# Parallel Implementation of Nudged Elastic Band Method

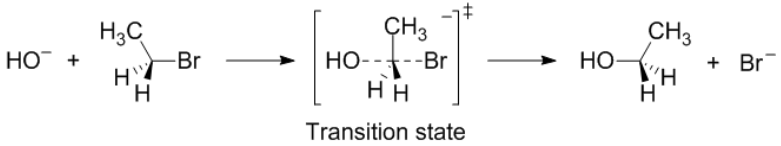
## 6.338 Final Project

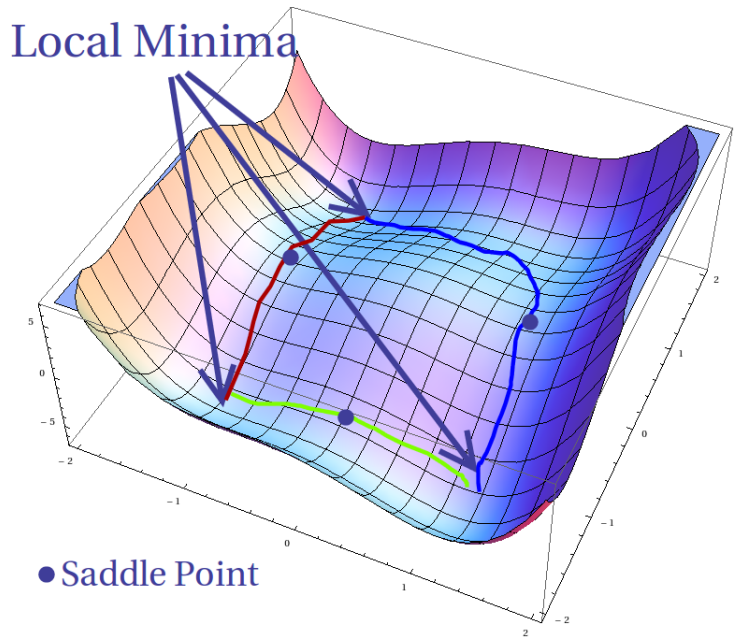
Anubhav Sinha

December 14, 2011

# Background

- Methods in computational chemistry give potential energy surfaces (PES).
- Purpose: find the transition state energy barrier and the intrinsic reaction coordinate or minimum energy path (MEP) between two stable configurations.



# Formal Problem Statement

## Problem

Determine the true MEP.

- The path $q$ has endpoints $\mathbf{q_i}$ and $\mathbf{q_f}$ that are local minima of the PES.
- The path is parametrized by some variable $s$ that goes from 0 to 1.
- At every point in the path, $\frac{d\mathbf{q}}{ds}$ is parallel to the gradient of the PES.
- The MEP passes through at least one saddle point since both $\mathbf{q_i}$ and $\mathbf{q_f}$ are minima.

- Very Computationally Intensive
  - Relaxed PES Scan: fix internal coordinate or linear combination of internal coordinates (i.e. bond lengths), and optimize other coordinates.
  - Hypersphere Search: locate all energy minima on hypersphere with dimensionality same as coordinate space and given radius; trace minima as a function of the hypersphere radius.
- Less Computationally Intensive
  - Eigenvector Following Method: compute Hessian matrix and gradient, take steps in all directions of eigenvectors except the one associated with the smallest eigenvector.
  - Gradient Extremal Following Method: follow Gradient Extremal Path
- Better Methods
  - String Method
  - Plain Elastic Band
  - Nudged Elastic Band

# Nudged Elastic Band Method

## NEB

Chain of States Method: $S^{\mathrm{NEB}}(\vec{Q}_i, \ldots, \vec{Q}_f)$.

- Come up with initial interpolation with $N$ images between fixed endpoints $\vec{Q}_i$ and $\vec{Q}_f$.
- Connect adjacent images with spring force: $\vec{F}_i^S = k(\vec{Q}_{i+1} - \vec{Q}_i) - k(\vec{Q}_i - \vec{Q}_{i-1})$.
- Compute force due to potential: $\vec{F}_i^V = -\nabla V(\vec{Q}_i)$
- Compute modified unit "tangent" $\vec{\tau}$
- Compute the force used $\vec{F}_i = \left(\vec{F}_i^V - (\vec{F}_i^V \cdot \vec{\tau})\vec{\tau}\right) + \left((\vec{F}_i^S \cdot \vec{\tau})\vec{\tau}\right) = \vec{F}_i^V|_\perp + \vec{F}_i^S|_\parallel$
- Numerical integration using Steepest Descent; $\vec{Q}_i = \vec{Q}_i + \alpha \cdot \vec{F}_i$

# Details of the Algorithm

## Unit Tangent

$$\vec{\tau} = \begin{cases} \vec{\tau}^{+}\Delta V^{\mathtt{MAX}} + \vec{\tau}^{-}\Delta V^{\mathtt{MIN}} & V_{i+1} > V_{i-1} \\ \vec{\tau}^{+}\Delta V^{\mathtt{MIN}} + \vec{\tau}^{-}\Delta V^{\mathtt{MAX}} & V_{i+1} < V_{i-1} \end{cases}$$

where

$\Delta V^{\mathtt{MAX}} = \max(|V_{i+1} - V_i|, |V_{i-1} - V_i|)$

$\Delta V^{\mathtt{MIN}} = \min(|V_{i+1} - V_i|, |V_{i-1} - V_i|)$

where $\vec{\tau}$ is renormalized.

## Vector Components

- Plain Elastic Band: $\vec{F}_i = \vec{F}_i^{V} + \vec{F}_i^{S}$
- Nudged Elastic Band $\vec{F}_i = \vec{F}_i^{V}|_{\perp} + \vec{F}_i^{S}|_{\parallel}$

Fix this by "Nudging"; vector projections decouple dynamics of point distribution along path from dynamics of path itself.

