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

C. T. Kelley, David Mokrauer, Alexei Bykhovski
NC State University
tim_kelley@ncsu.edu
Supported by ARO.

ICNONLA 2011, November 8, 2011
Outline

1. Objectives
   - Chemistry Application
   - Dynamic Formulation

2. Example Butene $C_4H_8$

3. Sparse Interpolation

4. Larger Molecule: Stilbene $C_{14}H_{12}$

5. Software: LITES

6. Conclusions
Molecular Geometry

- Stationary nuclei in cloud of moving electrons.
- $N$ atoms located by $3N - 6$ coordinates (Wilson et al 1955) torsion and bond angles
- Atoms move in response to external forces, such as light.
- Stable configurations (the only ones which occur in nature) are local minimizers of potential energy $\mathcal{E}$. 
Sparse Interpolation

Objectives

Chemistry Application

Potential Energy

\[ \mathcal{E} \text{ depends on} \]
- Configuration \( p \in \mathbb{R}^N \).
- Quantum state \( n = 0, 1, \ldots \)

Quantum states are ordered

\[ \mathcal{E}_0(p) \leq \mathcal{E}_1(p) \leq \mathcal{E}_2(p) \leq \ldots \]

and (as you will hear many more times) the local minimizers of \( \mathcal{E}_n \) are the only configurations in nature.
2-Butene

2-butene has 2 stable geometries for $n = 0$

Figure: trans 2-butene

Figure: cis 2-butene
Excitations

- Begin with a stable ground state \( p \)
  local minimizer of \( \mathcal{E}_0 \)

- Excite the molecule with light
  shifting the state to \( n = 1 \), say

- \( p \) may not be a local minimizer of \( \mathcal{E}_1 \)

This is an opportunity to apply light to change the configuration.
Results of Excitation

Excitation changes orbital, thus changing interatomic forces

**Figure:** Highest Occupied Molecular Orbital

**Figure:** Lowest Unoccupied Molecular Orbital
Our Assignment

Simulate this:

- begin at $p_0$ in state 0
- excite to new state
- relax to local minimum in new state
- emit energy and return to state 0
- relax to minimum $p_f$
- Goal: $p_f \neq p_0$

Applications: sensors, solar power, ...
Fantasy Algorithm

- Compute $\mathcal{E}$ with a good quantum chemistry code. We use Gaussian.
- Compute relaxation after excitation by integrating
  \[ p' = -\nabla \mathcal{E}(p) \]
  and thereby optimize as nature does it.

Using dynamics-unaware method can lead to incorrect results.
Molecules have 100s of atoms and 100s of degrees of freedom (torsion angles).

We can’t vary all of them and must pick a few, and must solve an optimization problem to resolve the rest.

So we need some collaborators to guide us as we reduce the number of degrees of freedom.

The collaborators need us to tell them if they kept to many or omitted some degrees of freedom.
Reduction in Problem Size

- $p = (x, \xi)$ molecular coordinates; $\mathcal{E}(p)$ energy.
- $x$ are the design coordinates
- $E(x)$ energy function of $x$
  - Fix $x$; let Gaussian solve
    $$\min_{\xi} \mathcal{E}(x, \xi)$$
    to find $\xi = \xi(x)$.
- Evaluation of $E$ is very expensive; $\nabla E$ out of the question.
- Internal iteration requires good initial iterate. Failure is very costly.
2-butene Excitation

Success with a single DOF [Luo, Gelmont, Woolard (2007)]

Figure: 1 variable simulation of 2-butene

Can you see why randomness might be needed?
Remarks

- Prior work used internal (expensive) Gaussian continuation.
- Gaussian will only handle one DOF.
- Internal Gaussian optimization can fail.
What we’d like to do.

- Begin with stable (local min) of $E$ in the ground state.
- Excite the molecule and move through a sequence of excited states.
- Simulate relaxation in excited state via $x' = -\nabla E(x)$ and find local min.
- Interrupt dynamics in mid-stream if something interesting happens.
- Return to ground state and (maybe) find different stable state.
- Query the optimization landscape afterwards to
  - Look for design variables that did not change much or slave variables that did.
  - Add some randomness to capture thermal fluctuations.
What we can do. Reduction: Step 2

- Interpolate $E$ in the region of interest with $x_0$ on boundary
  - Evaluate $E$ on a mesh
  - Organize the evaluations so that
    - Internal optimization for $\xi$ converges (ie has good initial data)
    - As many evaluations as possible done in parallel
- Interpolate $E$ to get $E^S$
- Use the interpolant to drive the dynamics and solve
  \[ x' = -\nabla E^S \]

Your bad idea alarm may be going off because we have a …
“Look-Ahead” IVP integrator

- Normally insane for \( u' = f \) because
  - you’d visit places the dynamics never see,
  - waste many calls to \( f \), and
  - the complexity would be a killer for high dimensional problems.

