OpenAtom Project: Ground and Excited Electronic State Simulations for large systems on massively parallel platforms

PIs: G.J. Martyna (IBM), S. Ismail-Beigi (Yale) and L.V. Kale (UIUC)
Goal: The accurate treatment of complex heterogeneous systems to gain physical insight via novel electronic structure computations.
Supercomputers and novel methods for new Science and Technology

Collaboration between Martyna, Ismail-Beigi and Kale groups to enable novel e-structure capabilities on massively parallel platforms
What is OpenAtom

Sohrab Ismail-Beigi
Applied Physics & Materials
Yale

Sanjay Kale
Computer Science
UIUC

Glenn Martyna
Physical Chemistry & Materials
IBM

NSF SI2-SSI: Scalable, Extensible, and Open Framework for Ground and Excited State Properties of Complex Systems

- OpenAtom software package: DFT MD now, GW next
- Plane waves and pseudopotentials
- charm++ parallel infrastructure
Density Functional Theory (DFT)

Energy functional $E[n]$ of electron density $n(r)$

$$E[n] = KE + E_{ion} + E_H + E_{xc}$$

Minimizing over $n(r)$ gives exact

- Ground-state energy $E_0$
- Ground-state density $n(r)$

Minimum condition

$$\frac{\delta E}{\delta n(r)} = 0$$

equivalent to Kohn-Sham equations

$$\left[-\frac{\nabla^2}{2} + V_{ion}(r) + V_H(r) + V_{xc}(r)\right] \psi_j(r) = \epsilon_j \psi_j(r) \quad V_{xc}(r) = \frac{\delta E_{xc}}{\delta n(r)}$$

- LDA/GGA for $E_{xc}$: good geometries and total energies
- Bad band gaps and excitations

DFT: problems with excitations

Energy gaps (eV)

<table>
<thead>
<tr>
<th>Material</th>
<th>LDA</th>
<th>Expt. [1]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diamond</td>
<td>3.9</td>
<td>5.48</td>
</tr>
<tr>
<td>Si</td>
<td>0.5</td>
<td>1.17</td>
</tr>
<tr>
<td>LiCl</td>
<td>6.0</td>
<td>9.4</td>
</tr>
</tbody>
</table>


Solar spectrum
DFT: problems with energy alignment

Interfacial systems:
- Electrons can transfer across
- Depends on energy level alignment across interface
- DFT has errors in band energies
- Is any of it real?
**One particle Green's function**

\[ G_1(r, r', \omega) = \sum_j \frac{\psi_j(r)\psi_j(r')^*}{\omega - \epsilon_j} \]

**Dyson Equation:**

\[
\left[ -\frac{\hbar^2 \nabla^2}{2m} + V_{\text{ion}}(r) + V_H(r) \right] \psi_j(r) + \int dr' \Sigma_{xc}(r, r', \epsilon_j) \psi_j(r') = \epsilon_j \psi_j(r)
\]

**DFT:**

\[
\left[ -\frac{\nabla^2}{2} + V_{\text{ion}}(r) + V_H(r) + V_{xc}(r) \right] \psi_j(r) = \epsilon_j \psi_j(r)
\]

\[ \Sigma \approx iG_1W, \quad W = \epsilon^{-1}(\omega) * u_c \quad (RPA) \]
Green’s functions successes

Quasiparticle gaps (eV)

<table>
<thead>
<tr>
<th>Material</th>
<th>LDA</th>
<th>GW</th>
<th>Expt.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diamond</td>
<td>3.9</td>
<td>5.6*</td>
<td>5.48</td>
</tr>
<tr>
<td>Si</td>
<td>0.5</td>
<td>1.3*</td>
<td>1.17</td>
</tr>
<tr>
<td>LiCl</td>
<td>6.0</td>
<td>9.1*</td>
<td>9.4</td>
</tr>
<tr>
<td>SrTiO$_3$</td>
<td>2.0</td>
<td>3.4-3.8</td>
<td>3.25</td>
</tr>
</tbody>
</table>


Band structure of Cu

Strokov et al., PRL/PRB (1998/2001)
What is a big system for GW?

P3HT polymer

Band alignment for this potential photovoltaic system?

100s of atoms/unit cell

Not possible *routinely* (with current software)

Zinc oxide nanowire
GW is expensive

Scaling with number of atoms N
- DFT : $N^3$
- GW : $N^4$ (gives better bands)
- BSE : $N^6$ (gives optical excitations)

But in practice the GW is the killer

e.g. a nanoscale system with 50-75 atoms (GaN)

- DFT : 1 cpu x hours
- GW : 91 cpu x hours
- BSE : 2 cpu x hours

∴ Focus on GW
What is so expensive in GW?

One key element: response of electrons to perturbation

\[ P(r, r') = \frac{\partial n(r)}{\partial V(r')} \]

\[ P(r; r') = \text{Response of electron density } n(r) \text{ at position } r \]
\[ \text{to change of potential } V(r') \text{ at position } r' \]
What is so expensive in GW?

One key element: response of electrons to perturbation

\[ P(r, r') = \frac{\partial n(r)}{\partial V(r')} = -2 \sum_v \sum_c \frac{\psi_v(r)\psi_c(r)\psi_v(r')\psi_c(r')}{\varepsilon_v - \varepsilon_c} \]

Standard perturbation theory expression

Problems:
1. Must generate "all" empty states (sum over \( c \))
2. Lots of FFTs to get functions \( \psi_i(r) \) functions
3. Enormous outer produce to form \( P \)
4. Dense \( r \) grid: \( P \) huge in memory
Steps for typical $G_0W_0$ calculation

Stage 1 : Run DFT calc. on structure $\rightarrow$ output : $\epsilon_i$ and $\psi_i(r)$

Stage 2.1 : compute Polarizability matrix $P(r, r') = \frac{\partial n(r)}{\partial V(r')}$

Stage 2.2 : double FFT rows and columns $\rightarrow P(G, G')$

Stage 3 : compute and invert dielectric screening function

$$\epsilon = I - \sqrt{V_{coul}} \ast P \ast \sqrt{V_{coul}} \rightarrow \epsilon^{-1}$$

Stage 4 : “plasmon-pole” method $\rightarrow$ dynamic screening $\rightarrow \epsilon^{-1}(\omega)$

Stage 5 : put together $\epsilon_i$, $\psi_i(r)$ and $\epsilon^{-1}(\omega) \rightarrow$ self-energy $\Sigma(\omega)$
**Steps for typical G$_0$W$_0$ calculation**

Stage 1 : Run DFT calc. on structure $\rightarrow$ output : $\varepsilon_i$ and $\psi_i(r)$

Stage 2.1 : compute Polarizability matrix  
\[
P(r, r') = \frac{\partial n(r)}{\partial V(r')}
\]

Stage 2.2 : double FFT rows and columns $\rightarrow P(G, G')$

Stage 3 : compute and invert dielectric screening function
\[
\varepsilon = I - \sqrt{V_{coul}} \ast P \ast \sqrt{V_{coul}} \rightarrow \varepsilon^{-1}
\]

Stage 4 : “plasmon-pole” method $\rightarrow$ dynamic screening $\rightarrow \varepsilon^{-1}(\omega)$

Stage 5 : put together $\varepsilon_i$, $\psi_i(r)$ and $\varepsilon^{-1}(\omega) \rightarrow$ self-energy $\Sigma(\omega)$
**G versus R space P calculation**

**G-space:**

\[
P(G, G') = -\sum_{v,c} \langle c | e^{-iG \cdot r} | v \rangle \langle v | e^{iG' \cdot r} | c \rangle \frac{2}{\varepsilon_v - \varepsilon_c}
\]

- Directly compute \( P \) in \( G \) space
- Many FFTs: \( N_v N_c \)
- Big multiply: \( N_v N_c N_G^2 = O(N^4) \)

\( N_v \): \# occupied states
\( N_c \): \# unoccupied states
\( N_G \): \# of g vectors

**R-space:**

\[
P(r, r') = -\sum_{v,c} \psi_c^*(r) \psi_v(r) \psi_v^*(r') \psi_c(r') \frac{2}{\varepsilon_v - \varepsilon_c}
\]

Big multiply: \( N_v N_c N_r^2 = O(N^4) \)

- \( N_v N_c \) FFTs needed
- Big \( O(N^4) \) matrix multiply

- \( N_v + N_c + 8N_c \) FFTs needed
- Big \( O(N^4) \) matrix multiply
Eric Mikida
Parallel Implementation

- Completed up to self-energy computation
- Memory is a primary constraint
- Formation of P is the most costly step

