S4443
Towards Real-Time Nanostructure Prediction with GPUs

Abhinav Sarje

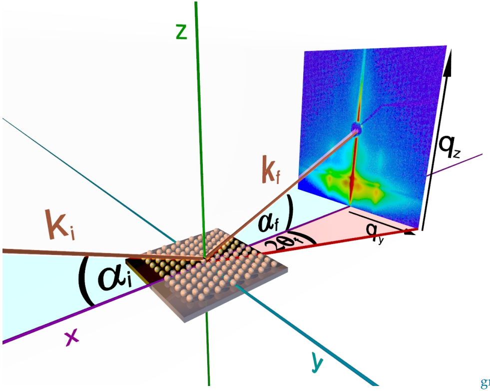
Computational Research Division
Lawrence Berkeley National Laboratory

03.25.14
GPU Technology Conference 2014
San Jose, CA

High-energy X-ray Scattering

- X-ray scattering to measure structural properties of materials, and
- Characterize macromolecules and nano-particle systems at micro and nano-scales.
  - probing the electronic structure of matter,
  - semiconductors,
  - 3D-biological imaging,
  - protein crystallography,
  - chemical reaction dynamics,
  - biological process dynamics,
  - optics,
  - ... and so on.
- Broad variety of applications. E.g.:
  - Materials: Design of energy efficient devices like solar cells, high-density storage media
  - Medicine: Design of synthetic enzymes, drugs and bio-membranes.

High-energy X-ray Scattering



graphic: courtesy of A. Meyer, *www.gisaxs.de*

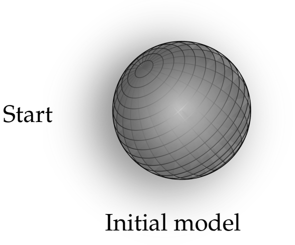
Examples:

- Small-angle X-ray Scattering (SAXS)
- Grazing Incidence SAXS (GISAXS).

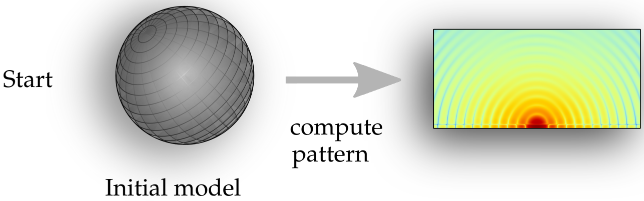
Outline

1. Computational problems in nanostructure prediction.
   - Scattering pattern simulations.
   - Inverse modeling or fitting.
2. Motivations and the need of HPC.
3. X-ray scattering simulations.
4. Performance results.
5. Nanostructure prediction with inverse modeling.
   - With Reverse Monte Carlo simulations.
   - With Particle Swarm Optimization.
6. Conclusions and ending notes.

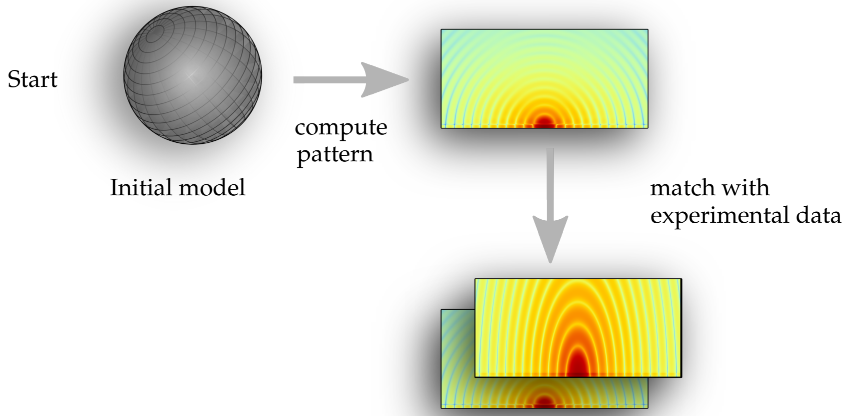
Computational Problems in Structure Prediction



