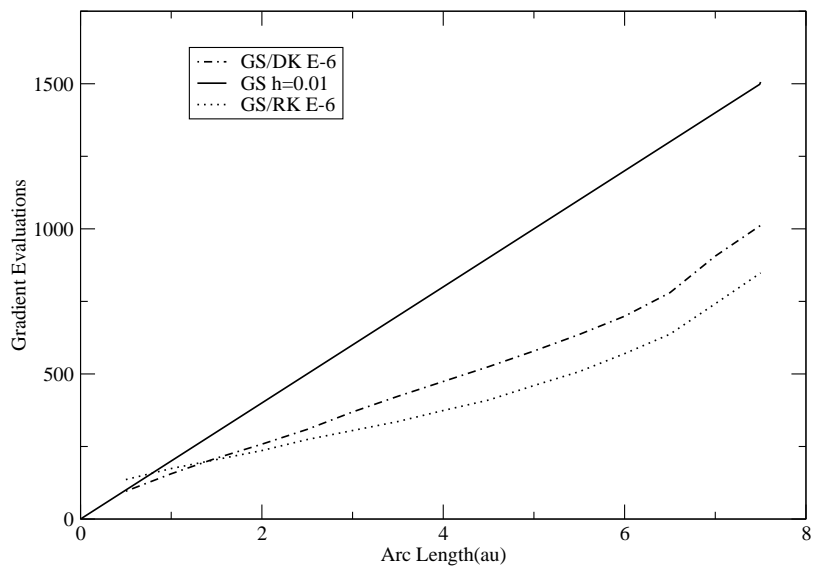


(a)

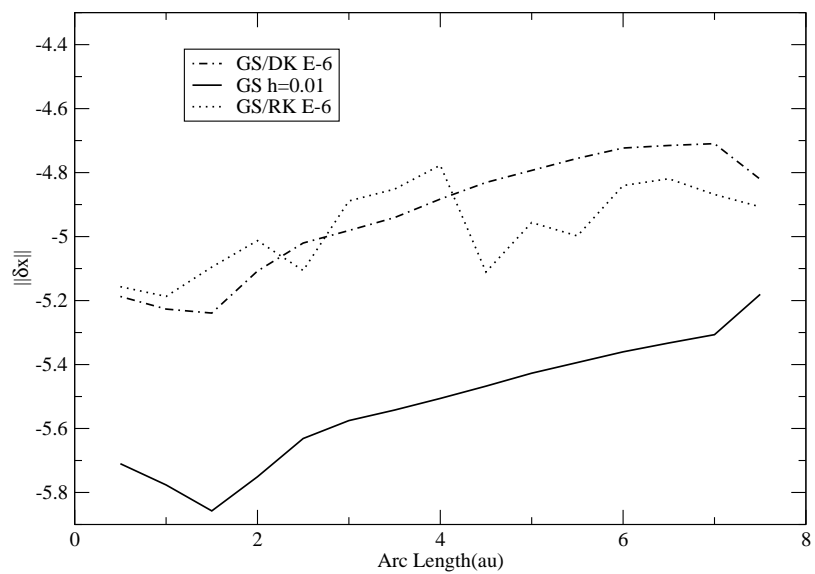


(b)

Figure 2.10: (a) Performance and (b) accuracy of explicit methods compared to the GS algorithm with $h = 0.01$ for $SiH_2 + H_2 \rightarrow SiH_4$.



(a)



(b)

Figure 2.11: (a) Performance and (b) accuracy of the combined explicit-implicit methods compared to the GS algorithm with $h = 0.01$ for $SiH_2 + H_2 \rightarrow SiH_4$.

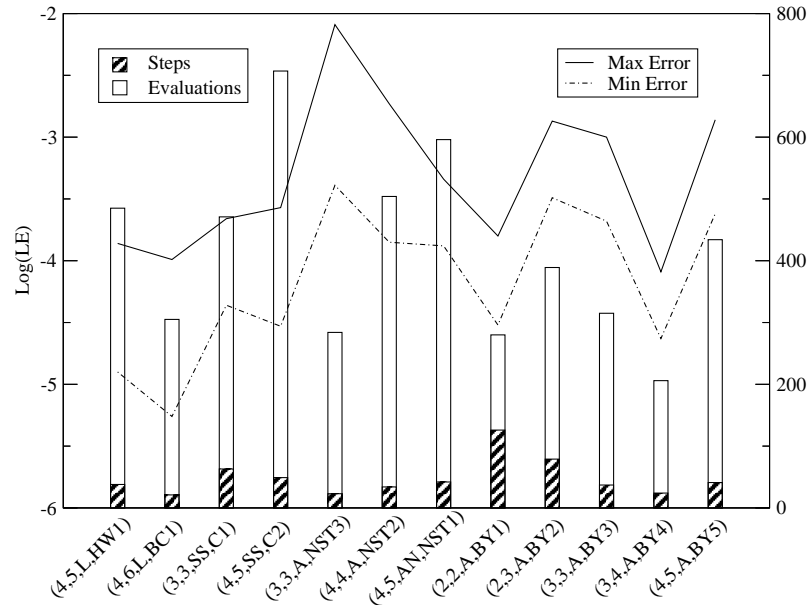


Figure 2.12: Variable time step embedded method accuracies for the system $SiH_2 + H_2 \rightarrow SiH_4$ with $tol = 10^{-3}$. The total arc length of the path (t_{end}) is 8 angstroms. 'Method' is Butcher array used, 'Max/Min Error' is the Log_{10} maximum/minimum local error (left axis), 'Evaluations' is the total number of gradient evaluations and 'Steps' is the number of steps taken (right axis).