Basic Computation

\[ f_{lm} = \Psi_l \times \Psi_m \] for all \( l, m \)

\[ P += f_{lm} f_{lm}^T \] for all \( f \)
GW-BSE Memory Concerns

• 1 MB per state
• 10,000 total states per k-point
• 10 k-points
• 100 GB to store all states
• 1 TB to store P
• 90,000,000 f vectors (90 TB total)
Parallel Decomposition

\[ \Psi \text{ Vectors} \]
\[ \begin{align*}
L \text{ occupied} \\
M \text{ unoccupied}
\end{align*} \]
\[ \text{P Matrix} \]
\[ \begin{align*}
R \\
R
\end{align*} \]

\[ \rightarrow \]
\[ \text{1D Chare Array} \]
\[ \begin{align*}
\text{2D Tiles} \\
\end{align*} \]
\[ \text{2D Chare Array} \]
Parallel Decomposition
Parallel Decomposition

1. Duplicate occupied states on each node
Parallel Decomposition

1. Duplicate occupied states on each node
2. Broadcast an unoccupied state to compute f vectors
Parallel Decomposition

1. Duplicate occupied states on each node
2. Broadcast an unoccupied state to compute f vectors
3. Locally update each matrix tile
Parallel Decomposition

1. Duplicate occupied states on each node
2. Broadcast an unoccupied state to compute f vectors
3. Locally update each matrix tile
4. Repeat step 2 for next unoccupied state
P Formation Scaling

54 atom bulk Si
~0.1MB per state
  108 occupied
  1000 unoccupied
  1 k point

32 processors per node on Vesta (IBM BG/Q @ ANL)

Note: used Berkeley GW v1.1 (8 months old compared to v1.2)
P Formation Scaling

108 atom bulk Si
~0.2 MB per state
  216 occupied
  1832 unoccupied
  1 k point

32 processors per node on Vesta (IBM BG/Q @ ANL)

Note: used Berkeley GW v1.1
(8 months old compared to v1.2)
FFT P to GSpace

1. Convert P to 1D decomposition
2. FFT each row (locally with fftw)
3. Transpose (requires message throttling)
4. FFT each row again
5. Transpose and convert back to 2D
Epsilon Inverse

- Iterative inverse of $\varepsilon$ ($\varepsilon = P$ multiplied and cutoff)
- Utilizes existing OpenAtom matrix multiply library
- Epsilon size is reduced by up to 10x from $P$

### Basic Computation

**Initial:** $X = \varepsilon \ast \varepsilon^T$

**Step 1:** $M1 = 2I - A \ast X$

**Step 2:** $X1 = X \ast M1$

*Converge on ($X = X1$)*

CLA Matrix Algorithm - 2D
Self-Energy Calculation

• Operation on pairs of $f_{nl}$ where $n$ is from an input set of state indices
• Bare Exchange and Screened Exchange

**Basic Computation**

$$f_{nl} A f_{n'l}^T$$ for all $n,l$

- Screened: $A = \varepsilon$
- Bare: $A = \nu(g)$
Parallel Decomposition

- Cache portions of f vectors during P calculation
- Multiply all pairs of $f_{nl}$
- Sum reduction for final result

Very Little Communication
Future Optimizations

- Pipeline unoccupied states in P formation
- Smarter node-level cache storage layout
- Dynamic creation/deletion of matrices
- GPGPUs for BLAS operations
- Overlap phases where possible
Sohrab Ismail-Beigi
54 atom bulk Si
108 occupied
1000 unoccupied
1 k point
32 processors per node

Supercomputer: Vesta (ANL): BlueGene/Q

Note: used Berkeley GW v1.1
(8 months old compared to v1.2)
Parallel performance: P calculation

108 atom bulk Si
216 occupied
1832 unoccupied
1 k point
16 processors per node

Supercomputer: Mira (ANL) : BQ BlueGene/Q

Note: used Berkeley GW v1.1
(8 months old compared to v1.2)
Parallel performance: P calculation

108 atom bulk Si
216 occupied
1832 unoccupied
1 k point
32 processors per node

Supercomputer: Vesta (ANL) : BQ BlueGene/Q

Note: used Berkeley GW v1.1
(8 months old compared to v1.2)
### Where we are with OpenAtom GW

<table>
<thead>
<tr>
<th>Phase</th>
<th>Serial</th>
<th>Parallel</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Compute P in RSpace</td>
<td>Complete</td>
<td>Complete</td>
</tr>
<tr>
<td>2 FFT P to GSpace</td>
<td>Complete</td>
<td>Complete</td>
</tr>
<tr>
<td>3 Invert epsilon</td>
<td>Complete</td>
<td>Complete</td>
</tr>
<tr>
<td>4 Plasmon pole</td>
<td>Complete</td>
<td>In Progress</td>
</tr>
<tr>
<td>5 COHSEX self-energy</td>
<td>Complete</td>
<td>In Progress</td>
</tr>
<tr>
<td>6 Dynamic self-energy</td>
<td>In Progress</td>
<td>Future</td>
</tr>
<tr>
<td>7 Coulomb Truncation</td>
<td>Future</td>
<td>Future</td>
</tr>
</tbody>
</table>

Aim to release COHSEX version early summer 2017
Minjung Kim
Static polarizability calculations

\[ P(r, r') = \sum_{v,c} \psi_c^*(r)\psi_v(r)\psi_v^*(r')\psi_c(r') \frac{2}{\varepsilon_v - \varepsilon_c} \]

\[ N^4 \quad \rightarrow \quad N^3 ? \]

Wilson, Gygi, and Galli, *PRB* 78 (2008)
Liu, Kaltak, Klimes, and Kresse, *PRB* 94 (2016)
Cubic scaling algorithm – 1. Interpolation

\[ P(r, r') = -2 \sum_{v,c} \frac{\psi_v(r)\psi_c^*(r)\psi_c(r')\psi_v^*(r')}{\varepsilon_c - \varepsilon_v} \]

\[ A(r, r'; z) = \sum_c \frac{\psi_c^*(r)\psi_c(r')}{\varepsilon_c - z} \]

1. Save values over some \( z \) grid

\[ P(r, r') = -2 \sum_v \psi_v(r)\psi_v^*(r')A(r, r'; \varepsilon_v) \]

- If \( N_z << N_v \) it scales \( N^3 \)
- \( N_{\text{int}} = 2 \) (linear interpolation) works well

\[ N_r^2 N_c N_v \sim N^4 \]

\[ N_r^2 N_c N_z \]

\[ N_r^2 N_v N_{\text{int}} \]

\( N_z \): number of points to be evaluated
\( N_{\text{int}} \): number of points for interpolation
Cubic scaling algorithm – 2. Laplace method

1. Laplace Identity:
\[
\frac{1}{\varepsilon_c - \varepsilon_v} = \int_0^\infty e^{-(\varepsilon_c-\varepsilon_v) x} dx \approx \sum_k \omega_k e^{-(\varepsilon_c-\varepsilon_v-1)x_k}
\]

2. Gauss-Laguerre quadrature:
\[
\int_0^\infty f(x) e^{-x} dx \approx \sum_k \omega_k f(x_k)
\]

\[
P(r, r') = -2 \sum_c \sum_v \frac{1}{\varepsilon_c - \varepsilon_v} \psi_v(r) \psi_c^*(r) \psi_c(r') \psi_v^*(r')
\]

\[
P(r, r') = -2 \sum_k \omega_k e^{-(\mu_c-\mu_v-1) x_k} \sum_c \psi_c^*(r) \psi_c(r') e^{-(\varepsilon_c-\mu_c) x_k} \sum_v \psi_v(r) \psi_v^*(r') e^{-(\mu_v-\varepsilon_v) x_k}
\]

Number of computation: \(N_r^2 N_{GL} (N_c + N_v) \sim N^3\) \(N_{GL}\) does not depend on system size
Cubic scaling algorithm – 2. Laplace method

3. Windowing:
   - Observation: $N_{GL}$ depends on $\frac{E_{bw}}{E_{gap}} N_r^2 N_{GL} (N_c + N_v)$  
     \[ P(r, r') = \sum_{l} \sum_{m} P_{lm}(r, r') \]
   - Example: 2 by 2 windows  
     \[ P = P_{11} + P_{21} + P_{12} + P_{22} \]

- Windowing can save computational costs
- Useful for materials with small band gap
Estimate the computational costs

- \( N_{GL} \) depends on \( \frac{E_{bw}}{E_{gap}} \)
  
  \( E_{bw} \): band width (\( E_{c,max} - E_{v,min} \))
  
  \( E_{gap} \): band gap

- Cost = \( N_{RS}^{2}N_{GL}(N_c + N_v) \)
  
  1. \( N_{GL} \propto \sqrt{\frac{E_{bw}}{E_g}} \)
  
  2. \( (N_c + N_v)/N \propto (E_{bw} - E_g) \)

Computation cost can be estimated by \( E_{bw} \) and \( E_{g} \)

\[
C \propto \sum_{l} \sum_{m} \sqrt{\frac{E_{bw}}{E_{g}}} \left( \frac{E_{vl}^{max} - E_{vl}^{min}}{E_{v}^{max} - E_{v}^{min}}N_v - \frac{E_{cm}^{max} - E_{cm}^{min}}{E_{c}^{max} - E_{c}^{min}}N_c \right)
\]
Estimate the computational costs