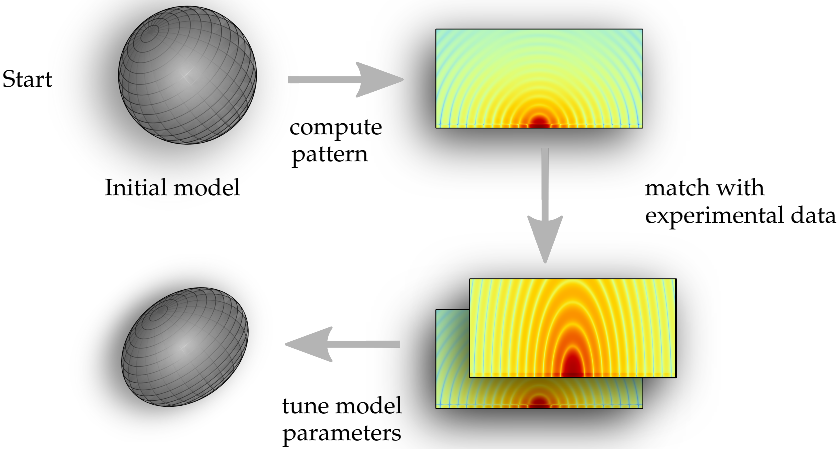
Computational Problems in Structure Prediction



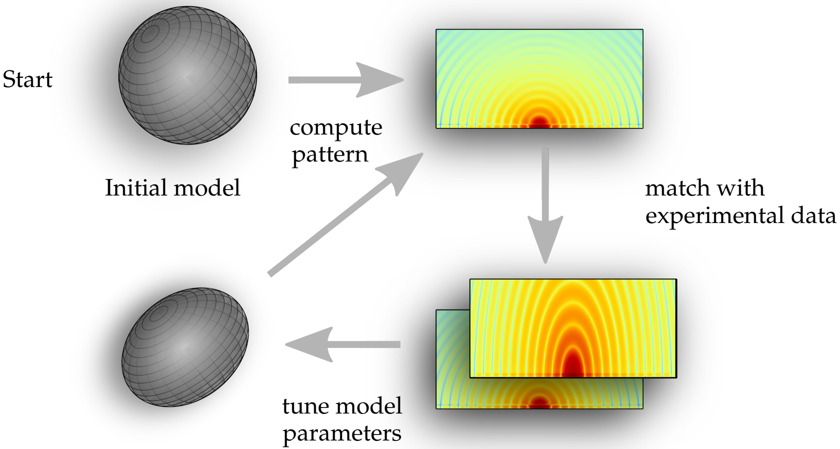
Computational Problems in Structure Prediction



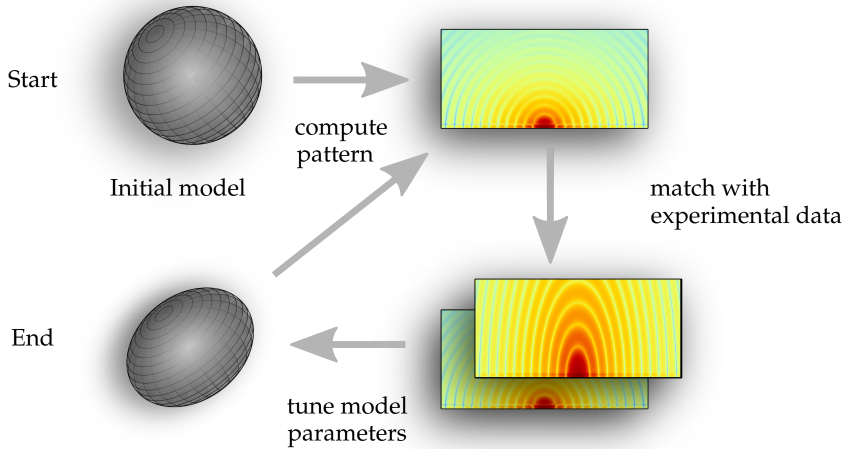
Computational Problems in Structure Prediction



Computational Problems in Structure Prediction



Computational Problems in Structure Prediction



Need for High-Performance Computing

Data generation and processing gap:

- High measurement rates of current state-of-the-art light beam detectors.
- Wait for days for analyzing data with previous softwares.
- Extremely inefficient utilization of facilities due to mismatch.
- *Example*: 100 MB raw data per second. Up to 12 TB per week.