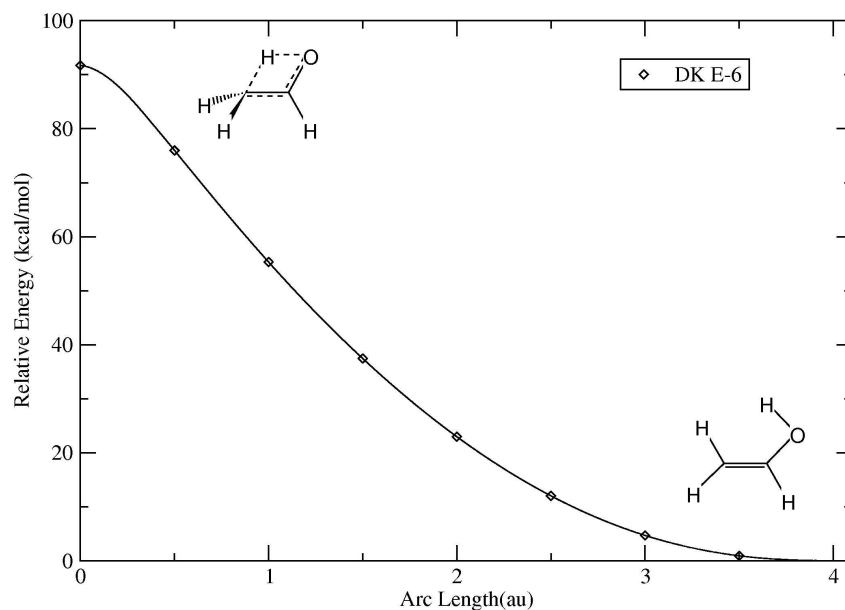
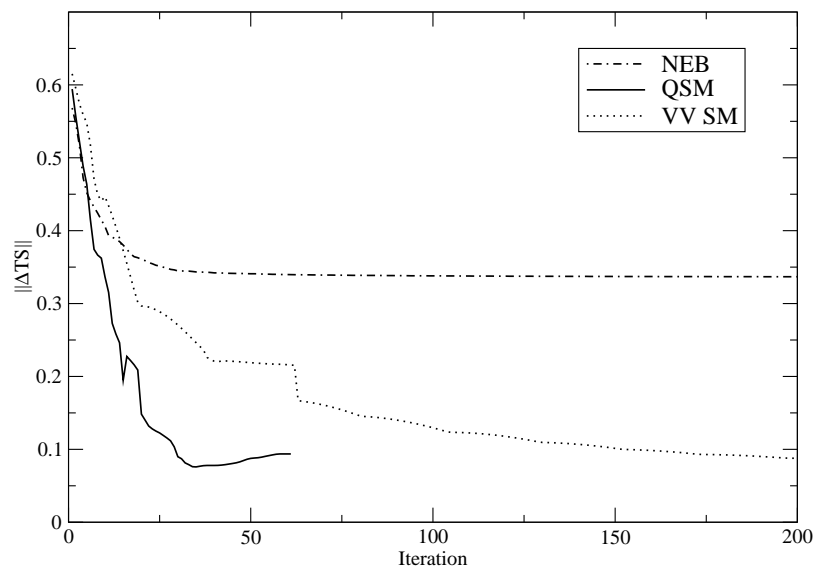


Figure 2.13: Energy profile for $CH_3CHO \rightarrow CH_2CHOH$.

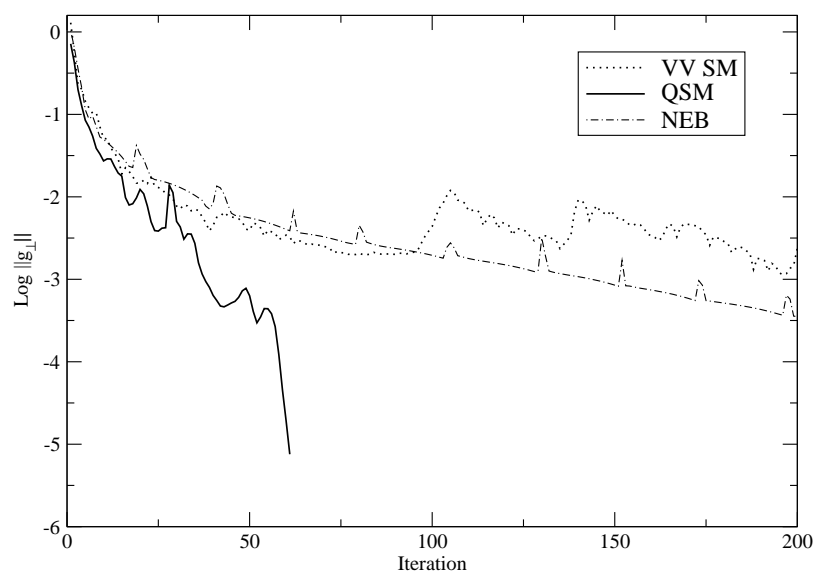
All the methods we developed BY1 through BY5 are able to integrate the path with a reasonably small number of evaluations. From the results of the method BY4 we observe the benefit of adding an extra stage to space out the stages and improve the error estimate. Method BY3 is the same order as BY4 with one less stage, but does not perform better. The same is true of BY5 which despite being one order higher than BY4 also does worse. In addition it is evident that adding an explicit stage to BY1 to get BY2 did improve the error estimate. This allowed a reduction in the number of steps, but because of the extra stage more gradient evaluations were necessary. Overall it appears that methods BC1, BY1 and BY4 are the best choices for automatic integration.