# Advantages and Disadvantages of NEB

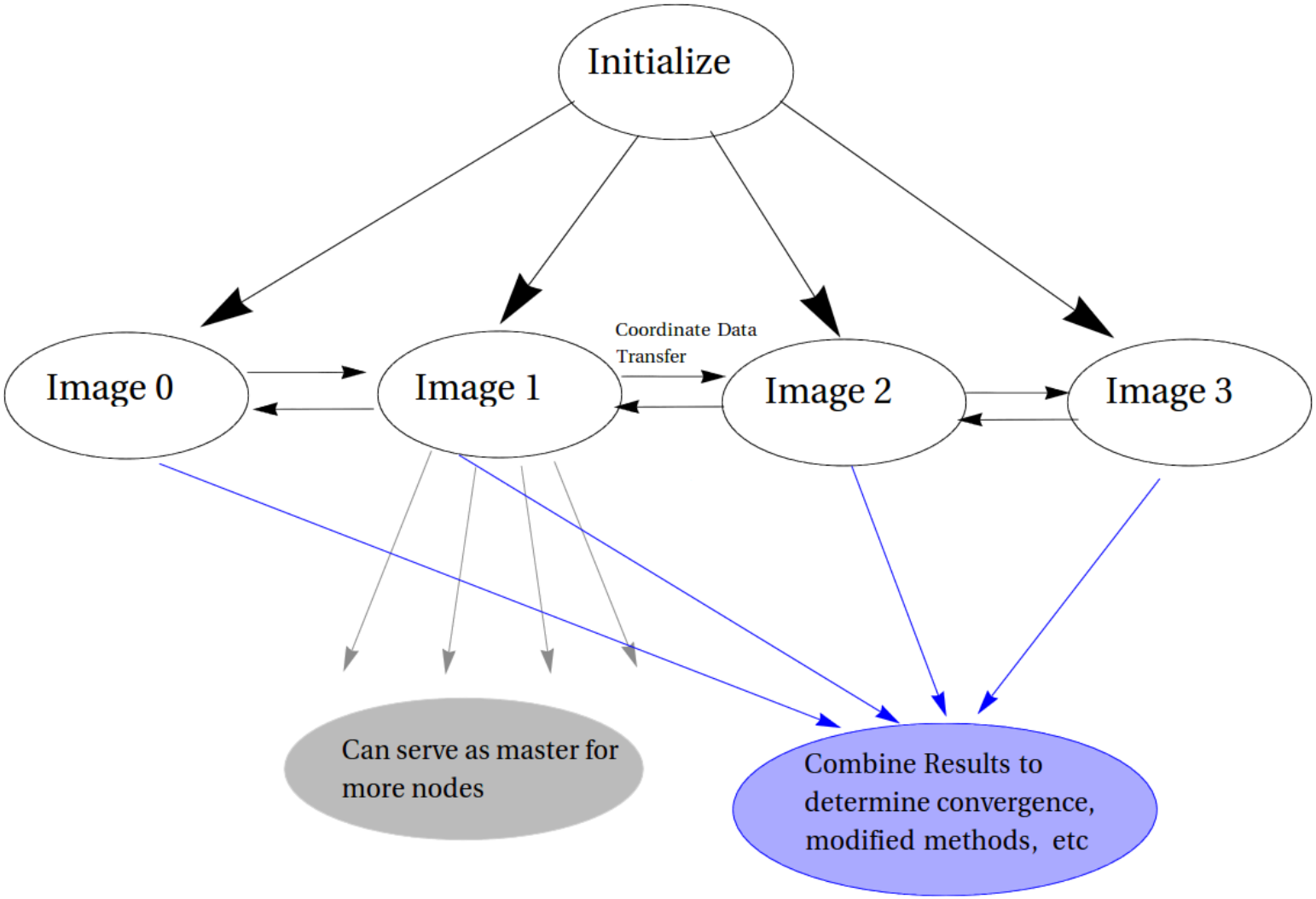
## Advantages

Converges to MEP
Does not require second derivative information
Embarrassingly parallel

## Disadvantages

Need sufficient number of images along path
Multiple MEPs may exist
Slow convergence

# Parallel Computing



# Implementation

- Using MPI/C++ on the Evolution Cluster
  - 60 nodes
  - 2 x 2-core 2.66 GHz Intel Xeon with 6 GB RAM/node
- Number of processors used and number of images/per processor are variable
- Serial reference implementation

# Parallelization Method

- Simple Parallelization: put one image per node
  - Slower for smaller problems with simpler potentials—MPI Send and MPI Recv are very slow operations.
  - Better for larger, more complex problems.
  - Requires lots of processors.
- More Involved Parallelization: put several images per node so that some processes can communicate directly via memory.
  - Better for simpler potentials; more complex potentials are better served with minimizing number of images per node.

# Outline of Code

- Initialize starting configuration; set up images on processors.
- Repeatedly calculate NEB forces and numerically integrate until convergence; output image data.
  - MPI Send/Recv coordinates of neighboring images to calculate Spring Force.
  - Calculate potential forces and compute vector projections.
  - Numerically integrate and output data.
  - MPI Reduce to determine convergence.