Need for High-Performance Computing

High computational and accuracy requirements:

- Errors are proportional to the resolutions of various computational discretization.
- Higher resolutions require higher computational power.
- Example:
  - $O(10^7)$ to $O(10^{14})$ kernel computations for one simulation.
  - $O(10^2)$ experiments per material sample.
  - $O(10^2)$ to $O(10^3)$ forward simulations for inverse modeling per scattering pattern.

Need for High-Performance Computing

Science Gap:

- Beam-line scientists lack access to high-performance algorithms and codes.
- In-house developed codes limited in compute capabilities and performance.
- Also, they are extremely slow – wait for days and weeks to obtain basic results.

Fortunately ...

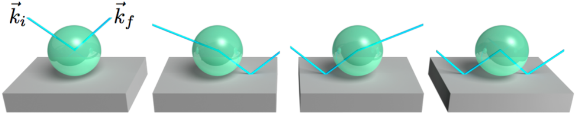
- Involved computations have high degree of parallelism.
- Largely independent computations.
- Perfect for "GPUization".

Forward Simulations: Computing Scattered Light Intensities

Given:
1. a sample structure model, and
2. experimental configuration,

simulate experiments and generate scattering patterns.

Based on *Distorted Wave Born Approximation* (DWBA) theory.

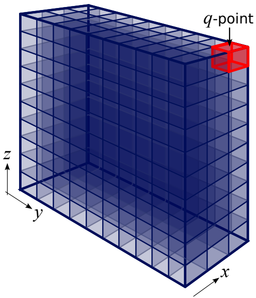


Forward Simulations: Computing Scattered Light Intensities

Q-grid: a 3D region grid in inverse space where scattered light intensities are to be computed.

Intensity: is computed at each q-point $\vec{q}$ in the Q-grid.
At a point $\vec{q}$, it is proportional to square of the sum of *Form Factors* at $\vec{q}$, due to all structures in the sample:

$$I(\vec{q}) \quad \propto \quad \left| \sum_{s=1}^{S} F(\vec{q}) \right|^2$$



Forward Simulations: Computing Form Factors

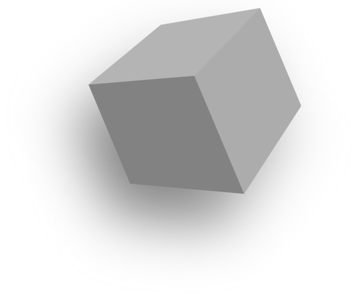
- Form Factor at $\vec{q}$ is an integral over shape surface.

$$F(\vec{q}) = -\frac{i}{|q|^2} \int_{S(\vec{r})} e^{i\vec{\mathbf{q}}_{\mathbf{r}} \cdot \vec{r}} q_n(\vec{r}) d^2\vec{r}$$

- Approximated as summation over a discretized surface:

$$F(\vec{q}) \approx -\frac{i}{|q|^2} \sum_{k=1}^{t} e^{i\vec{q}\cdot\vec{r}_k} q_{n,k} \sigma_k$$

- Complex number computations.
- Analytical Form Factors for simple shapes.



Forward Simulations: Computing Form Factors

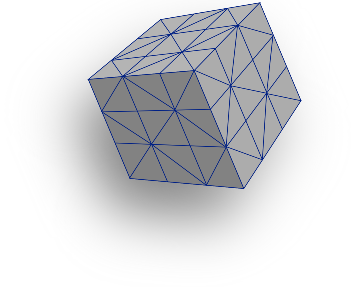
- Form Factor at $\vec{q}$ is an integral over shape surface.

$$F(\vec{q}) = -\frac{i}{|q|^2} \int_{S(\vec{r})} e^{i\vec{\mathbf{q}}_{\mathbf{r}} \cdot \vec{r}} q_n(\vec{r}) d^2\vec{r}$$

- Approximated as summation over a discretized surface:

$$F(\vec{q}) \approx -\frac{i}{|q|^2} \sum_{k=1}^{t} e^{i\vec{q}\cdot\vec{r}_k} q_{n,k} \sigma_k$$

- Complex number computations.
- Analytical Form Factors for simple shapes.