Example: 2x2 window

\[ E_{bw} = 2 \text{ Hartree} \]
\[ E_g = 0.02 \text{ Hartree} \]

\[ E_{v,\text{ratio}} = \frac{E_v^* - E_v,\text{min}}{E_v,\text{max} - E_v^*} \]
\[ E_{c,\text{ratio}} = \frac{E_c^* - E_c,\text{min}}{E_c,\text{max} - E_c^*} \]

Real computational costs

Estimated computational costs
Windowing

How many windows for occupied and unoccupied states?

\[ E_{v_{\text{max}}}-E_{v_{\text{min}}} = 0.44 \text{ Ha} \]
\[ E_{c_{\text{max}}}-E_{c_{\text{min}}} = 1.44 \text{ Ha} \]
\[ E_{bw} = 2 \text{ Ha} \]
\[ E_{g} = 0.02 \text{ Ha} \]

Optimized number of windows:
\[ N_{vw} = 1 \]
\[ N_{cw} = 4 \]
Results

- Si crystal with 16 atoms
- Number of bands: 433
- Number of windows: 1 for \( N_v \) & 4 for \( N_c \)

Maximum error of band gap
- Laplace + windowing: 0.02eV
- Interpolation: 0.23eV

Laplace method with windows wins!!
Results

- Scaling data
- Si crystal with 2, 4, 8, and 16 atoms

<table>
<thead>
<tr>
<th>Number of Atoms</th>
<th>Compute Time per Operation (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>N</td>
</tr>
<tr>
<td>2</td>
<td>N</td>
</tr>
<tr>
<td>4</td>
<td>N</td>
</tr>
<tr>
<td>8</td>
<td>N</td>
</tr>
<tr>
<td>16</td>
<td>N</td>
</tr>
</tbody>
</table>

For the `N^4` method, the operation time is given by:

\[
N^4 : \frac{\text{Operation Time}}{N_v N_c N_r^2}
\]

For the `L + W` method, the operation time is given by:

\[
L + W : \frac{\text{Operation Time}}{\sum_l^{N_{lw}} \sum_m^{N_{wc}} N_{GL} (N_v + N_c) N_r^2}
\]
Glenn Martyna and Qi Li
Projector Augmented Wave based Kohn-Sham Density Functional Theory in OpenAtom with $N^2 \log N$ scaling

OpenAtom team, Qi Li and Glenn Martyna

- **KS-DFT**: *Ground state electronic energy* expressed *exactly* as the *minimum of a functional of the zero temperature, 1-body density* written in terms of

\[
\rho(r, r') = \sum_{i=1}^{N_{KS}} \psi_i(r)\psi_i^*(r'), \quad n(r) = \rho(r, r), \quad N_{KS} = (# \text{ electrons})/2
\]

an orthonormal set of KS states, \(< \psi_i | \psi_j > = 2\delta_{ij}\).

- **KS Density Functional**: *Sum of the kinetic energy* of non-interacting electrons, *Hartree energy, electron-ion/external energy* and an unknown correction term, *exchange correlation energy functional*,

\[
E[n(r)] = -\frac{\hbar^2}{2m_e} \int dr \left( \nabla^2 \rho(r, r') \right)|_{r'=r} + \frac{e^2}{2} \int dr dr' \frac{n(r)n(r')}{|r - r'|}
\]

\[+ e \int dr \ n(r)V_{\text{ext}}(r; N) + E_{xc}[n(r)], \quad N = \# \text{ ions}, N_{KS} \sim N.
\]

- **Generalized Gradient Approximation (GGA)**: *Tractable approx. to* \(E_{xc}\)

\[
E_{xc}[n(r)] \approx \int dr \ e_{xc}(n(r), \nabla n(r))
\]
KS-DFT in OpenAtom

- **OpenAtom**: *Plane-wave* (PW) based KS-DFT within the GGA – expand KS states in the delocalized PW basis.

- **PW-KS-DFT in OpenAtom - Advantages:**
  - $N^2 \log N$ or better scaling of interactions & derivatives - *Euler Exponential Spline (EES) Interpolation*.
  - Only orthogonalization is $\sim N^3$.
  - High parallelism under charm++.
  - $k$-points, path integrals, LSDA & tempering implemented.

- **PW-KS-DFT in OpenAtom - Disadvantages:**
  - *Large basis set* required - millions and millions (c.f. Carl Sagan).
  - *Large memory* required – need large machines.
  - Heavy atoms computationally intensive.
**Projector Augmented Wave Method (PAW)**

- **Projector-Augmented Wave (PAW):** *accurate* treatment of *heavy atoms* in KS-DFT with *low computational cost*.

- **PAW-KS-DFT Advantages**
  - KS states split into localized and delocalized/smooth parts – *small basis* possible even for *heavy atoms*.
  - *NMR* and some other linear response methods require the core – PAW makes it *easy*.
  - *Small memory* requirement.

- **PAW-KS-DFT Disadvantages**
  - Implemented with inefficient $N^3$ methods for interactions.
  - *Parallel performance* of standard implementations *poor*.

**Goal:** Implement $N^2 \log N$ EES-based PAW with high parallel efficiency in OpenAtom.
**PAW Basics: KS states**

- **KS states:** _delocalized/smooth part, (S), + localized/core part, (core)._ Core localized within a sphere of radius $R_{pc}$ around each ion:

$$
\psi_I(r) = \psi_I^{(S)}(r) + \sum_{j=1}^{N} \psi_{ij}^{(\text{core})}(r), \quad \psi_{ij}^{(\text{core})}(r) = 0, |r - R_j| > R_{pc}
$$

- **Smooth:** fills all spaces & varies, _expanded in plane-waves:_

  $$
  \psi_I^{(S)}(s) = \frac{1}{\sqrt{V}} \sum_{g} \bar{\psi}_I^{(S)}(g) \exp(i\hat{g}s)
  $$

  $$
  r = hs, \quad V = \det h, \quad g = 2\pi h^{-1}\hat{g}, \quad \hat{g} \in \text{integer}
  $$

- **Core:** localized, written in _terms of fixed core projectors, \{\Delta p, p^{(S)}\}^*_I :_

  $$
  \psi_{ij}^{(\text{core})}(r) = \Delta p(r - R_j)Z_{ij}^{(S)}, \quad Z_{ij}^{(S)} = < \psi_I^{(S)} | \psi_I^{(S)} > = \int dr^3 p^{(S)}(r - R_j) \psi_I^{(S)}(r), \quad p^{(S)}(r - R_j) = 0, |r - R_j| > R_{pc}
  $$

* 1 ion type, 1 channel for simplicity
PAW Basics: Example KS state

\[ h = \begin{bmatrix} L_x & 0 & 0 \\ 0 & L_y & 0 \\ 0 & 0 & L_z \end{bmatrix} \]

Localized ion core states, \( \psi_{IJ}^{(\text{core})}(\mathbf{r}) \) embedded

in the smooth part of the state, \( \psi_{I}^{(S)}(\mathbf{r}) \), that fills \( D(h) \).
PAW Basics: KS-DFT within LDA under periodic boundary conditions at $\Gamma$

The whole enchilada:

$$E[n(r)] = E_{NIKE} + E_{ext} + E_H + E_{xc}$$

$$E_{NIKE} = -\frac{\hbar^2}{2m_e} \int_{D(h)} dr \sum_I \langle \psi_I | \nabla^2 | \psi_I \rangle$$

$$E_{xc} = \int_{D(h)} dr \varepsilon_{xc}(n(r))$$

$$E_H = \frac{e^2}{2} \int_{D(h)} dr \int_{D(h)} dr' \sum_m \frac{n(r)n(r')}{|r-r'+mh|}$$

$$E_{ext} = -\int_{D(h)} dr \sum_j \sum_m \frac{eQ_j n(r)}{|r-r_j + mh|}$$

Non-interacting electron kinetic energy: Smooth and core terms

$$E_{NIKE} = E_{NIKE}^{(S)} + E_{NIKE}^{(core1)} + E_{NIKE}^{(core2)}$$

$$E_{NIKE}^{(S)} = -\frac{\hbar^2}{2m_e} \int_{D(h)} dr \sum_I \langle \psi_I^{(S)} | \nabla^2 | \psi_I^{(S)} \rangle, \quad E_{NIKE}^{(core1)} = -\frac{\hbar^2}{2m_e} \sum_{ij} Z_{ij}^{(S)} Z_{ij}^{(\nabla^2,\Delta)}, \quad E_{NIKE}^{(core2)} = -\frac{\hbar^2}{2m_e} \sum_j Z_j^{(S,2)} \langle \Delta p | \nabla^2 | \Delta p \rangle$$