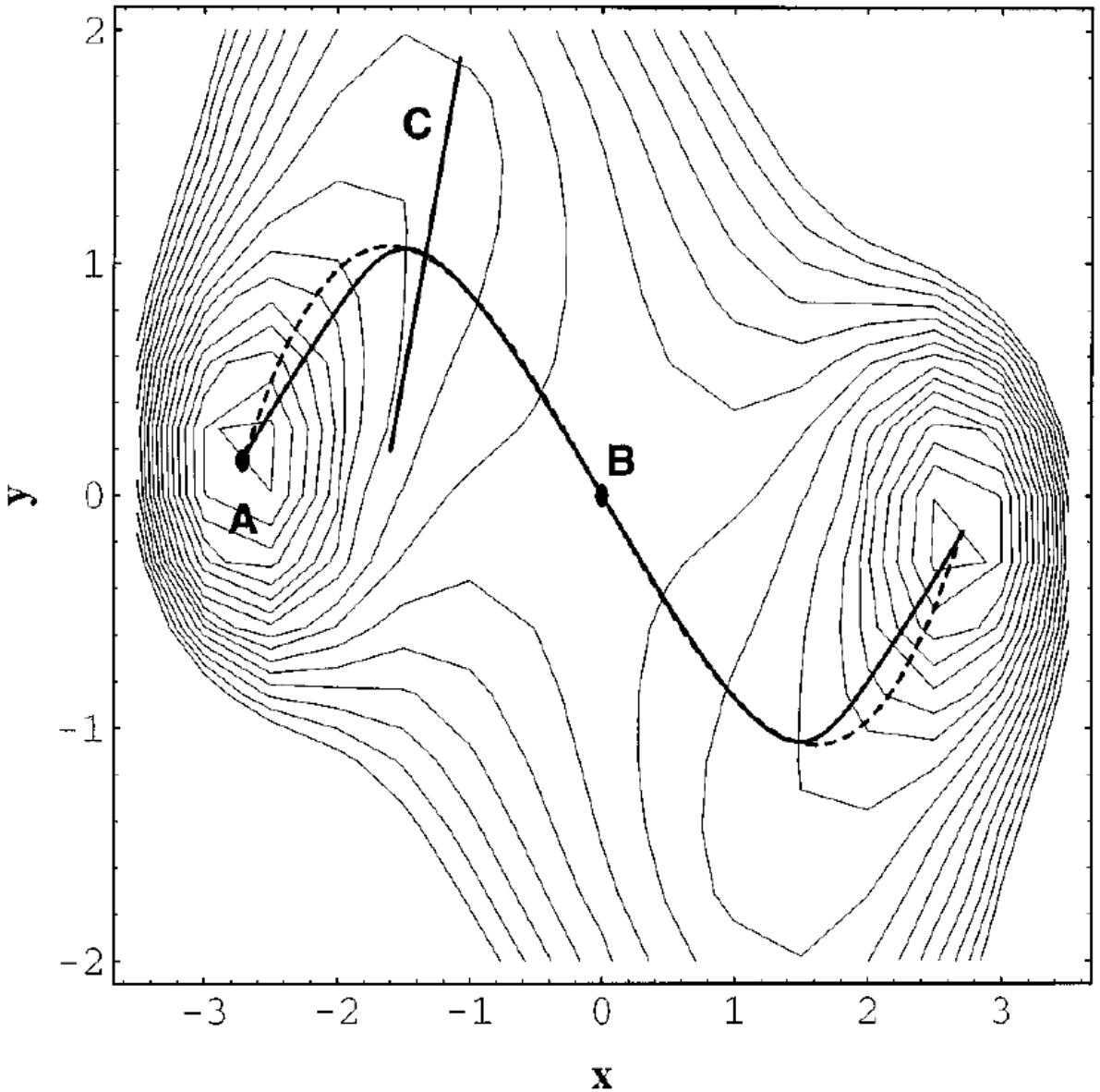
# Test Systems

- Karplus PES
- Wolfe-Quapp PES
- Muller-Brown PES

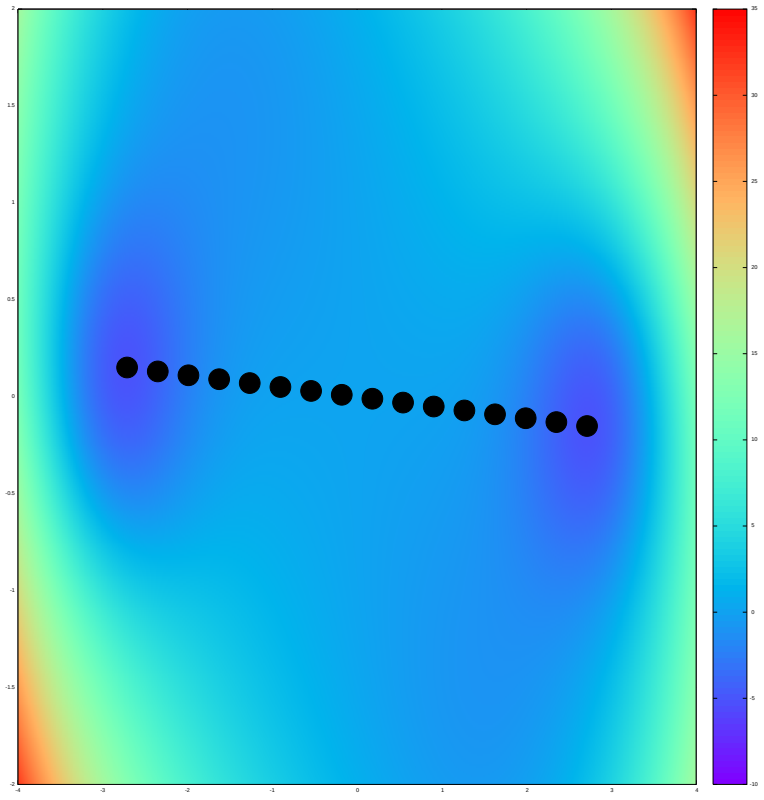
We use these because:

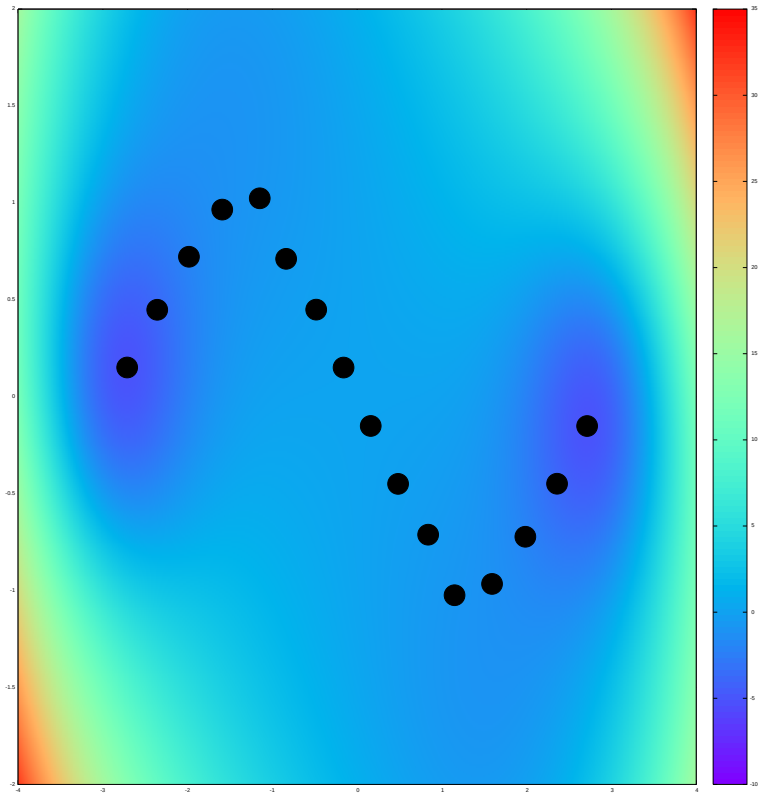
- Canonical simple 2D surfaces
- Easy to visualize