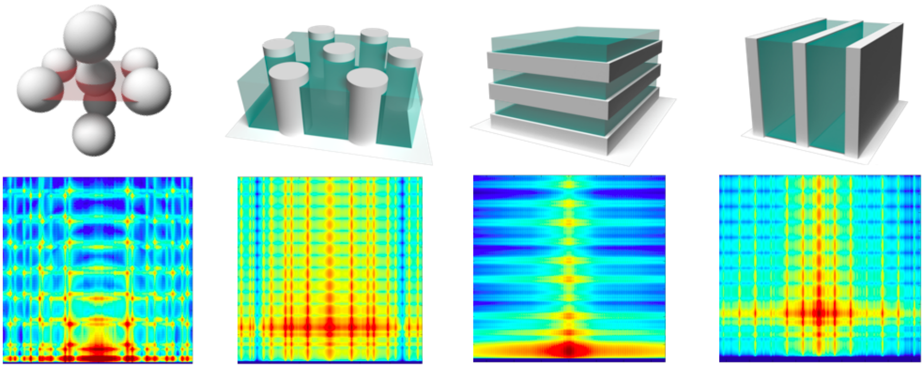


Forward Simulation Problem

Input: 3 arrays, q_x, q_y, q_z of lengths n_x, n_y, n_z, resp., representing a Q-grid of resolution $n = n_x \times n_y \times n_z$, and
Numeric: An array defining the triangulated shape surface as a set of t triangles.

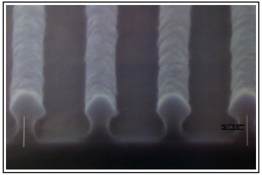
Output: A 3-D matrix M of size $n_x \times n_y \times n_z$, where each entry $M(i, j, k) = F(q_i, q_j, q_k) = F(\vec{q}_{i,j,k})$.

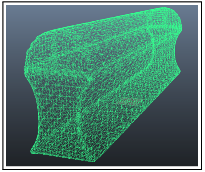
Forward Simulations: Analytic Computation Examples

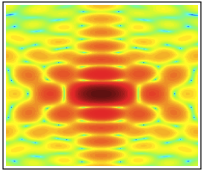


Forward Simulations: Numeric Computation Examples

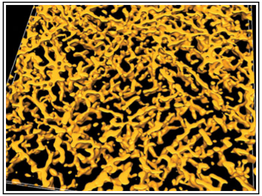
Rectangular Grating with Undercut

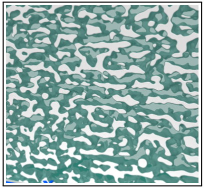


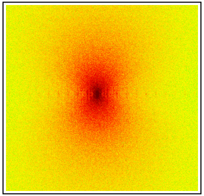




Organic Photovoltaics (OPV)







Real Sample | Model | Scattering Pattern

HipGISAXS: An Open-Source High-Performance GISAXS Data Analysis Software

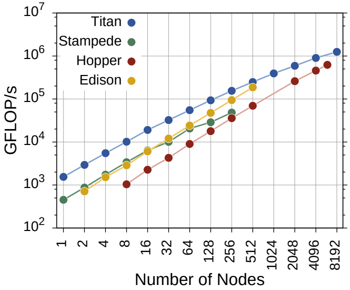
- *http://portal.nersc.gov/project/als/hipgisaxs*.
- Solves many limitations of previous codes.
- Implements new flexible algorithms to handle
  - any complex morphology,
  - multi-layered structures, and
  - all sample rotation directions and beam angles.
- Implements parallelization methods:
  - Deliver high-performance on massively parallel state-of-the-art supercomputers and clusters of multi-core CPUs, Nvidia GPUs, Intel MICs.
  - Bring forward simulation time down to milliseconds and seconds.
- Implements optimization algorithms for inverse modeling.
- Written in C++ with MPI, OpenMP and NVIDIA CUDA.

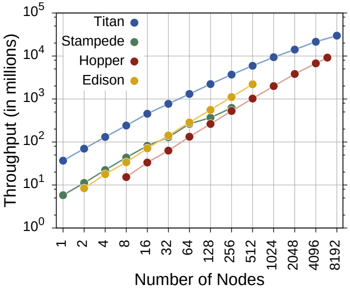
HipGISAXS Forward Simulation Performance

- **GPU Cluster:** *"Titan"*. Up to 8,192 nodes.
    - NVIDIA Tesla K20X Kepler GPUs,
    - 6 GB device memory,
    - 1.15 GHz CUDA core clock,
    - AMD Opteron Interlagos 16 core CPU,
    - 32 GB main memory,
    - Aries interconnects.
- Single precision complex number computations.