Exchange Correlation energy: Smooth and core terms

$$E_{xc} = E_{xc}^{(S)} + E_{xc}^{(core)} = \int_{D(h)} dr \varepsilon_{xc}(n^{(S)}(r)) + \sum_j \int_{D(R_{pc})} dr \left[ \varepsilon_{xc}(n_j(r)) - \varepsilon_{xc}\left(n_j^{(S)}(r)\right) \right]$$

$$n^{(S)}(r) = \sum_I |\psi_I^{(S)}(r)|^2, \quad n_j(r) = n^{(S)}(r - R_j) + n^{(core1)}(r - R_j) + n^{(core2)}(r - R_j), \quad n_j^{(S)}(r) = n^{(S)}(r - R_j)$$

$$\forall r \text{ in } D(h), \quad \forall \ |r-R_j| < R_{pc}$$

$$\forall |r-R_j| < R_{pc}$$
Due to the mixed localized and delocalized basis, there is no natural truncation scale for the long-range interactions of $E_H$ and $E_{ext}$ in $g$-space or $r$-space alone.

$$E_H = \frac{e^2}{2} \int_{D(h)} dr \int_{D(h)} dr' \sum_m \frac{n(r)n(r')}{|r - r' + m\mathbf{h}|}, \quad E_{ext} = -\int_{D(h)} dr n(r) \sum_j \sum_m \frac{eQ_j}{|r - R_j + m\mathbf{h}|}$$

Using Poisson summation and Ewald’s decomposition of $1/r$:

$$E_H = E_H^{(short)} + E_H^{(long)}$$

$$E_H^{(short)} = \frac{e^2}{2} \int_{D(h)} dr \int_{D(h)} dr' \frac{n(r)n(r') \text{erfc}(\alpha |r - r'|)}{|r - r'|}$$

$$E_H^{(long)} = \frac{e^2}{2V} \sum_{g \neq 0} \frac{4\pi}{|g|^2} \exp \left(-\frac{|g|^2}{4\alpha^2}\right) |\bar{n}(g)|^2$$

$$E_{ext} = E_{ext}^{(short)} + E_{ext}^{(long)}$$

$$E_{ext}^{(short)} = -e \int_{D(h)} dr n(r) \sum_j \frac{\text{erfc}(\alpha |r - R_j|)}{|r - R_j|}$$

$$E_{ext}^{(long)} = -\frac{e}{V} \sum_{g \neq 0} \frac{4\pi}{|g|^2} \exp \left(-\frac{|g|^2}{4\alpha^2}\right) \bar{n}(g)\tilde{S}(g)$$

$$+ \frac{\pi e\bar{n}(0)\tilde{S}(0)}{V\alpha^2}$$

$$\tilde{S}(g) = \sum_j Q_j \exp(-ig \cdot R_j)$$

Choose $\alpha$, such that the $g$-space cutoff $= G_c = \text{pw density cutoff}$.

Ensure $r$-space cutoff, $R_c = (3.5 / \alpha) > R_{pc}$, confines the $m$-sum to the 1st image.

Decompose short-range into smooth, core1 and core2 type terms, (not shown).
Accuracy of long/short decomposition

To approximately match long/short range accuracy: \[ \frac{G_c^2}{4} \approx \frac{\gamma^4}{R_c^2}, \quad \gamma = \alpha R_c \]

<table>
<thead>
<tr>
<th>PW cutoff: ( (\hbar^2 G_c^2 / 8\text{me}) ) Ryd</th>
<th>( \gamma = \alpha R_c )</th>
<th>erfc(( \gamma ))</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>3.0</td>
<td>2.21e-05</td>
</tr>
<tr>
<td>9.4</td>
<td>3.5</td>
<td>7.43e-07</td>
</tr>
<tr>
<td>16</td>
<td>4.0</td>
<td>1.54e-08</td>
</tr>
</tbody>
</table>

High accuracy can be obtained with both \( R_c \) and \( G_c \) small!
PAW Basics: Multi-Resolution, Grids, EES and $N^2 \log N$ scaling

How do we reduce scaling by one order in $N$ and maintain accuracy?

1. Discrete real-space: Fourier Coefficients and FFTs

- Given a discrete, $g = 2\pi \hbar^{-1} \hat{g}$, finite g-space, $|g| < G_c$, the Fourier coefficients, $\tilde{f}(g)$ of $f(r)$, can be converted to $\tilde{m}(g)$ from $m(r)$ exactly using an intermediate equally spaced $s$-space grid, $r=hs$, of side $N_{\text{FFT},\alpha} > 2m\hat{g}_{\text{max},\alpha}, \Delta s_\alpha = 1/N_{\text{FFT},\alpha}$.

- Using FFTs, the $\tilde{m}(g)$, can be computed exactly in $N \log N$ as:
  \[
  f(s) = \frac{1}{V} \text{FFT}(m,+)[\tilde{f}(g), G_c], \quad \tilde{m}(g) = \frac{V}{N_{\text{FFT}}} \text{FFT}(m,-)[m(s), mG_c], \quad V = \det h
  \]

2. Euler Exponential Spline Interpolation and FFTs

- To compute the $Z$-matrices, structure factors, $\bar{S}(g)$, and core functions, fast, it is useful develop a differentiable controlled approximation to $\exp(i g \cdot r)$ on a discrete $g$-space for all $r=hs$ in $D(h)$ via interpolation from an equally spaced $s$-space grid, enabling the use of FFTs.

- The Euler exponential spline (EES) delivers where $M_p$ are the cardinal B-splines and $p$ the spline order,
  \[
  e^{2\pi i \hat{g}s} = D_p(\hat{g}, N_{\text{FFT}}) \sum_{\hat{s}=0}^{N_{\text{FFT}}} \sum_{j=1}^{p} M_p(u-\hat{s}) e^{2\pi i \hat{g}\hat{s}} \delta_{\hat{s},l-j} + O\left(\frac{2\hat{g}}{N_{\text{FFT}}}\right)^p, \quad M_p \text{ has compact supp.}
  \]
  \[
  u = s \frac{N_{\text{FFT}}}{\hat{g}_{\text{max}}}, \quad l = \text{int} u, \quad N_{\text{FFT}} > 2\hat{g}_{\text{max}} \approx 2.8\hat{g}_{\text{max}}
  \]

Using 3 FFT grids, (1) Psi EES, (2) Density, (3) Density EES, and 1 discrete spherical polar grid around each ion, $|r| < R_{pc}$, all PAW energy terms & their derivatives can be accurately computed in $N^2 \log N$.
PAW Basics: \( g \)-space to \( s \)-space and back

Psi EES: \( N_{g}^{(\psi)} \sim N \)

\[
\tilde{\psi}_{I}^{(S,D)}(g) = \tilde{\psi}_{I}^{(S)}(g)D^{(\psi)}(g)
\]

\[
\text{FFT}(\psi, \text{EES}) \left[ \tilde{\psi}_{I}^{(S,D)}(g), \frac{G}{2} \right]
\]

Psi: \( N_{\text{FFT}}^{(\psi, \text{EES})} \sim N \)

\[
\psi_{I}^{(S,D)}(s)
\]

\[
S_{\alpha} \equiv \hat{s}_{\alpha}/N_{\text{FFT}, \alpha}^{(\psi, \text{EES})}
\]

\[
0 \leq \hat{s}_{\alpha} < N_{\text{FFT}, \alpha}^{(\psi, \text{EES})}
\]

\[
r = hs
\]

Density: \( N_{g}^{(\psi)} \sim N \)

\[
\tilde{\psi}_{I}^{(S)}(g)
\]

\[
\text{FFT}(\psi, \text{EES}) \left[ \tilde{\psi}_{I}^{(S)}(g), \frac{G}{2} \right]
\]

\[
V \text{ FFT}^{(n,-)} \left[ \tilde{n}^{(S)}(s), G_{c} \right]
\]

\[
N_{\text{FFT}}^{(n)} \sim N \]

Density EES: \( N_{g}^{(n)} \sim N \)

\[
\tilde{n}^{(S,D)}(g) = \tilde{n}^{(S)}(g)D^{(n)}(g)
\]

\[
\text{FFT}(n, \text{EES}) \left[ \tilde{n}^{(S,D)}(g), G_{c} \right]
\]

Density: \( N_{\text{FFT}}^{(n)} \sim N \)

\[
\tilde{n}^{(S)}(g)
\]

\[
\text{FFT}^{(n,+)} \left[ \tilde{n}^{(S,D)}(g), G_{c} \right]
\]

Density: \( N_{\text{FFT}}^{(n)} \sim N \)

\[
n^{(S)}(s)
\]

\[
r = hs
\]

\[
\tilde{n}^{(S,D)}(s)
\]

\[
r = hs
\]

The \( D^{(\tau)}(g) = \prod_{\alpha} D_{p} \left( \hat{g}_{\alpha}, N_{\text{FFT}, \alpha}^{(\tau, \text{EES})} \right) \) enables B-spline interpolation.
PAW Basics: $r$-space interpolation

EES provides an accurate, differentiable interpolation between the different resolutions and length scales of PAW.