- But we do well with this because
  - serial evaluation of \( f \) in an normal integrator performs poorly,
  - our dimension \( \leq 10 \) is low,
  - our function is REALLY expensive, and
  - look-ahead parallelizes easily.
Example Butene $C_4H_8$; D1/D2 coordinates

Rotate angles D1 (8,6 – 2,1) and D2 (11,8 – 6,2)
Tensor Product Mesh Results
Good Scalability, Too Many Gaussians, Reduction: step 3

- Solution: Local cubic interpolation, one patch at a time.
  - Integrate dynamics on each patch.
  - Locate and evaluate on the next one.
  - Potential for error estimation and control.

- Timings: Full Surface $\approx 2 \times \text{patch}$

- Parallel performance was worse,
  but would improve with more degrees of freedom.

- Exponential complexity still there.
Faster Parallel Evaluation in Two Dimensions
Sparse Interpolation: Reduction: step 4

Tensor product grids are hopeless for more than two design variables.
We shamelessly steal an idea (Smolyak, 1963) from high-dimensional integration.

- Generate mesh in $R^d$ so that integration has degree of exactness $k$
- Symmetric about coordinate lines and diagonals
- Nested meshes as $k$ increases so we can estimate error efficiently.

Current approach: uses patches with sparse grids.
Building a Sparse Interpolation: I

Barthelmann, Novak, Ritter (2000)

- Chebyshev extrema for $m_1 = 1, m_2 = 3, \ldots m_i = 2^{i-1} + 1$
- $X^i = \{x^i_j\}_{j=1}^{m_i}$

$$x^i_j = -\cos\left(\frac{\pi(j - 1)}{m_i}\right).$$

- Let $U^i(f)(x) = \sum_{j=1}^{m_i} f(x^i_j)l^i_j(x)$ where $l^i_j$ are the Lagrange interpolating polynomials.
- And now it’s time for several variables . . .
For $f : \mathbb{R}^d \rightarrow \mathbb{R}$, $\vec{i} = (i_1, \ldots, i_d)$, $|\vec{i}| = \sum_{j=1}^{d} i_j$, $x = (x_1 \ldots x_d)^T$

$$U^\vec{i}(f)(x) = \sum_{j_1=1}^{m_{i_1}} \cdots \sum_{j_d=1}^{m_{i_d}} f(x_{j_1}^{i_1} \ldots x_{j_d}^{i_d})(l_{j_1}^{i_1}(x_1) \ldots l_{j_d}^{i_d}(x_d))$$

Combine a few of the $U^\vec{i}$ interpolants to get ...
Building a Sparse Interpolation: III

\[ A(k, d) = \sum_{k+1 \leq |i| \leq d+k} (-1)^{d+k-|i|} \binom{d-1}{d+k-|i|} \mathcal{U}^i \]

Does all kinds of good stuff:

- Interpolates polynomials of degree \( k \) exactly
- Many fewer high-order cross terms than tensor products
- Uses sparse grids
Complexity: $x \in R^d$

- Best possible for degree $k$

\[
\binom{d + k}{k} \approx \frac{d^k}{k!} \text{ for large } d
\]

- Tensor product grid has $(k + 1)^d$ points.
- Sparse grid has $\approx 2^k d^k / k!$ points

Cubic interpolation for varying $d$.

<table>
<thead>
<tr>
<th>Dimension</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grid Size</td>
<td>29</td>
<td>69</td>
<td>137</td>
<td>241</td>
<td>389</td>
<td>589</td>
<td>849</td>
<td>1177</td>
<td>1581</td>
</tr>
</tbody>
</table>
Two-dimensional Sparse Grid
Three-dimensional Sparse Grid
Algorithm

- Begin with $x_c$, $E(x_c)$, and patch size $h$
- Interpolate on patch to build surrogate.
  
  If evaluation fails, reduce $h$ and try again.
- Integrate dynamics using surrogate $E^S$ until either
  - you hit a patch boundary or
  - the integration terminates at a local minimum (of the surrogate).
- Figure out how well you did and adjust the patch size.
Method until September 2011.

- Evaluate $E$ at the end of the path $x_+$
- Compare $ared = E(x_c) - E(x_+)$ to $pred = E(x_c) - E^S(x_+)$
- We want $ared / pred$ to be near 1
  - If $ared / pred$ is near 1 and $x_+$ is on the boundary, $h \rightarrow 2h$.
  - If $ared / pred$ is far from 1 $h \rightarrow h / 2$.
  - Otherwise leave $h$ alone.

This is a crude form of error estimation.