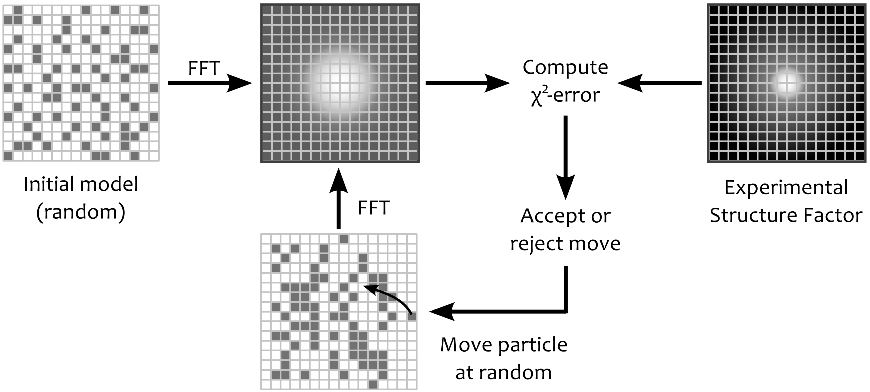
Strong Scaling with Number of Nodes

- GPU cluster (Titan): One MPI process per node. 16 OpenMP threads on host.
- Compared with CPU clusters (Hopper and Edison) and cluster of Intel MICs (Stampede).
- Q-grid size = 8M q-points. Shape definition = 7.6M triangles.





Reconstructing Nano-particle Systems with Reverse Monte Carlo Simulations

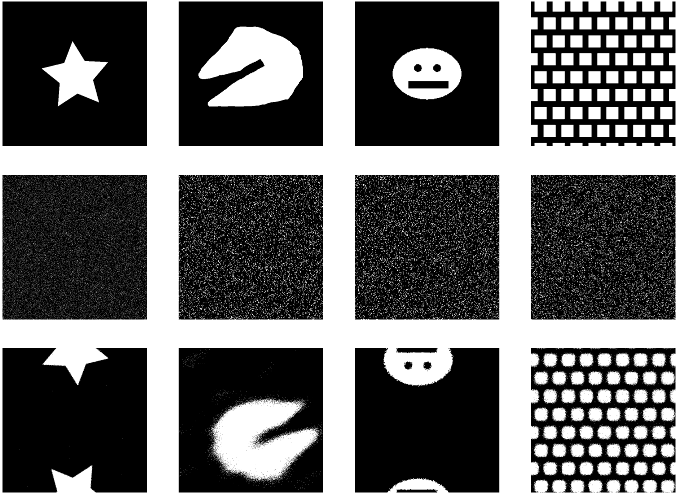


Reconstructing Nano-particle Systems with Reverse Monte Carlo Simulations

Actual Pattern

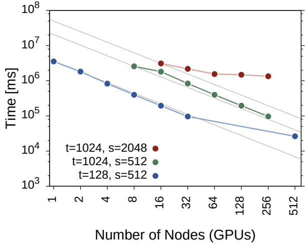
Initial Model

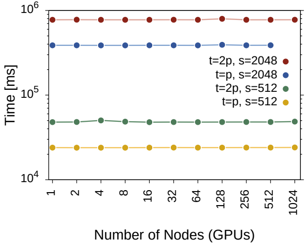
Computed Model



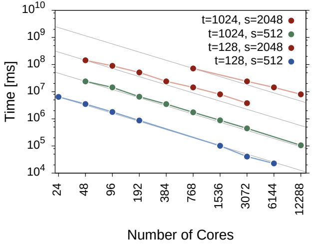
Strong and Weak Scaling of *HipRMC*

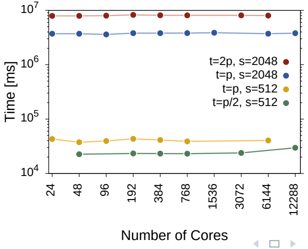
Titan





Hopper





Reconstructing Nano-Structures with Particle Swarm Optimization

- Stochastic method.
- Multiple agents, *"particle swarm"*, perform search for optimal points in the parameter search space.
- Agent velocities influenced by history of traveled paths.