**Psi EES:** $N_{\text{FFT}}^{(\psi, \text{EES})} \sim N$

- $h$ defines $D(h) = \text{cuboid}$
- $V = \det h \sim N$
- $\psi_{I}^{(S,D)}(s)$

**Density EES:** $N_{\text{FFT}}^{(n, \text{EES})} \sim N$

- $h$ defines $D(h) = \text{cuboid}$
- $V = \det h \sim N$
- $n^{(S,D)}(s)$

*N-Partition*

FFT grid points, $\{N_B^{(0,n/\psi)}, N_B^{(f,n/\psi)}\}$, $s \in \text{near ion } J$

$N_B^{(\alpha,\beta)} \sim 1$

for EES interpolation.

**All grid spacings are independent of system size.**

*N-Partition*

**2**

**Fine spherical polar grid ($N_f$)**

**N-ion cores in $D(h)$:** $N_f \sim 1$

$2R_{pc} \rightarrow R_{pc} \sim 1$

not to scale
Creating the \( r \)-space representation of the e-density

In the following, the multi-length scale PAW method is used to construct the electron density in \( N^2 \log N \) as a demonstration:

\[
n(r) = n^{(S)}(r) + \sum_j \left[ n_j^{(\text{core } 1)}(r_f) + n_j^{(\text{core } 2)}(r_f) \right], \ n_j^{(S)}(r_f)
\]

(1) Create the smooth KS states in real space, \( \psi_i^{(S)}(s): N^2 \log N \).
(2) Create the smooth density in real space, \( n^{(S)}(s): N^2 \).
(3) *Create the smooth density in the ion cores, \( n_j^{(S)}(r_f): N \log N \).
(4) Create the smooth Z-matrix, \( Z_{ij}^{(S)}: N^2 \log N \).
(5) *Create the core-2 densities, \( n_j^{(\text{core2})}(r_f): N^2 \).
(6) *Create the core-1 densities, \( n_j^{(\text{core1})}(r_f): N^2 \log N \).

* New terms.

Formulae for all other components of PAW-DFT have been derived including ionic and pw expansion coefficient derivatives.
1. Creating the smooth part of the KS states, $\psi_i^{(S)}(s)$, on the density $s$-space FFT grid, $s \in N_{FFT}^{(n)}$

Smooth part of the KS states in $g$-space

$\bar{\psi}_i^{(S)}(g)$, $|g| < \frac{G_c}{2}$, $g = 2\pi h^{-1} \hat{g}$, $\hat{g} \in \text{integer}$

$\frac{1}{\sqrt{V}} \text{FFT}^{(n,+)} \left[ \bar{\psi}_i^{(S)}(g), \frac{G_c}{2} \right]$
2. Creating the smooth density, \( n^{(S)}(s) \) on the density \( s \)-space FFT grid, \( s \in N_{FFT}^{(n)} \)

Smooth part of the KS states on discrete \( s \)-space

Point by point sum of mod squares:

\[
n^{(S)}(s) = \sum_{i} |\psi_{i}^{(S)}(s)|^2 \quad \forall \ s \in N_{FFT}^{(n)}
\]

Smooth part of the density on discrete \( s \)-space

\[
|\psi_{i}^{(S)}(s)|^2 \quad s \in N_{FFT}^{(n)} , \quad r = hs
\]
3. Creating the smooth density, $n_j^{(S)}(r_f)$ around each ion $J$, on the fine grid, $f \in N_f$

EES weighted smooth density in $g$-space

EES weighted smooth density on discrete $s$-space

EES weighted smooth density around each $J$

EES interpolated smooth density around each $J$

$\tilde{n}^{(S,D)}(g) = D^{(n)}(g) \tilde{n}^{(S)}(g)$, $|g| < G_c$

$n^{(S,D)}(s)$, $s \in N^{(n, EES)}_{FFT}$, $r = hs$

$n_j^{(S,D)}(s)$, $N_B^{(f,n)}: s \in \text{near } J$

$n_j^{(S)}(r_f)$, $f \in N_f$

$N = \text{number of ions, } J=1..N$, $N \neq N_{KS}$

$N_f = \text{number points on spherical-polar grid around each ion.}$

$N_f$ and $N_B^{(f,n)}(s \in \text{near } J)$ independent system size.
4. Creating the $Z_{IJ}^{(S)}$ the matrix elements

EES x $\tilde{\psi}^{(S)}$ weighted KS states in $g$-space

FFT ($\psi_+, EES$) $[\tilde{\psi}_I^{(S,D,\tilde{\rho})}(g), g = G_c/2]$

$\tilde{\psi}_I^{(S,D,\tilde{\rho})}(g) = D^{(\psi)}(g)\tilde{\psi}_I^{(S)}(g)\tilde{\rho}^{(S)}(g)$, $g < G_c/2$

$\psi_I^{(S,D,\tilde{\rho})}(s)$, $s \in N_{FFT}^{(\psi,EES)}$

EES x $\tilde{\rho}^{(S)}$ weighted KS states in $s$-space

$N$-Partition, $N$ B-Spline Interps.

Smooth Z-matrix

$Z_{1J}^{(S)}$ $J = 1 \ldots N$

$Z_{2J}^{(S)}$ $J = 1 \ldots N$

$Z_{3J}^{(S)}$ $J = 1 \ldots N$

$Z_{N_{KS}J}^{(S)}$ $J = 1 \ldots N$

$N$–Partition, $N_B^{(0,\psi)}$ B-Spline Interpolations per KS state

$s \in$ near ion 1

$s \in$ near ion 2

$s \in$ near ion 3

$s \in$ near ion $N$

A small set of points, $s \in$ near ion $J$, interpolated to obtain $Z_{1J}^{(S)}$ for all $IJ$ ($N_B^{(0,\psi)} \sim 1$),

$$Z_{1J}^{(S)} = \sum_{\substack{s \in \text{near } J \\text{independent of } I}} \psi_I^{(S,D,\tilde{\rho})}(s)M_{j,p}^{(3)}(s)$$
5. Creating the core density component, \( n_j^{(\text{core2})}(r_f) \), around each ion \( J \), on the fine grid, \( f \in N_f \)

Each KS state contributes to \( N \) unique reductions

\[
Z_J^{(S,2)} = \sum_I |Z_{IJ}^{(S)}|^2
\]

In this example we have 1 projector

\[
n_j^{(\text{core2})}(r_f) = Z_J^{(S,2)} \Delta p^2(r_f) \quad \forall f \in N_f
\]
6. Creating the core density component, $n_j^{(\text{core1})}(r_f)$, around each ion $J$, on the fine grid, $f \in N_f$.

EES weighted KS states in $g$-space

$\psi_i^{(S,D)}(g)$ = $F^T(\psi, +, EES) [\bar{\psi}_i^{(S,D)}(g), G/2]$

N reductions of $s$ around $J=1$

EES weighted KS states in $s$-space

$\bar{\psi}_i^{(S,D)}(g) = D(\psi)(g)\bar{\psi}_i^{(S)}(g), |g| < G_c/2$

$\psi_i^{(S,D)}(s), s \in N_{FFT}^{(\psi, EES)}$

$\psi_j^{(S,D,Z)}(s), N_B^{(f,\psi)}: s \in \text{near } J$

EES weighted KS states around $J$

$Z_{ij}$

N reductions of $s$ around $J=N$

EES interpolated PAW 1 density around each $J$

$B$-Spline interpolation

$N_{KS}$

Each KS state contributes to $N$ unique reductions

$\psi_j^{(S,D,Z)}(s) = \sum_i Z_{ij} \psi_i^{(S,D)}(s) \forall s \in \text{near } J: N_B^{(f,\psi)}$

$Z_{ij}$ = weight for points $s \in \text{near } J$ from KS state, $i$. 
Grand Challenge Application: Perovskite solar cells

- **Pros:**
  - High eff., low cost, tunable band gap (ABX$_3$)

- **Cons:**
  - Instability: water, air, light, interface ... & toxic compounds.

- **Understand:** mechanism of instability/degradation.

- **Search:** non-toxic B$^{2+}$ (Fe, Co, Ni,...) for new high perf. materials.

- **Design:** new interface/encapsulation for novel devices with long lifetime.

- **System size:** 512 atoms (4x4x2 MAPbI$_3$ +128 water), 1264 states
Conclusions

• PAW-KS-DFT is an important method in computational science that allows computations beyond PW-KS-DFT – heavy atoms.

• Using EES Interpolation, we have derived a multi-length scale PAW technique that scales as $N^2 \log N$ (all energy terms and all derivatives) – an important advance.