2.4.3 $CH_3CHO \rightarrow CH_2CHOH$

Similar behavior as the previous example is seen in this rearrangement for the combined algorithm. The energy profile is shown in Fig. 2.13 with the two structures. The convergence results appear in Fig. 2.14.



(a)



(b)

Figure 2.14: (a) Performance and (b) accuracy of the combined explicit-implicit methods compared to the GS algorithm for $CH_2CHOH \rightarrow CH_3CHO$.

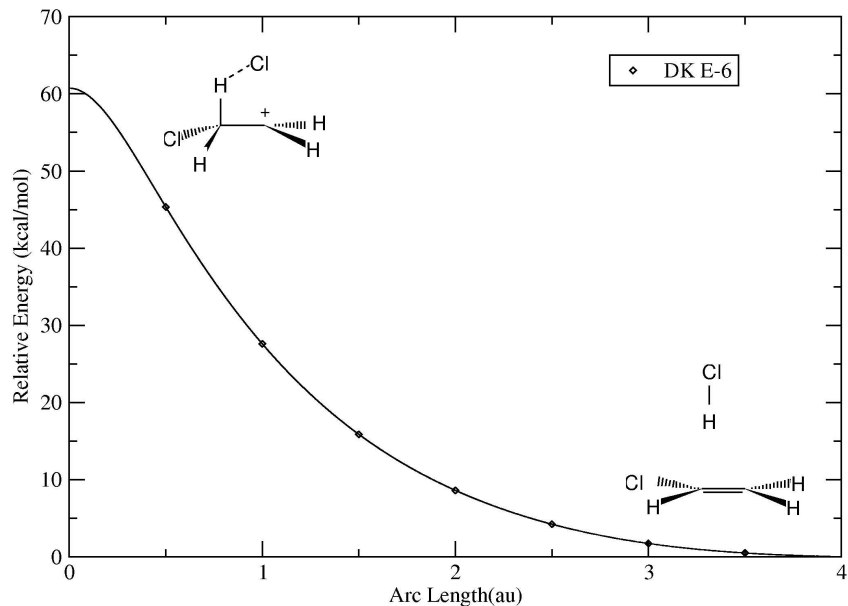
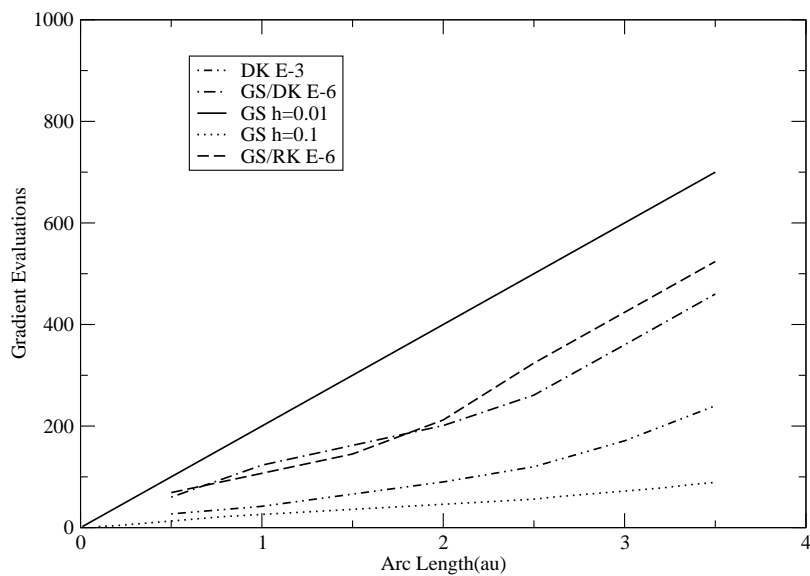


Figure 2.15: Energy profile for $H_2ClC-C(+H_2) + Cl^- \rightarrow ClHC=CH_2 + HCl$ as calculated with DUMKA3.

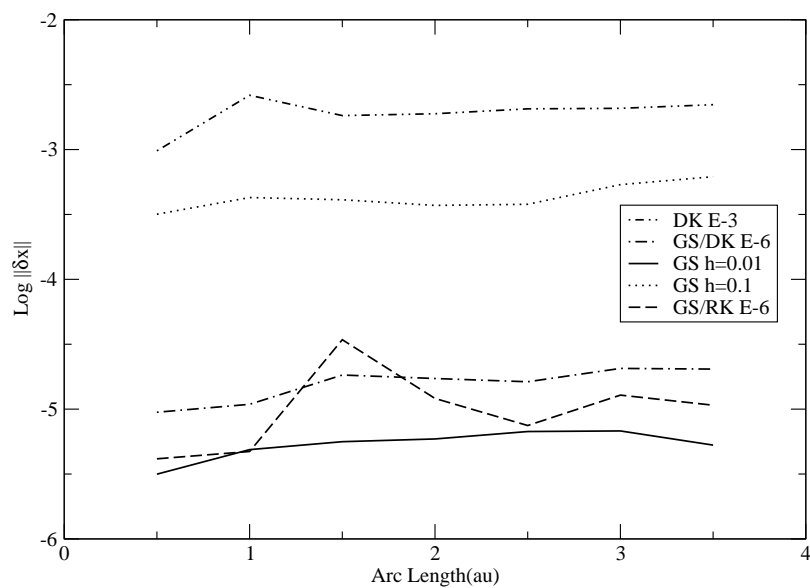
Again, the GS algorithm performs best at low accuracy while the explicit methods do better at high accuracy. Here DUMKA3 does slightly better than RKSUITE when the equation becomes stiffer. DUMKA3 does not need to switch to the GS algorithm in this case even near the end of the integration. The RK method switches near $3.5au$.