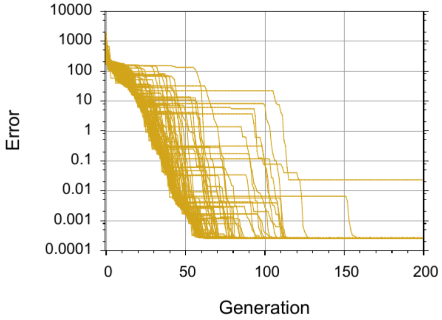
$$v_i = \omega v_i + \phi_b r_b (b_i - x_i) + \phi_g r_g (b_g - x_i)$$

$$p_i = p_i + v_i.$$

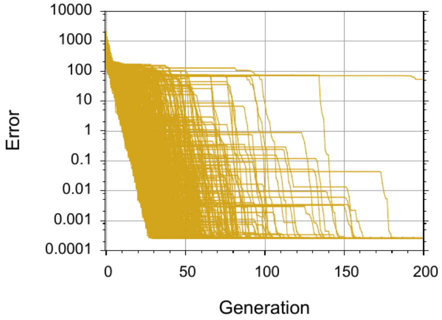
- Inherently suited for coarse parallelism.

Fitting GISAXS Data with PSO

- Fitting a case with just 2 parameters.



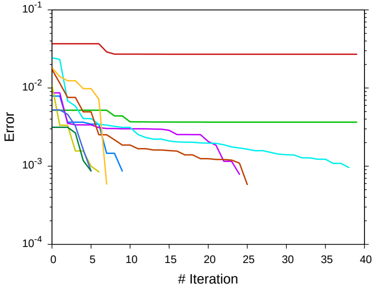
100 Agents

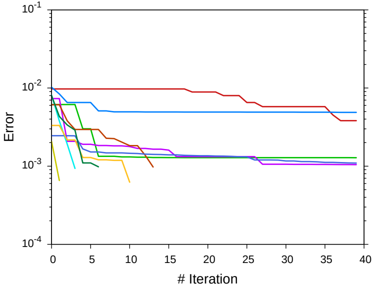


1000 Agents

Fitting GISAXS Data with PSO

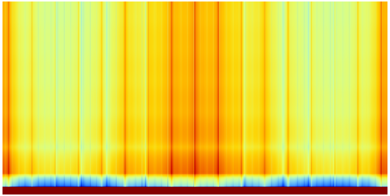
- Fitting a case with 8 parameters.

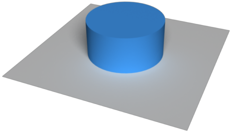


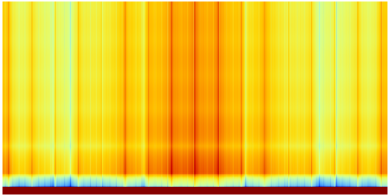


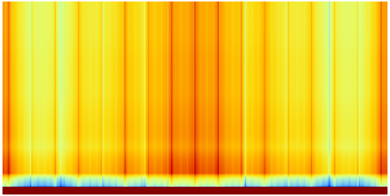
Fitting GISAXS Data with PSO

- Fitting a case with 12 parameters. Normalized error $\approx 10^{-3}$.









End Notes: An Ongoing Work ...

- It has already brought down data analysis time from days and weeks to minutes and seconds.
- HipGISAXS is being used at many light-sources world-wide.
- Incorporate more and better optimization algorithms for inverse modeling:
  - Derivative-based optimization algorithms do not perform well on GISAXS data.
  - Derivative-free methods, such as the trust-region based POUNDers, from the TAO optimization package, are included in HipGISAXS.
- Use machine learning techniques for better feature detection and structural classification.
- Build a pipeline for real-time data processing with dedicated supercomputers using HipGISAXS.
- and much more ...

Work in collaboration with ...

- Xiaoye S. Li, Computational Research Division, Berkeley Lab.
- Slim Chourou, Computational Research Division, Berkeley Lab.
- Alexander Hexemer, Advanced Light Source, Berkeley Lab.

Acknowledgments

- HipGISAXS suite development used GPU cards donated by Nvidia.
- Supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.
- Also supported by DoE Early Career Research grant awarded to Alexander Hexemer.
- Used resources of the National Energy Research Scientific Computing Center, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.
- Used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

Thank you!