• We are currently developing the charm++ implementation to allow very large systems to be studied efficiently.
Supplementary: More PAW method pictures
Creating the \( g \)-space representation of the e-density

\[
\bar{n}(g) = \bar{n}^{(S)}(g) + \bar{n}^{(\text{core }1)}(g) + \bar{n}^{(\text{core }2)}(g)
\]

\( \bar{n}^{(S)}(g) \): Sampling theorem from \( n^{(S)}(s) \)
\( \bar{n}^{(\text{core }1)}(g) \): Numerical integration over core 1 density + EES
\( \bar{n}^{(\text{core }2)}(g) \): Bessel transform (precompute) + EES

(1) Create the smooth density in g-space,
(2) *Create core-1 density in g-space,
(3) *Create core-2 density in g-space,

*new terms

\( \bar{n}^{(S)}(g) \): \( N \log N \)
\( \bar{n}^{(\text{core }1)}(g) \): \( N \log N \)
\( \bar{n}^{(\text{core }2)}(g) \): \( N \log N \)
1. Creating the $g$-space representation of smooth density

Density: $N_{FFT}^{(n)} \sim N$

$n^{(S)}(s)$

$s_\alpha \equiv \hat{s}_\alpha / N_{FFT,\alpha}^{(n)}$

$0 \leq \hat{s}_\alpha < N_{FFT,\alpha}^{(n)}$

$\hat{s}_\alpha$ integer

$r = hs$

$V \frac{FFT^{(n,-)}}{N_{FFT}^{(n)}} \left[ \tilde{n}^{(S)}(s), G_c \right]$

Density: $N_g^{(n)} \sim N$

$\tilde{n}^{(S)}(g)$

Density Fourier coefficients $\tilde{n}^{(S)}(g), |g| < G_c$ are exact, through intermediate discrete $s$-space – Theorem 1.

$N \log N$ method given $n^{(S)}(s)$. 

$\hat{s}_\alpha$
2. Creating the $g$-space core 1 density, $\bar{n}^{\text{core,1}}(g)$

- EES interpolated core 1 density around each ion J
- EES weighted core 1 density around each J
- EES weighted core 1 density consolidated
- EES interpolated core 1 density in $g$-space

\[ n^{\text{core,1,EES,2}}(s), \quad s \in N^{(n,EES)}_{\text{FFT}} \]

\[ n^{\text{core,1,EES,2}}(s) = \sum_j \sum_{s' \in \text{near } j} n_j^{\text{core,1,EES,2}}(s') \delta_{s,s'} \]

\[ \bar{n}^{\text{core,1,EES}}(g) = \sum_j \sum_f w_f n_j^{\text{core,1,EES}}(r_f) \exp \left( ig \cdot (r_f + R_j) \right) \]

\[ f \in N_f, \quad N^{(f,n)}_B : s \in \text{near } J \]

$g$-space representation of core 1 density

Treated via EES above
3. Creating the $g$-space core 2 density, $\overline{n}^{(\text{core 2,EES})}(g)$

$$Z_{j}^{(S,2)} \text{ vector}$$

Structure Factor $S_{j}^{(Z,2,EES)}(s)$ around each J

EES interpolated Structure Factor $S^{(Z,2,EES)}(s)$

EES interpolated $g$-space core 2 density

$Z_{1}^{(S,2)}$

B-Spline

$Z_{2}^{(S,2)}$

$Z_{3}^{(S,2)}$

$\vdots$

$Z_{N}^{(S,2)}$

$S_{j}^{(Z,2,EES)}(s)$,

$N_{B}^{(0,n)}: s \in \text{near J}$

$D^{(n)}(g)\overline{p}^{2}(g) \times \text{FFT}_{(n,-,EES)}[S^{(Z,2,EES)}(s), G_{c}]$

$\overline{n}^{(\text{core 2,EES})}(g), \quad |g| < G_{c}$

$\overline{S}^{(Z,2)}(g) = \sum_{j} Z_{j}^{(S,2)} \exp(ig \cdot R_{j})$

Treated via EES to achieve $N \log N$

core 2 density in $g$-space by EES

$$\overline{n}^{(\text{core 2,EES})}(g) = \Delta p^{2}(g)\overline{S}^{(Z,2,EES)}(g) \quad \forall \ |g| < G_{c}$$
Creating the Energy

(1) Kinetic Energy of non-interacting electrons
   i. Smooth
   ii. Core 1*
   iii. Core 2*

(2) Local e-ion energy
   i. Smooth
   ii. Core 1 (short and long)*
   iii. Core 2 (short and long)*

(3) Exchange-Correlation
   i. Smooth
   ii. Core*

(4) Hartree
   i. Smooth-Smooth : long + short range
   ii. Long range*
   iii. Smooth-Core 1/2 : short range*
   iv. Core 1/Core 2 : short range*

*new terms
1. Kinetic Energy of non-interacting electrons

$$E_{NIKE} = E_{NIKE}^{(S)} + E_{NIKE}^{(core1)} + E_{NIKE}^{(core2)}$$

$$E_{NIKE}^{(S)} = -\frac{\hbar^2}{2m_e} \int_{D(h)} d\mathbf{r} \sum_l \left| \psi_I^{(S)} \right| ^2 \right| \nabla^2 \psi_I^{(S)} \right| = \frac{\hbar^2}{2m_e} \sum_l \sum_g \sum_{|g|<G_c/2} g^2 \left| \bar{\psi}_I^{(S)}(g) \right| ^2$$

*$$E_{NIKE}^{(core1)} = -\frac{\hbar^2}{2m_e} \sum_{IJ} Z_{IJ}^{(S)} Z_{IJ}^{(\nabla^2 S, \Delta)}$$

*$$E_{NIKE}^{(core2)} = -\frac{\hbar^2}{2m_e} \sum_J Z_J^{(S,2)} \left\langle \Delta p \left| \nabla^2 \right| \Delta p \right\rangle = \Delta p^{(KE)} \sum_J Z_J^{(S,2)}$$

$$\Delta p^{(KE)} = \text{constant} = -\frac{\hbar^2}{2m_e} \int_{D(R_pc)} d\mathbf{r} \Delta p(\mathbf{r}) \nabla^2 \Delta p(\mathbf{r})$$

*new terms

Note, the EES computation of $Z_J^{(S,2)}$ and $Z_{IJ}^{(S)}$ has already been presented and computing

$$Z_{IJ}^{(\nabla^2 S, \Delta)} = -\frac{\hbar^2}{2m_e} \int_{D(h)} d\mathbf{r} \Delta p(\mathbf{r} - R_J) \nabla^2 \psi_I^{(S)}(\mathbf{r})$$

by EES just requires utilizing a slightly different input in slide 17,

$$\bar{\psi}_I^{(\nabla^2 S, D, \Delta p)}(g) = D^{(\psi)}(g) g^2 \bar{\psi}_I^{(S)}(g) \bar{\Delta p}(g)$$
## Accuracy of long/short decomposition

To approximately match long/short range accuracy:

\[
\frac{G_c^2}{4} \approx \frac{\gamma^4}{R_c^2}, \quad \gamma = \alpha R_c
\]

<table>
<thead>
<tr>
<th>$R_c = 4$ bohr</th>
<th>$\gamma = \alpha R_c$</th>
<th>erfc((\gamma))</th>
</tr>
</thead>
<tbody>
<tr>
<td>PW cutoff: ((\hbar^2 G_c^2 / 8me)) Ryd</td>
<td>5.1</td>
<td>3.0</td>
</tr>
<tr>
<td></td>
<td>9.4</td>
<td>3.5</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>4.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$R_c = 2$ bohr</th>
<th>$\gamma = \alpha R_c$</th>
<th>erfc((\gamma))</th>
</tr>
</thead>
<tbody>
<tr>
<td>PW cutoff: ((\hbar^2 G_c^2 / 8me)) Ryd</td>
<td>20.3</td>
<td>3.0</td>
</tr>
<tr>
<td></td>
<td>37.5</td>
<td>3.5</td>
</tr>
<tr>
<td></td>
<td>64.0</td>
<td>4.0</td>
</tr>
</tbody>
</table>

High accuracy can be obtained with both $R_c$ and $G_c$ small!
OpenAtom Ground State Software Overview

PPL Contributors: Eric Bohm, Nikhil Jain, Prateek Jindal, Eric Mikida, Michael Robson
Software Infrastructure

- GIT (Gerrit) based repository:
  - http://charm.cs.illinois.edu/gerrit/openatom
  - Or https://github.com/ericbohm/OpenAtom/
- Test system datasets available in git
  - Make test - Basic feature verification
  - Make full_test - Extensive use case verification
- Jenkins testing
  - Release branch in nightly Charm++ testing
  - Release branch in Charm++ continuous integration testing
## Ground State Feature Status

<table>
<thead>
<tr>
<th>Feature</th>
<th>Minimization Status</th>
<th>Dynamics Status</th>
<th>Test Integration</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPAIMD Dynamics</td>
<td>NA</td>
<td>Production</td>
<td>Automated</td>
</tr>
<tr>
<td>Path Integrals</td>
<td>Production</td>
<td>Production</td>
<td>Automated</td>
</tr>
<tr>
<td>K-Points</td>
<td>Production</td>
<td>Needs Verification</td>
<td>Automated</td>
</tr>
<tr>
<td>Spin Orbitals</td>
<td>Production</td>
<td>Production</td>
<td>Automated</td>
</tr>
<tr>
<td>Tempering</td>
<td>NA</td>
<td>Production</td>
<td>Automated</td>
</tr>
<tr>
<td>Born Oppenheimer Dynamics</td>
<td>NA</td>
<td>Production</td>
<td>Automated</td>
</tr>
<tr>
<td>Band Generation</td>
<td>Being Evaluated</td>
<td>Being Evaluated</td>
<td>Manual</td>
</tr>
</tbody>
</table>
Control flow in OpenAtom (PW-DFT)
QBOX COMPARISON
Performance comparison for Water-32M-10Ry

- **Qbox**
- **OpenAtom**

**Axes:**
- **Y-axis:** Time per step (in ms)
- **X-axis:** Number of cores (nodes * 16)
Water - 512 M - 70 Ry on Blue Gene/Q

Time per step (s)

Number of nodes (up to 16 cores and 64 threads per node)

Qbox
OpenAtom
Topology aware mapping

Two dimensions based on states
Matrix-matrix multipliers

Two dimensions based on planes

Density
$n_r$ planes

$n_s$ states

RSpace

$GSpace$

$n_g$ planes

$n_s$ states
Adapting to different systems

• Separate the logical operations and machine-specific operations. Example:
  – Logical operation: get an ordered list of nodes
  – Machine specific: Hilbert curve traversal, blocked traversal, plane-traversal

• Density FFTs: require use of full bisection bandwidth – spread throughout the allocation.