# Karplus PES

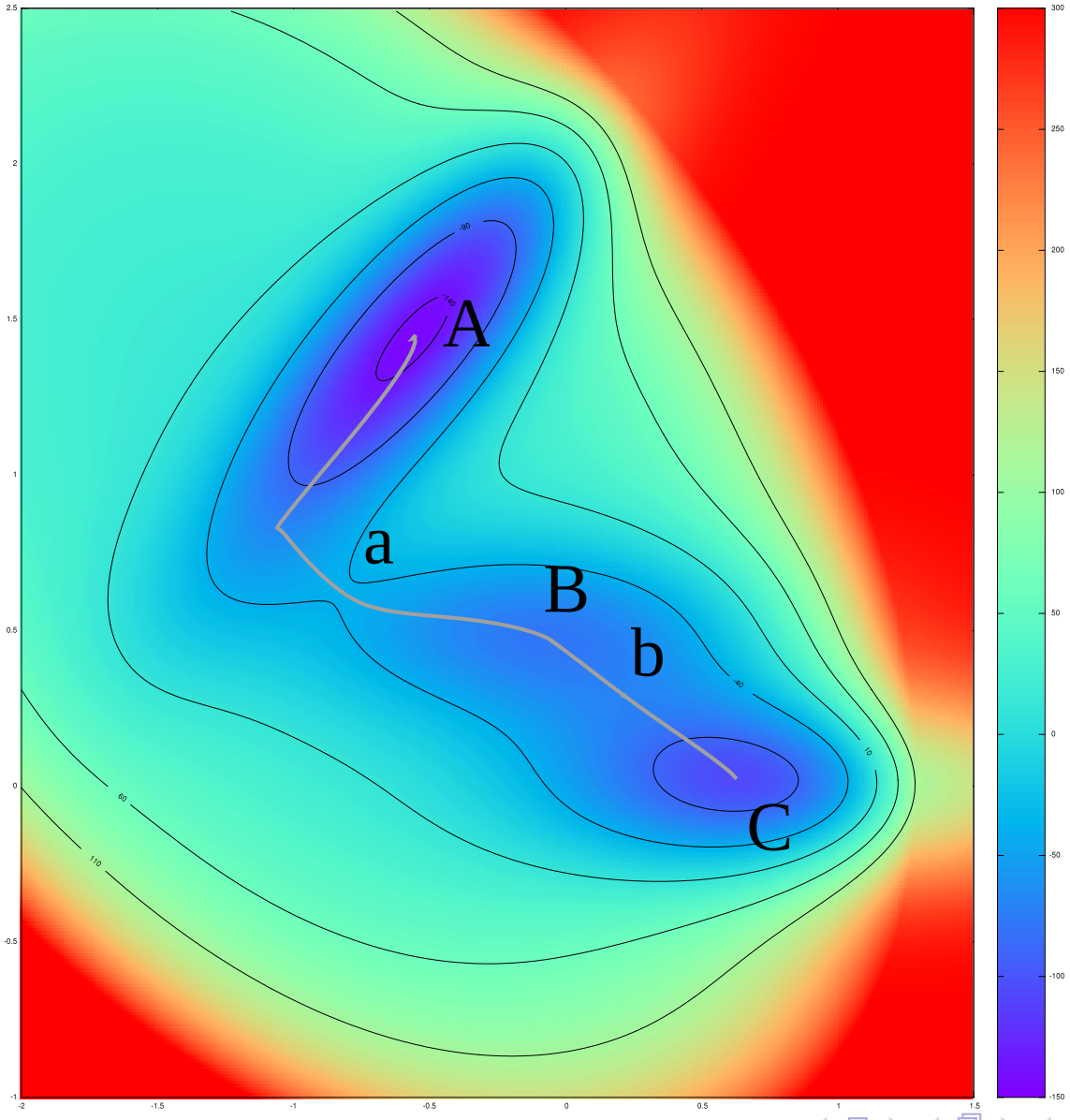


# Karplus MEP

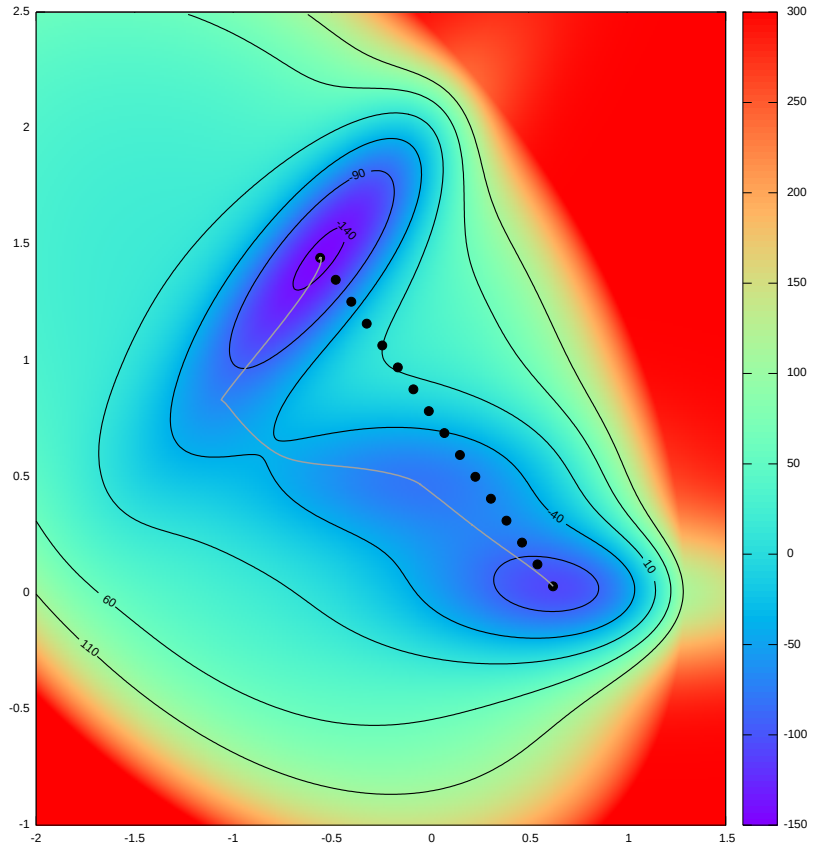


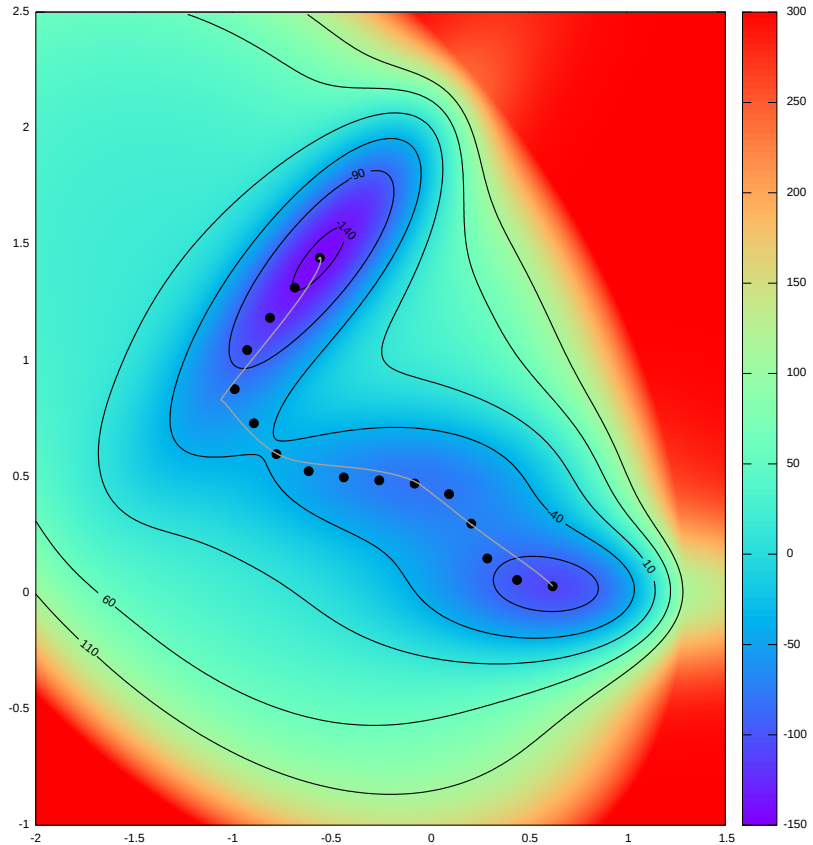


# Muller-Brown PES



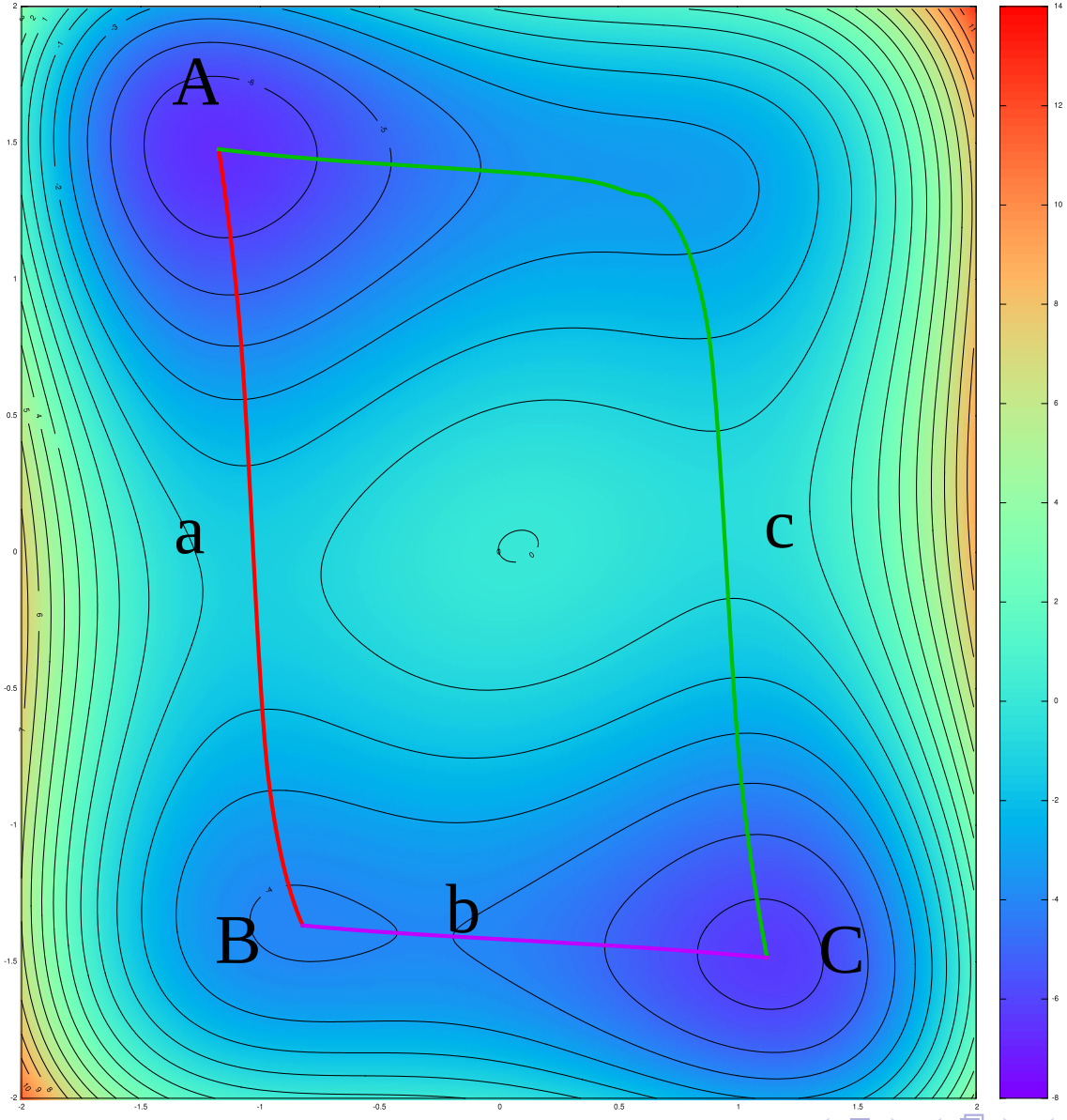