2.4.4 $H_2ClC-C(+H_2) + Cl^- \rightarrow ClHC=CH_2 + HCl$

In contrast to the hydrogenation reaction, $SiH_2 + H_2 \rightarrow SiH_4$ discussed earlier this is a dissociation reaction. Here $\|g\|$ goes to zero over a larger stretch of the reaction making for a much longer stiff region. The energy profile is shown in Fig. 2.15, and in Fig. 2.16 we again see that GS does best at low accuracy, while the explicit methods do much better at high accuracy. Here again because of the larger stiff region DUMKA3 is able to outperform RKSUITE at the end of the path. In this case RKSUITE switches to the GS algorithm at a path length of $2.5au$ while DUMKA3 switches near $3au$.



(a)



(b)

Figure 2.16: (a) Performance and (b) accuracy of the combined explicit-implicit methods compared to the GS algorithm for $H_2ClC-C(+)H_2 + Cl^- \rightarrow ClHC=CH_2 + HCl$.

For the DIRK framework the non-embedded method results are given in Table 2.5. Again the GS2 and IM methods are the fastest way to achieve a low accuracy path. Table 2.5 also shows the dangers of using non A-stable methods. For method N1 when $\gamma = \gamma_{-1}$ and for N2 when $\gamma = \gamma_1$ the stability region is small and for stiff regions, like those near the near of the path, the methods do not perform as well as expected. For this system the more stable A1 and A2 methods do better, while N1 with $\gamma = \gamma_1$ appears to be the method of choice for higher accuracy.

Method	γ	min(LE)	max(LE)	Eval
GS2		-3.54	-3.7	3.31
IM		-3.15	-3.75	3.54
N1	γ_1	-3.74	-4.6	8.4
N1	γ_{-1}	-2.16	-4.93	6.46
N2	γ_0	-3.46	-3.71	14
N2	γ_1	-1.64	-5.64	9.6
N2	γ_{-1}	-3.98	-5.33	9.86
A1		-3.77	-3.96	4.94
A2		-4.1	-4.3	9.03

Table 2.5: Non-embedded method accuracies for the system $H_2ClC-C(+H_2 + Cl^- \rightarrow ClHC=CH_2 + HCl$ with $h = 0.1$ and 35 steps. Heading are defined as in Table 2.4.

In Table 2.6 results are given for the embedded methods using the same fixed step size as in Table 2.5. This gives a picture at how well each method estimates the error. Methods NST3, NST2 and BY3 appear to systematically underestimate the error most severely, while HW1 and BY1 overestimate the error. In this table, BY4 stands out as being the best method to achieve high accuracy for a reasonably small number of gradient evaluations, while BY1 works well for low accuracy.

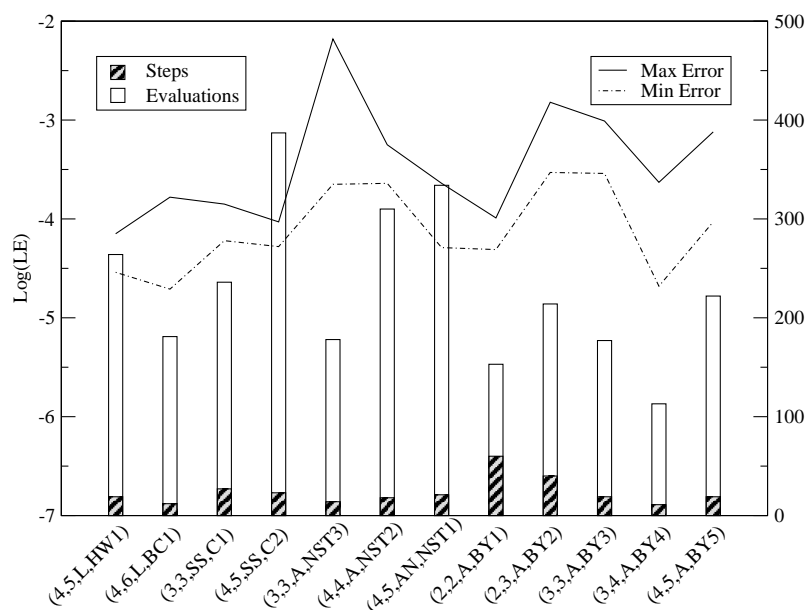


Figure 2.17: Variable time step embedded method accuracies for the system $H_2ClC-C(+)H_2 + Cl^- \rightarrow ClHC=CH_2 + HCl$ with $tol = 10^{-3}$. Heading are defined as in Table 2.12.