• Matrix-matrix multiplies (pair calculators): place near the GSpace planes, but load balance is important.
System utilization without mapping
(barriers introduced for clarity)

States: Many G -> R FFTs
Density: G -> R -> G
Non-local G -> R -> G
Density to States States R -> G
Force correction Ortho-normalization

[Diagram showing time series with various data points and analysis types]
System utilization with mapping
Impact of mapping on Blue Gene/Q: up to 30% improvement

Scaling of MOF on Blue Gene/Q

Time per step (s)

Number of cores (3 hardware threads per core)
Impact of mapping on Blue Waters: up to 32% improvement

Scaling of MOF on Cray XE6

Default Mapping

Topology-aware mapping

Time per step (s)

Number of cores (1 hardware thread per core)
UBERS : MULTI-INSTANCE METHODS
Multi Instance Methods

- Retain all existing code with minimal changes
- Any feature available for CP minimization or dynamics automatically available for multi-instance use
- Add Master Index of objects
  - Uber[temper][bead][k-point][spin]
  - Objects in any instance can be referenced by any object
- Support simulations with many kinds of multi instance physics
- Instance Controller
- Temper Controller
  - Sum energies across Tempers and Beads
  - Switch Energies and Temperatures
- Bead Controller
  - Intrapolymer force evaluation and integration
Spin Orbitals (LSDA)

- Each Spin shares atom and energy shares
- Electron density from down passed to up
  - VKS computed for each spin
  - Returns to standard flow of control
- Independent I/O for state data
- Independent placement for instance shares
K-Points

- Each k-point shares:
  - electron density, atoms, energy charges
- Electron density = sum over KP electron states
- Wave functions outside the first Brillouin zone forces use of complex (e.g., ZGEMM)
  - Instead of the “doublepack” optimization used at the Γ point
- Independent I/O for state data
- Independent placement for electron state instance charges
Path Integral Beads

- Path Integral Bead replica contains independent instances of all phases of CPAIMD
  - May contain k-point and spin ensembles
- Intrapolymer force evaluation in PIBeadAtoms
  - Interacts with each Bead instance's AtomsCompute
  - Supplements CPAIMD nucleic force integration phase
  - Computation Parallelized across NumAtoms and NumBeads
- Independent I/O for state and coordinate data
- Independent placement for instance shares
Tempers

- Contains independent instances of all phases of CPAIMD
- Each temper may contain Beads, K-points, and Spin instances
- Temper controller manages random neighbor shuffle to exchange temperatures across temper replicas
- Independent I/O for state and coordinate data
- Independent placement for instance charis
Temper Performance

Scaling Tempers on Blue Gene/Q

Time per step (s)

4K  8K  16K  32K  65K  131K

Number of cores (3 hardware threads per core)

4 tempers  8 tempers  16 tempers  32 tempers  64 tempers
Combined Performance

Scaling Beads (B) and Tempers (T) on Blue Gene/Q

- B1-T1
- B2-T1
- B2-T2
- B4-T2
- B4-T4

Time per step (s)

Number of cores (3 hardware threads per core)
Please Refer to: Heterogeneous Computing in Charm++
OpenAtom CPU Performance
Water 256M_70Ry 64 Nodes XK
OpenAtom GPU Performance
Water 256M_70Ry 64 Nodes XK
PAW PARALLELIZATION

Eric Bohm, Qi Li, Glenn Martyna
Parallelization of PAW

- PAW method variation by Li and Martyna
  - Smooth component uses existing DFT code
  - Core components
    - Implemented via EES FFT
      - Reuse prior work

- New challenges:
  - effective overlap between smooth and core
  - Communication and Memory management
    - num_coretype * num_channel * num_projector
    - core_1 core_2
Parallelization Design

- Control flow dependencies introduced by PAW
  - PAW elements
    - f_grid
    - Bsplines
    - core zmatrices
  - Interactions with existing data structures
    - $\rho^s$
      - PW $\rho$ is now the smooth part of $\rho$
    - $\psi^s$
      - PW $\psi$ is now the smooth part of $\psi$
    - Zmatrix
      - With PAW projectors, but otherwise same operations of smooth Zmatrix
PAW Design II

• F_grids are relatively small
  • <500 grid points
    – Multicast and reduce to produce results dependent on f_grid
• Z-matrices comparatively large
  – Decomposed same as in particle plane
• Computation of each core_1 and core_2 are mutually independent, also independent by channel
  – Can be overlapped
  – Expected to require scheduling to constrain memory and bandwidth consumption
• **Key take away**: PAW will greatly expand the portion of the time step spent in non-local and density.
Ground State Future Work

• PAW
• Section/Partition optimizations for Uber Instances
• Band generation (automated testing)
• Improved heuristics for default decomposition parameter choices
• Fast Hartree-Fock
• Charm-FFT
  – Integrate use in electron state and non-local
  – Offload to GPGPU and Xeon-Phi
Thank you!

- NSF: SI\textsubscript{2}-SSI: Collaborative Research: Scalable, Extensible, and Open Framework for Ground and Excited State Properties of Complex Systems
- NCSA: BlueWaters
- ANL: Mira
- LLNL: Vulcan
Conclusions

Thanks for listening!

… to the update on the OpenAtom GW work

Questions?
Reducing the scaling: quartic to cubic

\[ P(G, G') = \sum_{v,c} \langle c | e^{-iG \cdot r} | v \rangle \langle v | e^{iG' \cdot r} | c \rangle \frac{2}{\varepsilon_v - \varepsilon_c} \]

\[ P(r, r') = \sum_{v,c} \psi_c^*(r) \psi_v(r) \psi_v^*(r') \psi_c(r') \frac{2}{\varepsilon_v - \varepsilon_c} \]

- Both are \( O(N^4) \)
- Sum-over-state (i.e., sum over unoccupied “c” band) not to blame: removal of unocc. states still \( O(N^4) \) but lower prefactor*

- Working in R-space can reduce to \( O(N^3) \) [see also †]

*Umari, Stenuit, Baroni, PRB 81, (2010)
*Giustino, Cohen, Louie, PRB 81, (2010)

† Liu, Kaltak, Klimes, and Kresse, PRB 94, (2016)
**Steps for typical $G_0W_0$ calculation**

Stage 1: Run DFT calc. on structure $\rightarrow$ output: $\varepsilon_i$ and $\psi_i(r)$

Stage 2.1: compute Polarizability matrix

$$P(r, r') = \frac{\partial n(r)}{\partial V(r')}$$

Stage 2.2: double FFT rows and columns $\rightarrow P(G, G')$

Stage 3: compute and invert dielectric screening function

$$\varepsilon = I - \sqrt{V_{coul}} \ast P \ast \sqrt{V_{coul}} \rightarrow \varepsilon^{-1}$$

Stage 4: “plasmon-pole” method $\rightarrow$ dynamic screening $\rightarrow \varepsilon^{-1}(\omega)$

Stage 5: put together $\varepsilon_i$, $\psi_i(r)$ and $\varepsilon^{-1}(\omega) \rightarrow$ self-energy $\Sigma(\omega)$
Iterative matrix inversion for Hermitian matrix A:


\[ X_0 = \alpha A^\dagger \quad \alpha \in \left(0, \frac{2}{R}\right) \quad R = \max_i \sum_j (AA^\dagger)_{i,j} \]

\[ X_{n+1} = X_n (2I - AX_n) \]

We just “borrow” the pre-existing OpenAtom+charmm fast parallel matrix multiplication
**Steps for typical \( G_0 W_0 \) calculation**

Stage 1: Run DFT calc. on structure \( \rightarrow \) output: \( \varepsilon_i \) and \( \psi_i(r) \)

Stage 2.1: compute Polarizability matrix

\[
P(r, r') = \frac{\partial n(r)}{\partial V(r')}
\]