# Muller-Brown MEP



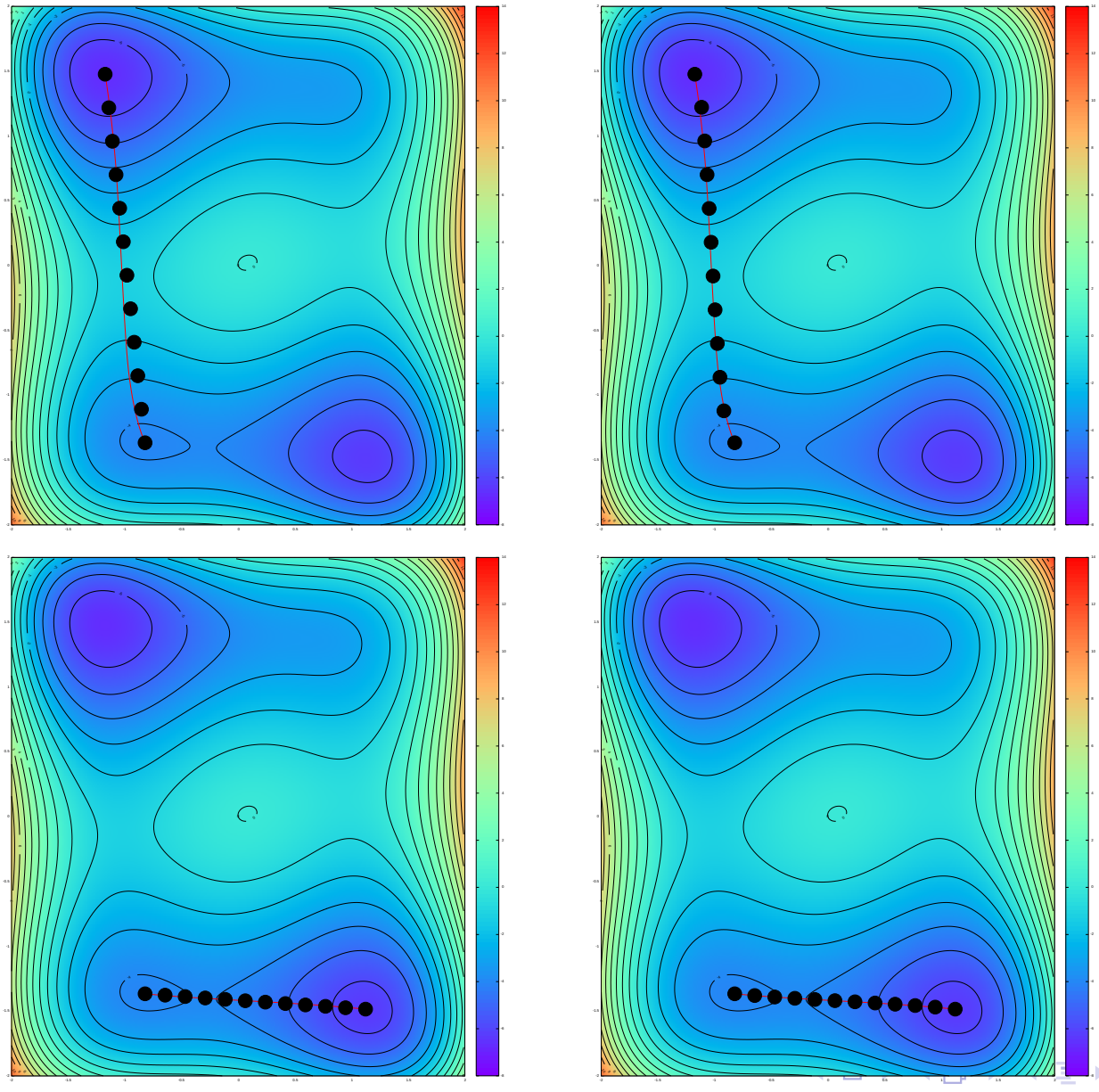




http://web.mit.edu/anusinha/www/6.338/MB.gif

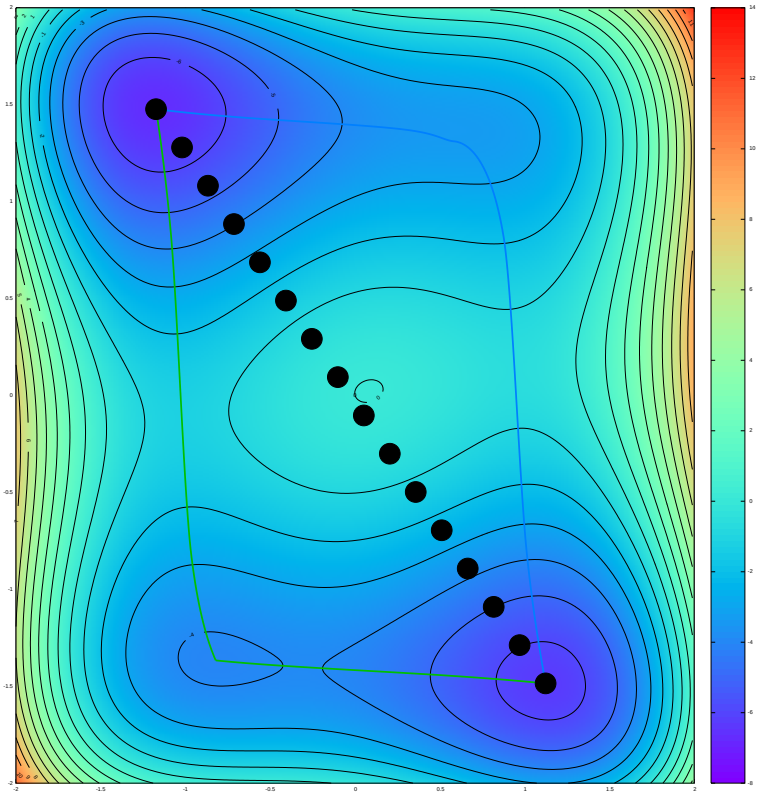
# Wolfe-Quapp PES

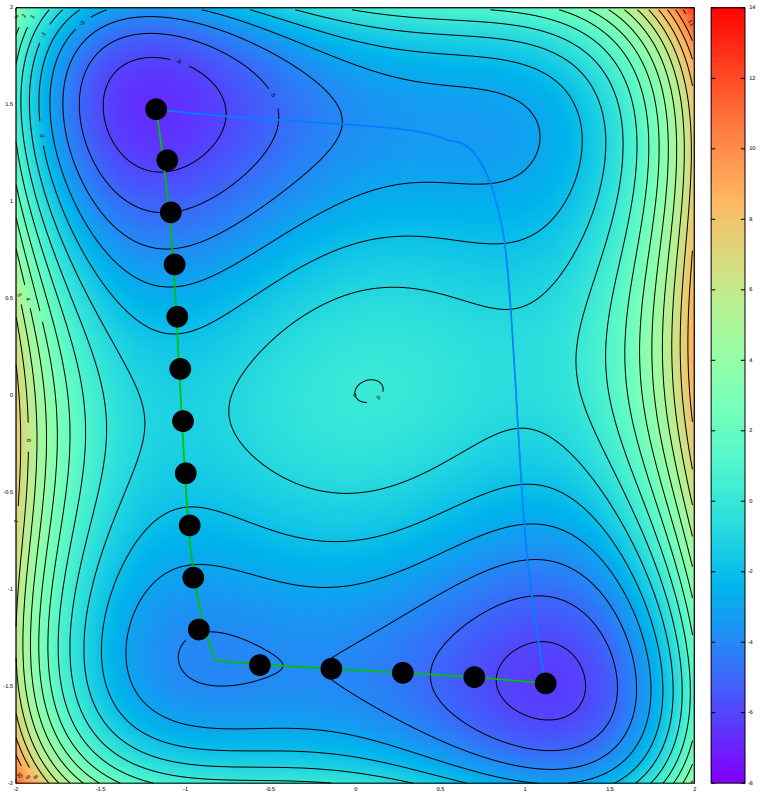