Problem: evaluation at interpolation nodes fails too often.
Computing: I

- **Hardware:** IBM Blade Center
  - 32 blades; dual quad-core Xenons = 256 cores
  - Infiniband network

- **Parallelism:**
  - Energy evaluations at each node in parallel.
  - Gaussian uses 4-way parallelism on each blade
  - 64 Gaussian calls at once. Managed with Python.
Example Butene $C_4H_8$; D1/D2/D3 coordinates

Rotate angles D1 (8,6 – 2,1), D2 (12,8 – 6,2), D3 (4,1 – 2,6)
Patch from 2-butene Computation
Success
## Parallel Performance

<table>
<thead>
<tr>
<th>degree</th>
<th># points</th>
<th>cores</th>
<th>time(sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>25</td>
<td>13</td>
<td>190.96</td>
</tr>
<tr>
<td>3</td>
<td>69</td>
<td>39</td>
<td>194.05</td>
</tr>
</tbody>
</table>
Something Larger: \((C_{14}H_{12})\) Stilbene-1
Something Larger: Stilbene-2
Sparse Interpolation

Larger Molecule: Stilbene $C_{14}H_{12}$

Something Larger: Stilbene-3
Sparse Interpolation

Larger Molecule: Stilbene $C_{14}H_{12}$

Something Larger: Stilbene-4
Something Larger: Stilbene-5
Larger Molecule: Stilbene $C_{14}H_{12}$
### Bottom Line

<table>
<thead>
<tr>
<th></th>
<th>D1</th>
<th>D2</th>
<th>D3</th>
<th>D4</th>
<th>D5</th>
<th>Energy</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Start</strong></td>
<td>-6.00</td>
<td>140.00</td>
<td>-40.00</td>
<td>140.00</td>
<td>-40.00</td>
<td>-2343.94</td>
</tr>
</tbody>
</table>
Larger Molecule: Stilbene $C_{14}H_{12}$
Sparse Interpolation

Larger Molecule: Stilbene $C_{14}H_{12}$

NEW
Patch Size Control II

Respond to Internal Optimization:

- Internal iteration limit (BFGS) of 30.
- Shrink patch when max observed count $> 15$
  Shrink by how much?
- Allow patch to grow when max observed count $< 5$
  Growth will also depend on error estimation.
Degree $k$ sparse interpolant $E^k$ has error

$$\epsilon_k = O(h^{k+1})$$

and $\nabla E^k$ has order $O(h^k)$.

Grids $\Omega^k$ are nested ($\Omega^k \subset \Omega^{k+1}$).

Estimate error of degree $k$ approximation before the integration:

- Compute $E^k$ and $E^{k+1}$ from the evaluations on $\Omega^{k+1}$,
- Computed estimate error $\bar{\epsilon}_k$ by $\|E^k - E^{k+1}\|_\infty$ over most recent trajectory
Patch Size Control IV

Then,

- Base the integration on $\nabla E^{k+1}$ (order extrapolation).
- Use RK45 ideas to control patch size from desired error.

Example: Want energy error $\leq \tau$.

If $\bar{\epsilon}_k \approx \epsilon_k \approx Ch^{k+1}$ then $C \approx \bar{\epsilon}_k/h^{k+1}$.

Pick $h_{\text{new}}$ to force $\epsilon_k \leq \tau$ by $h_{\text{new}} < (\tau/C)^{1/(k+1)}$. 
Light-Induced Transition Effects Simulator

- Python driver + numerics
- Error and patch-size control
- Manages calls to Gaussian in parallel
- Documentation aimed at chemists. Now in beta-test with a real chemist.
- Uses Gaussian data structure: Z-matrix
  Easy to swap Gaussian for something else.
LITES I/O

- **Input:**
  - ordered list of excited states beginning and ending at 0
  - Initial ground state configuration

- **Output:**
  - Final point + energy + configuration
  - Z-matrix at each interpolation node
  - Z-matrix at start/end of each patch
LITES Results: 2-butene Revisited
2-butene 3D Simulation Accuracy

$$\delta = 10^{-3}$$

Figure: Actual vs Approximate Error on each Patch
LITES Results: Azobenzene

Figure: Cis-Azobenzene

Figure: Trans-Azobenzene
Cis-Azobenzene Simulation

Figure: Cis returns to Cis
Trans-Azobenzene Simulation

Figure: Trans returns to Trans

Kelley, Mokrauer, Bykhovski

Sparse Interpolation

Software: LITES

Xiamen, 2011 53 / 57
Remarks

- The tool works, but we did not get what we wanted.
- Chemistry expertise is needed to
  - identify the correct design variables,
  - chose the sequence of exited states, and
  - figure out the frequency and pulse rate to get to these states.
Future

- Include thermal effects (randomness).
- Query patches for low barriers (more randomness).
- Experiments.
Opportunities in Quantum Chemistry/Physics

- Many opportunities for mathematicians.
- You must have professional help (ie collaborators).
- Learning curve is steep (be patient).
- You will function with very incomplete information.
Conclusions

- Sparse interpolatory surrogates for molecular simulation.
- Application: sensors, solar power
- Natural parallel evaluation of Gaussian.
- Good speedup for 100s of cores.