Stage 2.2: double FFT rows and columns \( \rightarrow P(G, G') \)

Stage 3: compute and invert dielectric screening function

\[
\epsilon = I - \sqrt{V_{coul}} \ast P \ast \sqrt{V_{coul}} \rightarrow \epsilon^{-1}
\]

Stage 4: “plasmon-pole” method \( \rightarrow \) dynamic screening \( \rightarrow \epsilon^{-1}(\omega) \)

Stage 5: put together \( \varepsilon_i \), \( \psi_i(r) \) and \( \epsilon^{-1}(\omega) \) \( \rightarrow \) self-energy \( \Sigma(\omega) \)
GW-Static Self-Energy (COHSEX)

For v1 of software: make a simplifying “static” self-energy approximation

- An approximation to “real” GW
- Easier to code and test correctness
- Good quality results with tweaking of approximation

<table>
<thead>
<tr>
<th>System</th>
<th>Experiment</th>
<th>GW (full)</th>
<th>COHSEX</th>
<th>Corrected COHSEX*</th>
<th>DFT-LDA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diamond</td>
<td>5.48</td>
<td>5.70</td>
<td>6.99</td>
<td>5.93</td>
<td>4.2</td>
</tr>
<tr>
<td>Si</td>
<td>1.17</td>
<td>1.29</td>
<td>1.70</td>
<td>1.18</td>
<td>0.49</td>
</tr>
</tbody>
</table>

\[
\begin{align*}
\Sigma(r, r') &= \Sigma^X(r, r') + \Sigma^{SEX}(r, r') + \Sigma^{COH}(r, r') \\
\Sigma^X(r, r') &= -\sum_v \psi_v(r)\psi_v(r')^* \frac{1}{|r - r'|} \\
\Sigma^{SEX}(r, r') &= -\sum_v \psi_v(r)\psi_v(r')^*[W(r, r') - 1/|r - r'|] \\
\Sigma^{COH}(r, r') &= \frac{1}{2}\delta(r - r')[W(r, r') - 1/|r - r'|]
\end{align*}
\]

*Kang & Hybertsen, Phys. Re.v B, 82, (2010)*
Interestingly, direct real space method is not best here. Wave vector (Fourier) space is better computationally. Serial version written and correctness tested.

\[
f^{nl}(G') = \int dr e^{-iG \cdot r} \psi_n(r)^* \psi_l(r) = \text{FFT}[\psi_n(r)^* \psi_l(r)]
\]

\[
S_{G,G'} = \sqrt{V(G)} \times [\epsilon^{-1} - I]_{G,G'} \times \sqrt{V(G')}
\]

\[
\langle n | \Sigma^X | n' \rangle = - \sum_{l,G} f^{nl}(G') \times V(G) \times f^{n'l}(G')^*
\]

\[
\langle n | \Sigma^{SEX} | n' \rangle = - \sum_l \sum_G f^{nl}(G') \times \sum_{G'} S_{G,G'} \times f^{n'l}(G')^*
\]

\[
\langle n | \Sigma^{COH} | n' \rangle = \frac{1}{2} \sum_{G,G'} S_{G,G'} \times f^{nn'}(G - G')
\]
GW: some math details

1. Calculate RPA polarizability $P$

$$P_{q,q'}(\omega) = \sum_{c,v} \frac{2(\varepsilon_c - \varepsilon_v)}{\omega^2 - (\varepsilon_c - \varepsilon_v)^2} \cdot \langle v | e^{i q \cdot \hat{r}} | c \rangle \langle v | e^{i q' \cdot \hat{r}} | c \rangle^*$$

2. Calculate screened interaction $W$

$$\epsilon(\omega) = I - VP(\omega) \quad W(\omega) = \epsilon(\omega)^{-1} V$$

$$W_{q,q'}(\omega) = V_{q,q'} + \sum_p \frac{2\omega_p}{\omega^2 - \omega_p^2} B_{q,q'}^p$$

3. Calculate self–energy correction $\Sigma$ for each state $n$

$$\langle n | \Sigma(\varepsilon_n) | n \rangle = \sum_{q,q'} \sum_l \left( \sum_p \frac{B_{q,q'}^p}{\varepsilon_n - \varepsilon_l - \omega_p} \right) \cdot \langle n | e^{i q \cdot \hat{r}} | l \rangle \langle n | e^{i q' \cdot \hat{r}} | l \rangle^*$$

$$- \sum_{q,q'} \sum_{\nu} W_{q,q'}(\varepsilon_n - \varepsilon_{\nu}) \cdot \langle n | e^{i q \cdot \hat{r}} | \nu \rangle \langle n | e^{i q' \cdot \hat{r}} | \nu \rangle^*$$
GW: matrix elements

How do matrix elements $\langle l | \exp(iq \cdot \hat{r}) | n \rangle$ converge with $l$?

Simple sum rule

$$1 = \sum_{l=1}^{\infty} |\langle l | \exp(iq \cdot \hat{r}) | n \rangle|^2 = \sum_{l=1}^{\infty} \langle n | \exp(-iq \cdot \hat{r}) | l \rangle \langle l | \exp(iq \cdot \hat{r}) | n \rangle = \langle n | n \rangle$$

$q$ with $\hbar^2 q^2 / 2m = 6 \text{ Ryd}$

$\therefore$ Need $\varepsilon_l \approx E_q$ to converge

Why?

High energy $|l\rangle$ are $\approx$ free–e$^-$ with $\varepsilon_l \approx \hbar^2 q_l^2 / 2m$

So must sample $|q_l| \sim |q|$ to catch dominant parts of $|vbm\rangle$
GW: details 1

\[ \Sigma(r, r', t) = iG_1(r, r', t) W(r, r', t) \]

\[ \Sigma(r, r', \omega) = i \int_{-\infty}^{\infty} \frac{d\omega'}{2\pi} G_1(r, r', \omega - \omega') W(r, r', \omega') \]

Screened interaction \( W \) given by convolution

\[ W(\omega) = \varepsilon^{-1}(\omega) * v_c \]

Dielectric function \( \varepsilon(\omega) \) given by polarizability \( P \)

\[ \varepsilon(\omega) = I - v_c * P(\omega) \]

RPA polarizability \( P(r, r', \omega) = \frac{\delta n(r, \omega)}{\delta V_{tot}(r', \omega)} \) given by

\[ P(r, r', \omega) = \sum_{c,v} \psi_c(r) \psi_v^*(r) \psi_c^*(r') \psi_v(r') \times \left[ \frac{1}{\omega - (\varepsilon_c - \varepsilon_v)} - \frac{1}{\omega + (\varepsilon_c - \varepsilon_v)} \right] \]
GW: details 2

Solving Dyson’s equation: write as perturbation on DFT

\[
\begin{align*}
[T + V_{ion} + V_H + V_{xc} + (\Sigma - V_{xc})] \psi_j &= \epsilon_j \psi_j \\
[H^{DFT} + (\Sigma - V_{xc})] \psi_j &= \epsilon_j \psi_j
\end{align*}
\]

Take matrix elements among DFT states \( \rightarrow \) diagonalize

\[
H_{jk} = \epsilon_j^{DFT} \delta_{jk} + \langle \psi_j^{DFT} | \Sigma(\epsilon_j) - V_{xc} | \psi_k^{DFT} \rangle
\]

Common approximations:

- Take \( |\psi_j> \approx |\psi_j^{DFT}> \) so system already diagonal

\[
\epsilon_j^{DFT} + \langle \psi_j^{DFT} | \Sigma(\epsilon_j) - V_{xc} | \psi_j^{DFT} \rangle = \epsilon_j
\]

- Evaluate \( \Sigma \) and \( d\Sigma/d\epsilon \) at \( \epsilon_j^{DFT} \) and solve

\[
\epsilon_j = \epsilon_j^{DFT} + \frac{\langle \psi_j^{DFT} | \Sigma(\epsilon_j^{DFT}) - V_{xc} | \psi_j^{DFT} \rangle}{1 - \langle \psi_j^{DFT} | d\Sigma/d\epsilon | \psi_j^{DFT} \rangle |_{\epsilon_j^{DFT}}}
\]
Density Functional Theory

For a interacting electronic system, can get

- exact ground-state energy $E_0$
- exact ground-state electron density $n(r)$

by solving self-consistent effective single-particle problem

$$\left[-\frac{\nabla^2}{2} + V_{\text{ion}}(r) + \phi(r) + V_{xc}(r)\right] \psi_j(r) = \epsilon_j \psi_j(r)$$

$$\phi(r) = \int dr' \frac{n(r')}{|r - r'|}, \quad V_{xc}(r) = \frac{\delta E_{xc}}{\delta n(r)}, \quad n(r) = \sum_j |\psi_j(r)|^2$$

Typical: Local Density Approximation (LDA)

$$E_{xc}[n(r)] \approx E_{xc}^{LDA}[n(r)] = \int dr \ n(r) \ e_{xc}(n(r))$$