# Wolfe-Quapp MEPs



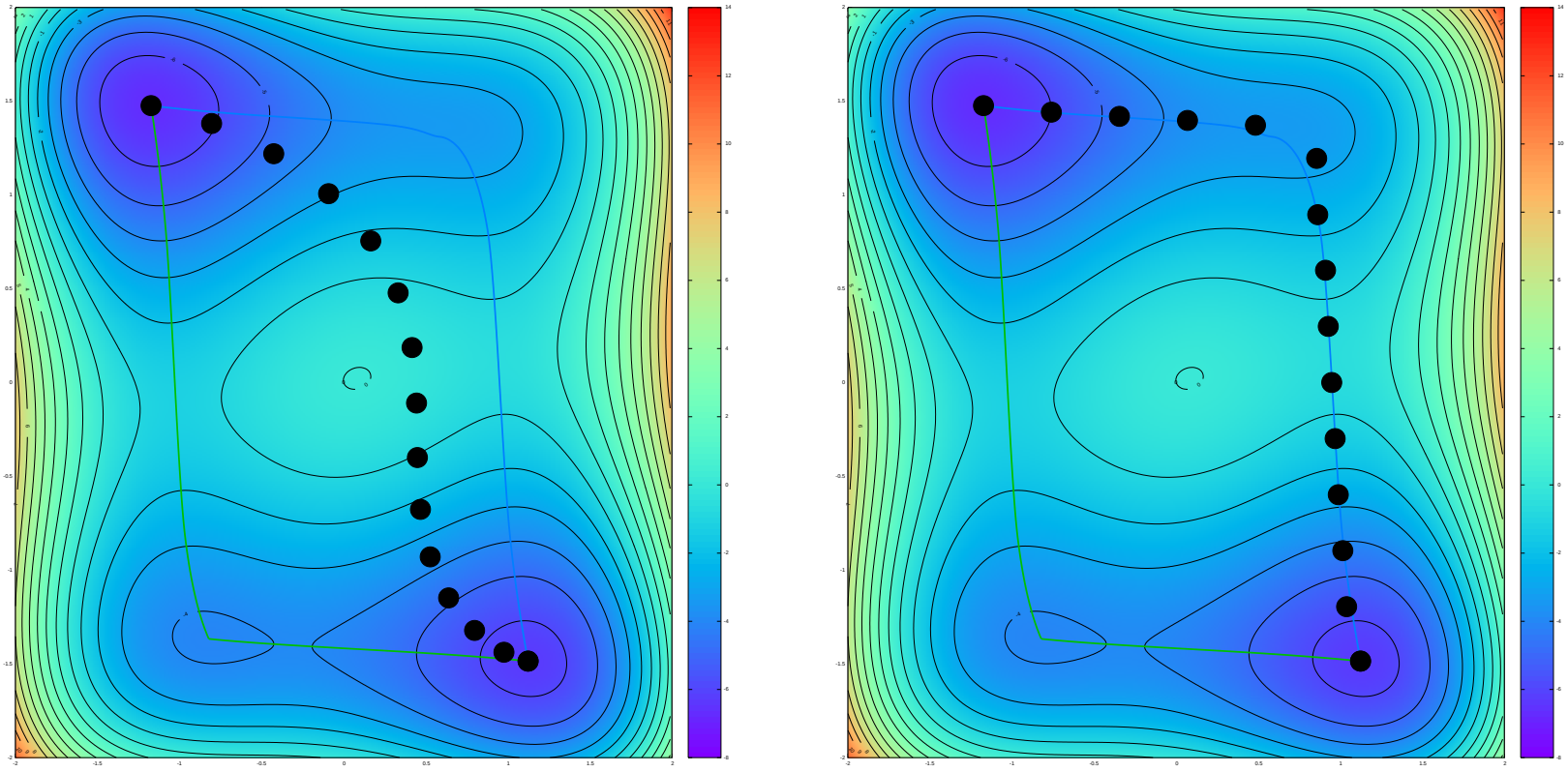
# Wolfe-Quapp MEPs





http://web.mit.edu/anusinha/www/6.338/AC_1.gif

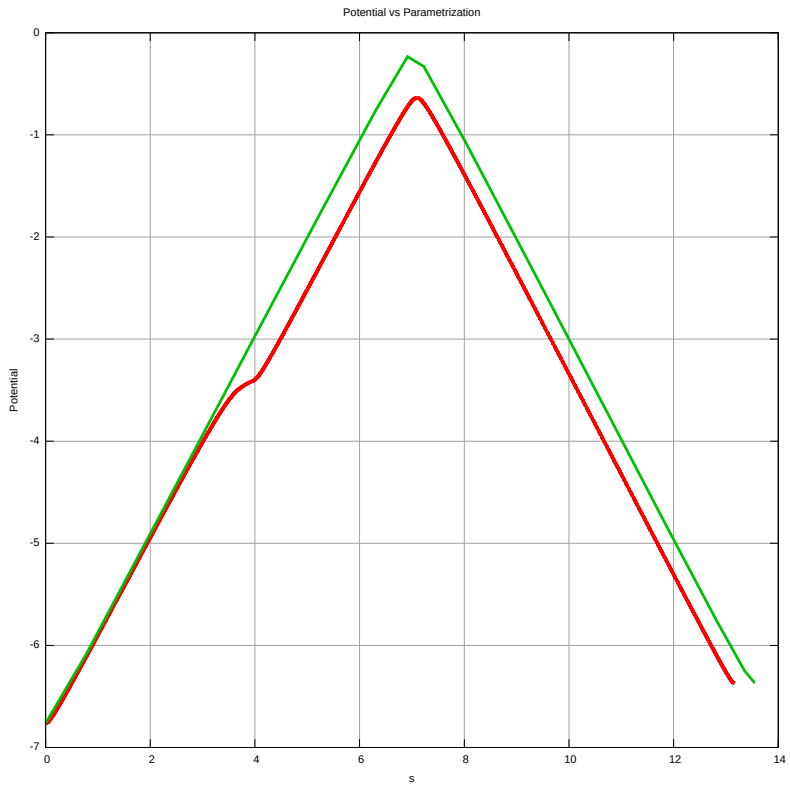
# Wolfe-Quapp MEPs



http://web.mit.edu/anusinha/www/6.338/AC_2.gif

# Accuracy of Convergence

Sample Potential vs Arclength plot for Wolfe-Quapp potential, long path from A to C through saddle point c. All the potential vs arclength paths that NEB finds come very close to their corresponding Newton-Raphson paths.