Method	max(Δ_0 -LE)	min(Δ_0 -LE)	max(LE)	min(LE)	Eval
HW1	1.2	0.63	-5.13	-5.68	13.7
BC1	0.623	-0.436	-5.41	-6.5	13
C1	0.822	-0.2	-4.11	-4.3	8.97
C2	0.504	-0.12	-4.38	-4.56	17.2
NST3	-0.678	-1.09	-3.77	-4.61	11.5
NST2	-0.39	-1.43	-3.73	-3.99	17.1
NST1	0.098	-0.564	-3.56	-4.58	16.4
BY1	1.2	0.433	-3.54	-3.7	3.31
BY2	-0.0409	-0.494	-2.71	-3.57	5.4
BY3	-0.144	-0.902	-3.78	-4.21	8.74
BY4	0.868	-0.582	-5.11	-5.77	7.29
BY5	0.185	-0.565	-3.79	-4.67	12.5

Table 2.6: Embedded method accuracies for the system $H_2ClC-C(+)H_2 + Cl^- \rightarrow ClHC=CH_2 + HCl$ with $h = 0.1$ and 35 steps. Max/Min(Δ_0 -LE) is given by $\max / \min(\log_{10}(\Delta_0) - \log_{10}(LE))$ and the other heading are defined as in Table 2.4.

In Figure 2.17 the embedded results are given for a desired accuracy of 10^{-3} . Similar to the previous example, methods BC1, BY1, BY3 and BY4 give good performance, with method BY4 performing best. In Figure 2.17 embedded results are given for a desired accuracy of 10^{-4} . At this higher accuracy the higher order methods do better. Methods BY3 and NST3 have trouble estimating the accuracy well for this case and have steps that fall outside of the tolerance. Overall, method BY4 is again the best choice.

2.5 Summary

For high accuracy reaction path following, explicit methods can perform significantly better than implicit methods in non-stiff regions. These non-stiff regions exist in parts of the reaction path where the gradient is large, usually in the middle of the reaction path, further away from stationary points. Implicit methods by contrast consistently performed better in lower accuracy calculations and in highly stiff regions.

Between the two explicit methods tested, the DUMKA3 algorithm was shown to give slightly better results than the traditional RK method. This was most likely due to DUMKA3's ability to enlarge its stability region to match the stiffness of the problem.

The optimal algorithm resulted from a combination of two methods with a simple switch criterion based on the current efficiency of the method. The utility of this combined explicit-implicit method was demonstrated using a variety of molecular systems, where it was shown that the combined method could reduce the number of some calculations by almost a half.

While the overall efficiency of integrating the path is increased by using an explicit solver, often the stiff regions are of primary interest. To this end the DIRK framework is a generalization of the popular implicit trapezoidal rule introduced in Ref. [65] This generalization allows the construction of methods of any order which can then be embedded to obtain error estimates, allowing for efficient and accurate automatic integration.

There are many different ways to construct a method. It was shown that adding additional explicit stages beyond the first stage did not necessarily improve the method. Also it was shown that there was no significant benefit to constructing methods that were more stable than A-stable.

The results demonstrated the best method was an A-stable, four stage, third order ESDIRK which we referred to as BY4. This method was constructed with stages that were evenly spaced out so the Hessian could be easily updated from the previous stage. Method BY4 was demonstrated on two chemical reactions to be consistently the most efficient method while also bounding the error well.

Bibliography

- [1] A. Warshel and M. Levitt. Theoretical studies of enzymic reactions - dielectric, electrostatic and steric stabilization of carbonium-ion in reaction of lysozyme. *J. Mol. Biol.*, 103:227, 1977.
- [2] M. Svensson, S. Humbel, R.D.J. Froese, T. Matsubara, S. Sieber, and K. Morokuma. Oniom: A multilayered integrated mo+mm method for geometry optimizations and single point energy predictions. a test for diels-alder reactions and pt(p(t-bu)(3))(2)+h-2 oxidative addition. *J. Phys. Chem.*, 100:19357–19363, 1996.
- [3] H.M. Senn and W Thiel. Qm/mm methods for biological systems. *Top. Curr. Chem.*, 268:173–290, 2007.
- [4] H. Lin and D.G. Truhlar. Qm/mm: what have we learned, where are we, and where do we go from here? *Theo. Chem. Acc.*, 117:185–199, 2007.
- [5] P.A. Kollman, B. Kuhn, and M. Perakyla. Computational studies of enzyme-catalyzed reactions: Where are we in predicting mechanisms and in understanding the nature of enzyme catalysis? *J. Phys. Chem. B*, 106:1537–1542, 2002.
- [6] V. Gogonea, D. Suarez, A. van der Vaart, and K.W. Merz. New developments in applying quantum mechanics to proteins. *Curr. Opin. Struct. Bio.*, 11:217–223, 2001.
- [7] H. Hu, Z.Y. Lu, and W.T. Yang. Qm/mm minimum free-energy path: Methodology and application to triosephosphate isomerase. *J. Chem. Theory Comput.*, 3:390–406, 2007.
- [8] J.P. Ryckaert, G. Ciccotti, and H.J.C. Berendsen. Fep. *J. Comp. Phys.*, 23:327, 1977.
- [9] C. Dellago, P. G. Bolhuis, F. S. Csajka, and D. Chandler. Transition path sampling and the calculation of rate constants. *J. Chem. Phys.*, 108:1964, 1998.
- [10] C. Dellago, P. G. Bolhuis, and D. Chandler. On the calculation of reaction rate constants in the transition path ensemble. *J. Chem. Phys.*, 108:9263, 1998.
- [11] B.K. Dey, M.R. Janicki, and P.W. Ayers. Hamilton-jacobi equation for the least-action/least-time dynamical path based on fast marching method. *J. Chem. Phys.*, 121:6667, 2004.
- [12] S. Huo and J. E. Straub. The maxflux algorithm for calculating variationally optimized reaction paths for conformational transitions in many body systems at finite temperature. *J. Chem. Phys.*, 107:5000, 1997.
- [13] K. Fukui. The path of chemical-reactions - the irc approach. *Acc. Chem. Res.*, 14:363, 1981.
- [14] W. H Miller. Tunneling corrections to unimolecular rate constants, with application to formaldehyde. *J. Am. Chem. Soc.*, 101:6810, 1979.

- [15] Yingkai Zhang, Haiyan Liu, and Weitao Yang. Free energy calculation on enzyme reactions with an efficient iterative procedure to determine minimum energy paths on a combined ab initio qm/mm potential energy surface. *J. Chem. Phys.*, 112:3483–3492, 2000.
- [16] P. Y. Ayala and H. B. Schlegel. A combined method for determining reaction paths, minima and transition state geometries. *J. Chem. Phys.*, 107:375, 1997.
- [17] H. Jónsson, G. Mills, and K.W. Jacobsen. “*Nudged Elastic Band Method*”, in *Classical and quantum dynamics in condensed phase simulations*. World Scientific, Singapore, 1998.
- [18] B. Peters, A. Heyden, A. Bell, and A. Chakraborty. A growing string method for determining transition states: Comparison to the nudged elastic band and string methods. *J. Chem. Phys.*, 120:7877, 2004.
- [19] W. Quapp. A growing string method for the reaction pathway defined by a newton trajectory. *J. Chem. Phys.*, 122:174106, 2005.
- [20] W. E., W. Ren, and E. Vanden-Eijnden. Minimum action method for the study of rare events. *Comm Math Sci*, 1:377, 2002.
- [21] W. Ren. String method for the study of rare events. *Phys. Rev. B*, 66:052301–1, 2003.
- [22] W. E., W. Ren, and E. Vanden-Eijnden. Simplified and improved string method for computing the minimum energy paths in barrier-crossing events. *J. Chem. Phys.*, 126:164103, 2007.
- [23] H. Liu, Z. Lu, G. A. Cisneros, and W. Yang. Parallel iterative reaction path optimization in ab initio quantum mechanical/molecular mechanical modeling of enzyme reactions. *J. Chem. Phys.*, 121:697, 2004.
- [24] L. Xie, H. Liu, and W. Yang. Adapting the nudged elastic band method to determine minimum energy paths of enzyme catalyzed reactions. *J. Chem. Phys.*, 120:8039, 2004.
- [25] G. A. Cisneros, H. Liu, Z. Lu, and W. Yang. Reaction path determination for quantum mechanical/molecular mechanical modeling of enzyme reactions by combining first order and second order “chain-of-replicas” methods. *J. Chem. Phys.*, 122:114502, 2005.
- [26] P. Maragakis, S. A. Andreev, Y. Brumer, D. R. Reichman, and E. Kaxiras. Adaptive nudged elastic band approach for transition state calculation. *J. Chem. Phys.*, 117:4651, 2002.
- [27] S. A. Trygubenko and D. J. Wales. A doubly nudged elastic band method for finding transition states. *J. Chem. Phys.*, 120:2082, 2004.
- [28] B. Trout and J. W. Chu. A super-linear minimization scheme for the nudged elastic band method. *J. Chem. Phys.*, 119:12708, 2003.

- [29] R. Crehuet and JM Bofill. The reaction path intrinsic reaction coordinate method and the hamiltonjacobi theory. *J. Chem. Phys.*, 122:234105, 2005.
- [30] W Quapp. Reaction pathways and projection operators: Application to string methods. *J. Comp. Chem.*, 25:1277, 2004.
- [31] S.K. Burger and W. Yang. Quadratic string method for determining the minimum-energy path based on multiobjective optimization. *J. Chem. Phys.*, 124:054109, 2006.
- [32] M. Page and J. M. McIver. On evaluating the reaction-path hamiltonian. *J. Chem. Phys.*, 88:922, 1988.
- [33] D. J. Wales. *Energy Landscapes: Applications to Clusters, Biomolecules and Glasses*. Cambridge University Press, Cambridge, England, 2003.
- [34] W. Quapp and D. Heidrich. Analysis of the concept of minimum energy path on the potential energy surface of chemically reacting systems. *Thoer. Chim. Acta*, 66:245, 1984.
- [35] R. Olender and R. Elber. Yet another look at the steepest descent path. *J. Mol. Struct.*, 398:63, 1997.
- [36] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, New York, 1999.
- [37] R. Fletcher. *Practical Methods of Optimization*. John Wiley and Sons, New York, 1987.
- [38] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C*. Cambridge University Press, Cambridge, 1992.
- [39] Yingkai Zhang, Taisung Lee, and Weitao Yang. A pseudo-bond approach to combining quantum mechanical and molecular mechanical methods. *J. Chem. Phys.*, 110:46–54, 1999.
- [40] Y. Zhang, H. Liu, and W. Yang. “*Ab Initio QM/MM and Free Energy Calculations of Enzyme Reactions*”, in *Computational Methods for Macromolecules—Challenges and Applications*. Springer Verlag, Heidelberg, Germany, 2002.
- [41] M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, V. G. Zakrzewski, J. A. Montgomery, R. E. Stratmann, J. C. Burant, S. Dapprich, J. M. Millam, A. D. Daniels, K. N. Kudin, M. C. Strain, O. Farkas, J. Tomasi, V. Barone, M. Cossi, R. Cammi, B. Mennucci, C. Pomelli, C. Adamo, S. Clifford, J. Ochterski, G. A. Petersson, P. Y. Ayala, Q. Cui, K. Morokuma, D. K. Malick, A. D. Rabuck, K. Raghavachari, J. B. Foresman, J. Cioslowski, J. V. Ortiz, B. B. Stefanov, G. Liu, A. Liashenko, P. Piskorz, I. Komaromi, R. Gomperts, R. L. Martin, D. J. Fox, T. Keith, M. A. Al-Laham, C. Y. Peng, A. Nanayakkara, C. Gonzalez, M. Challacombe, P. M. W. Gill, B. Johnson, W. Chen, M. W. Wong, J. L. Andres, M. Head-Gordon, E. S. Replogle, and J. A. Pople. *Gaussian 03*. Gaussian, Inc., Pittsburgh PA, 2003.