# Analytic Potentials

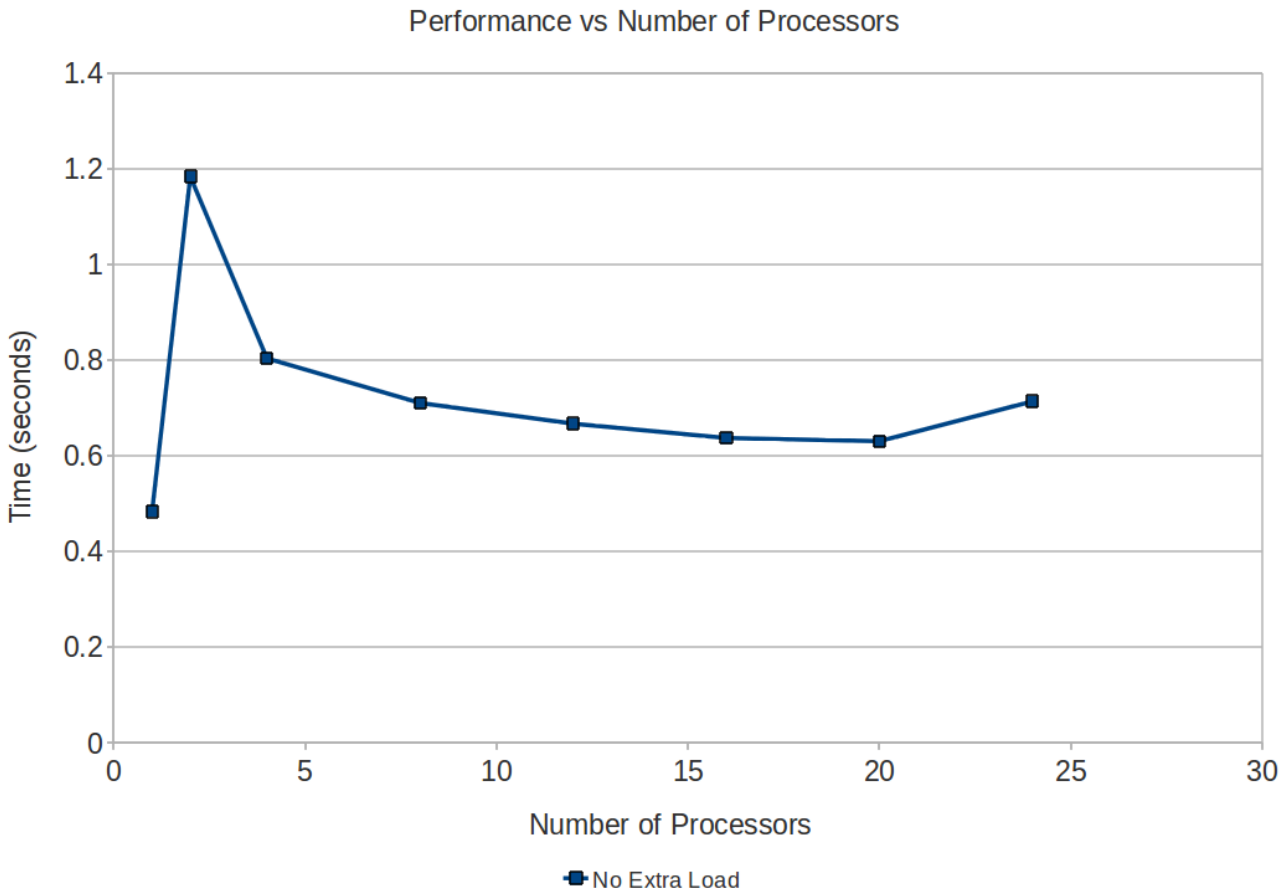
Simple potential energy surfaces used

- Karplus:
  $V(x,y) = 0.6(x^2+y^2)^2 + xy - 9(e^{-(x-3)^2-y^2} + e^{-(x+3)^2-y^2})$
- Muller-Brown:
  $V(x,y) = \sum_{i=0}^{i=3} A_i e^{a_i(x-x_i^0)^2 + b_i(x-x_i^0)(y-y_i^0) + c_i(y-y_i^0)^2}$
- Wolfe-Quapp:
  $V(x,y) = x^4 + y^4 - 2x^2 - 4y^2 + xy + 0.3x + 0.1y$

Nonrealistic use case; functions evaluate very quickly so serial code runs faster, since there are no communications at all.

# Simple Runtime

24 images used on Muller-Brown Potential.



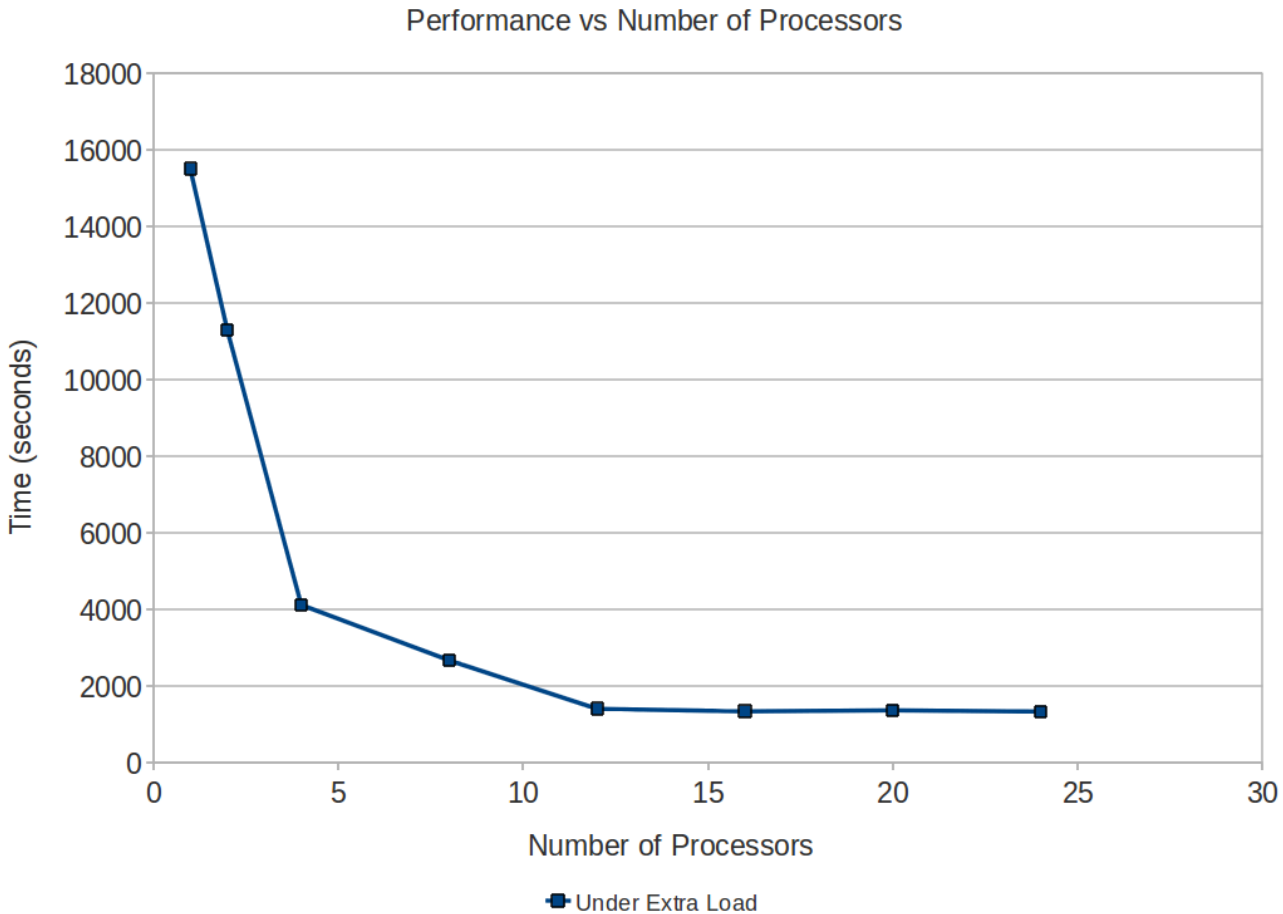
# Simple Runtime Analysis

This is an embarrassingly parallel problem. Why is it so slow?

- The potentials are simple; direct memory accesses are much faster than message passing between multiple separate nodes.
- The peak at two processors is unclear; could be because of laggy node.
- Settles at value with passing; the distributedness of the passing doesn't matter. Let's add a delay to the potential energy calls to simulate a realistic test case.

# Realistic Use Case

We add a delay to simulate the use of this program on a real system.



# Summary of Results

Implemented parallelized version of NEB algorithm.

- Tested for accuracy of method on canonical 2D PES.
- Runtime decreases on realistic test cases.
- Have easily modifiable framework for testing variants of chain-of-states methods.

# Future Work

- More complex potential surfaces; code is written is written for $n$-dimensional potentials.
  - First step: Lennard-Jones potential: $V_{LJ} = \epsilon \left[ \left( \frac{r_m}{r} \right)^{12} - 2 \left( \frac{r_m}{r} \right)^{6} \right]$
  - Interface with molecular dynamics software
- Test out variants of NEB method
  - Climbing Image NEB
  - Doubly Nudged Elastic Band
  - Related: String method
- Better optimization method—Steepest Descent is simple, but slow. Conjugate Gradient, Fast Inertial Relaxation Engine, L-BGFS are faster.
- Better approximations than simple linear interpolation; $\nabla V \approx \mathbf{H}$ at the endpoints.

# Image Sources

- Transition State: public domain image available at `http://en.wikipedia.org/wiki/File:Transition_State.png`.
- Karplus: E. Neria, S. Fischer, and M. Karplus, J. Chem. Phys. **105**, 1902 (1996).

# Acknowledgements

I'd like to thank Jeff Bezanson, and Prof. Troy Van Voorhis and Prof. Alan Edelman.

Questions?