- [42] J.W. Ponder. *TINKER, Software Tools for Molecular Design, Version 3.6: the most updated version for the TINKER program can be obtained from J.W. Ponder's WWW site at <http://dasher.wustl.edu/tinker>*. Washington University, St. Louis, 1998.
- [43] Wendy D. Cornell, Piotr Cieplak, Christopher I. Bayly, Ian R. Gould, Kenneth M. Merz, David M. Ferguson, David C. Spellmeyer, Thomas Fox, James W. Caldwell, and Peter A. Kollman. A second generation force field for the simulation of proteins, nucleic acids, and organic molecules. *J. Am. Chem. Soc.*, 117:5179, 1995.
- [44] W.L. Jorgensen, J. Chandrasekhar, J. Madura, R.W. Impey, and M.L. Klein. Comparison of simple potential functions for simulating liquid water. *J. Chem. Phys.*, 79:926, 1983.
- [45] B.H. Besler, K.M. Merz Jr., and P.A. Kollman. Ecp. *J. Comp. Chem.*, 11:431, 1990.
- [46] I.H. Williams and G.M. Maggiora. Rcdm. *J. Mol. Struct.*, 89:365, 1982.
- [47] J.B. Foresman and A. Frisch. *Exploring Chemistry with Electronic Structure Methods*. Gaussian Inc., Pittsburgh, PA, 1996.
- [48] R. W. Zwanzig. *Fep. J. Chem. Phys.*, 22:1420–1426, 1954.
- [49] G. Henkelman and H. Jónsson. Improved tangent estimate in the nudged elastic band method for finding minimum energy paths and saddle points. *J. Chem. Phys.*, 113:9978–9985, 2000.
- [50] W. li, S. Xu, G. Zhao, and L.P. Goh. Cubic spline knot placement. *Computer-Aided Design*, 37:791, 2005.
- [51] K. Muller and L.D. Brown. Location of saddle points and minimum energy paths by a constrained simplex optimization procedure. *Theor. Chim. Acta*, 53:75, 1979.
- [52] J.P.K. Doye and D.J. Wales. Global minima for transition metal clusters described by sutton-chen potentials. *J. Phys. Chem. A*, 101, 1997.
- [53] D. J. Wales, J. P. K. Doye, A. Dullweber, M. P. Hodges, F. Y. Naumkin F. Calvo, J. Hernandez-Rojas, and T. F. Middleton. <http://www-wales.ch.cam.ac.uk/CCD.html>. 2005.
- [54] S. Antonczak, M.F. Ruiz-Lopez, and J.L. Rivail. Ab initio analysis of water-assisted reaction mechanisms in amide hydrolysis. *J. Am. Chem. Soc.*, 116:3912, 1994.
- [55] G. A. Cisneros, M. Wang, P. Silinski, M.C Fitzgerald, and W. Yang. The protein backbone makes important contributions to 4-oxalocrotonate tautomerase enzyme catalysis: Understanding from theory and experiment. *Biochemistry*, 43:6885, 2004.
- [56] Anglada JM Bofill JM. Finding transition states using reduced potential-energy surfaces. *Theo. Chem. Acc.*, 105:463, 2001.
- [57] D. G. Truhlar and B. C. Garret. Variational transition-state theory. *Acc. Chem. Res.*, 13:440, 1980.

- [58] E. Kraka. *Encyclopedia of Computational Chemistry v.2 edited by P.v.R. Schleyer, N.L. Allinger, P.A. Kollman, T. Clark, H.F. Schaefer II, J. Gasteiger, and P.R. Schreiner.* John Wiley and Sons, Chichester, NY, 1998.
- [59] B.K. Dey, M.R. Janicki, and P.W. Ayers. Hamilton-jacobi equation for the least-action/least-time dynamical path based on fast marching method. *J. Chem. Phys.*, 121:6667, 2004.
- [60] C. Gonzalez and H. B. Schlegel. Improved algorithms for reaction-path following - higher-order implicit algorithms. *J. Chem. Phys.*, 95:5853, 1991.
- [61] C. W. Gear. *Numerical Initial Value Problems in Ordinary Differential Equations.* Prentice-Hall Inc., Englewood Cliffs, NJ, 1971.
- [62] K.K. Baldridge; M.S. Gordon; R. Steckler and D.G. Truhlar. Ab initio reaction paths and direct dynamics calculations. *J. Chem. Phys.*, 93:5107, 1989.
- [63] H. P. Hratchian and H. B. Schlegel. Accurate reaction paths using a hessian based predictor-corrector integrator. *J. Phys. Chem.*, 120:9918, 2004.
- [64] K. Dekker and J.G. Verwer. *Stability of Runge-Kutta methods for stiff nonlinear differential equations.* Elsevier Science, New York, 1984.
- [65] C. Gonzalez and H.B. Schlegel. An improved algorithm for reaction-path following. *J. Chem. Phys.*, 90:2154, 1989.
- [66] H.B. Schlegel. Some thoughts on reaction-path following. *J. Chem. Soc. Faraday Trans.*, 90:1569, 1994.
- [67] A. A. Medovikov. High order explicit methods for parabolic equations. *BIT*, 38 No.2, 1998.
- [68] A. Baboul and H.B. Schlegel. Improved method for calculating projected frequencies along a reaction path. *J. Chem. Phys.*, 107:9413, 1997.
- [69] J.J.J.P Stewart, L.P. Davis, and L.W. Burggraf. Semiempirical calculations of molecular trajectories - method and applications to some simple molecular-systems. *J. Comp. Chem.*, 8:1117, 1987.
- [70] S.A. Maluendes and M. Dupuis. A dynamic reaction coordinate approach to ab initio reaction pathways - application to the 1,5 hexadiene cope rearrangement. *J. Chem. Phys.*, 93:5902, 1990.
- [71] T. Taketsugu and M.S. Gordon. Dynamic reaction-path analysis based on an intrinsic reaction coordinate. *J. Chem. Phys.*, 103:10042, 1995.
- [72] H.P. Hratchian and H.B. Schlegel. Following reaction pathways using a damped classical trajectory algorithm. *J. Phys. Chem. A*, 106:165, 2002.
- [73] G. Wanner and E. Hairer. *Solving Ordinary Differential Equations II 2nd Ed.* Springer, NY, 1993.

- [74] B.P. Sommeijer, L.F. Shampine, and J.G. Verwer. *RKC: an Explicit Solver for Parabolic PDEs. Technical Report MAS-R9715 CWI*. Amsterdam, 1997.
- [75] S.K. Burger and W. Yang. A combined explicit-implicit method for high accuracy reaction path integration. *J. Chem. Phys.*, 124:224102, 2006.
- [76] K. Burrage and J.C. Butcher. Stability-criteria for implicit runge-kutta methods. *SIAM J. Numer. Anal.*, 16:46, 1979.
- [77] R. Alexander. Diagonally implicit runge-kutta methods for stiff odes. *SIAM J. Numer. Anal.*, 14:1006, 1977.
- [78] A. Kvrn. Singly diagonally implicit runge-kutta methods with an explicit first stage. *BIT*, 44:489, 2004.
- [79] K. Burrage, J. C. Butcher, and F. H. Chipman. An implementation of singly-implicit runge-kutta methods. *BIT*, 20:326, 1980.
- [80] S.P. Nørsett, P. Syvert, and P.G. Thomsen. Local error control in sdirk-methods. *BIT*, 26:100, 1986.
- [81] N. V. Shirobokov. A definition of stiff differential problems. *Comput. Math. Math. Phys.*, 42:974, 2002.
- [82] A. H. Al-Rabeh. Optimal order diagonally implicit runge-kutta methods. *BIT*, 33:620, 1993.
- [83] G. J. Cooper and A. Sayfy. Semi-explicit a-stable runge-kutta methods. *Math. Comp.*, 33:541, 1979.
- [84] J. C. Butcher and D.J.L. Chen. A new type of singly-implicit runge-kutta method. *Appl. Numer. Math.*, 34:179, 2000.
- [85] S.P. Nørsett, P. Syvert, and P.G. Thomsen. Embedded sdirk-methods of basic order 3. *BIT*, 24:634, 1984.
- [86] J. R. Cash. Diagonally implicit runge-kutta formulas with error estimates. *J. Inst. Maths Applics*, 24:293, 1979.
- [87] S.P. Nørsett. *Semi-explicit Runge-Kutta Methods – Report Mathematics and Computation No. 6/74*. University of Trondheim, 1974.
- [88] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. *Numerical Recipes in FORTRAN*. Cambridge University Press, Cambridge, 1992.
- [89] G. Wanner and E. Hairer. *Solving Ordinary Differential Equations I 2nd Ed*. Springer, NY, 1993.
- [90] A. Medovikov. <http://www.math.tulane.edu/~amedovik/>. 2005.
- [91] <http://www.netlib.org>. 2007.

Biography

Born: March 21, 1977; Chicago, IL, USA

EDUCATION

B.Sc. in Chemistry, University of Pennsylvania, May 2000.

Ph.D. in Chemistry, Duke University, July 2007.

PUBLICATIONS

Burger S.K., Yang W., Quadratic string method for determining the minimum-energy path based on multiobjective optimization, *J. Chem. Phys.*, 124, 054109, 2006.

Burger S.K., Yang W., A combined explicit-implicit method for high accuracy reaction path integration, *J. Chem. Phys.*, 124, 224102, 2006.

Burger S.K., Yang W., Automatic integration of the reaction path using diagonally implicit Runge-Kutta methods, *J. Chem. Phys.*, 125, 224108, 2006.

Burger S.K., Yang W., Sequential Quadratic Programming Method for Determining the Minimum Energy Path, *J. Chem. Phys.*, submitted.

Hu, H., Lu, Z., Parks J., **Burger S.K.**, Yang W., QM/MM Minimum Free Energy Path for Accurate Reaction Energetics in Solution and Enzymes: Iterative Optimization on the Potential of Mean Force Surface, *J. Chem. Phys.*, submitted.

AWARDS

Duke Chemistry Fellowship. , May 2001–July 2007.

Duke Graduate Student Travel Award. 2004 and 2006.

Hamilton Foundation Fellowship 2007–