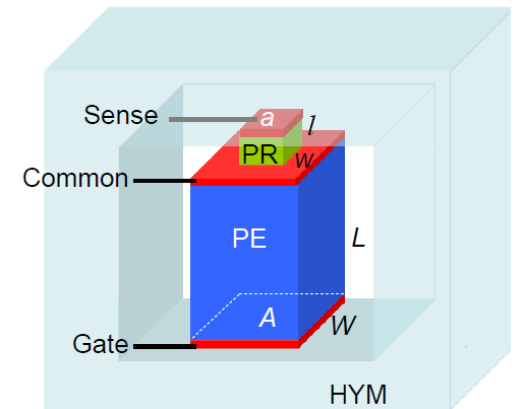
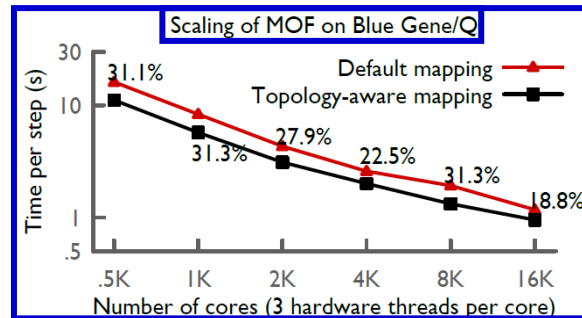
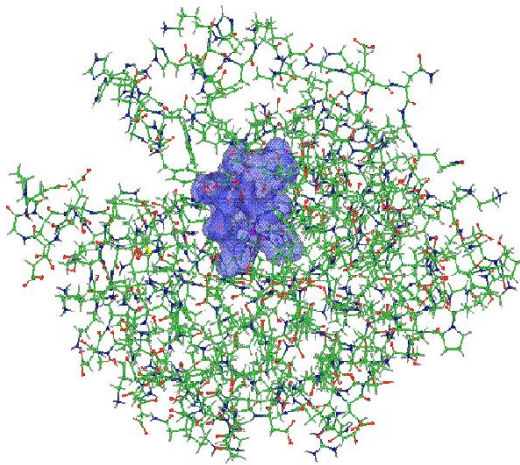
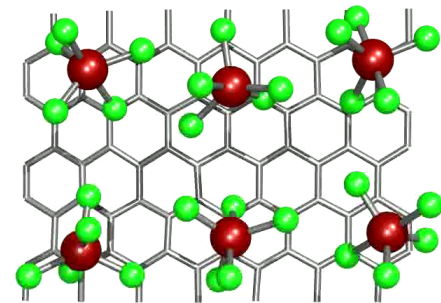
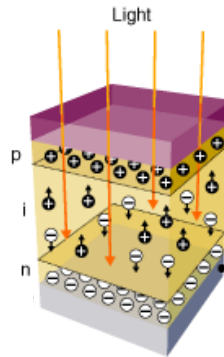
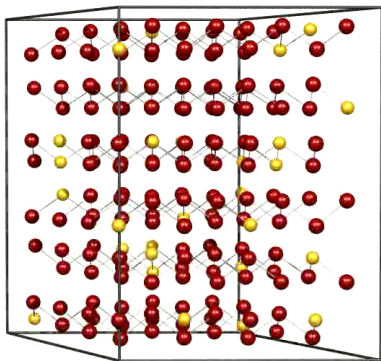
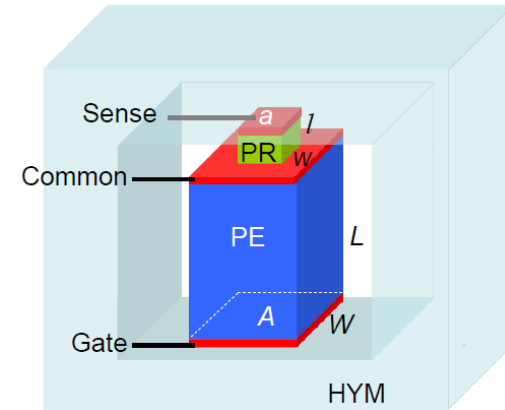
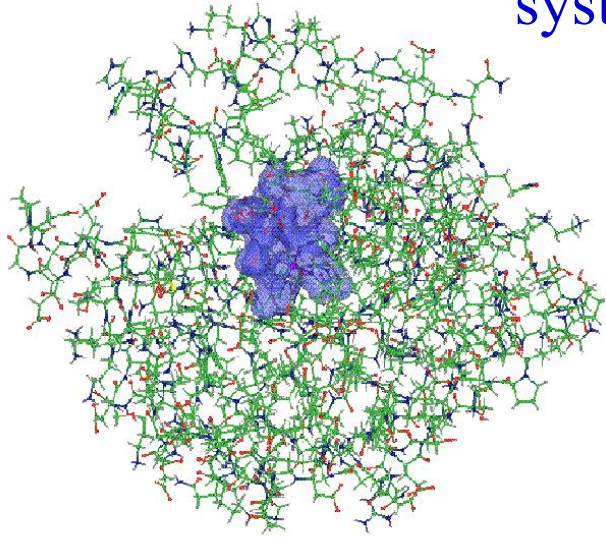


OpenAtom Project: Ground and Excited Electronic State Simulations for large systems on massively parallel platforms

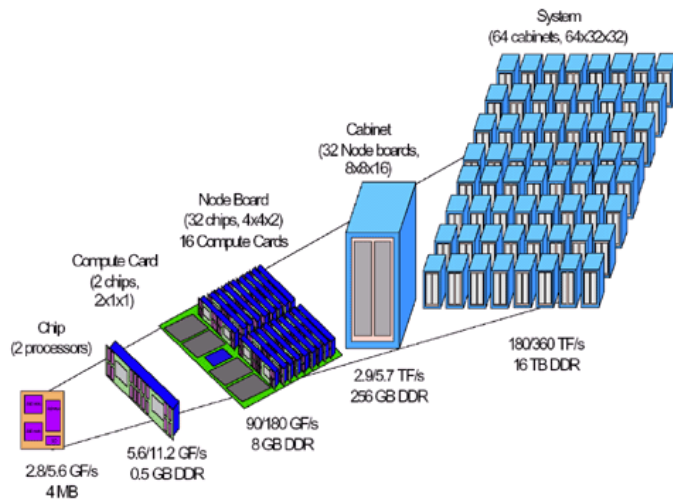
PIs: G.J. Martyna (IBM), S. Ismail-Beigi (Yale) and L.V. Kale (UIUC)



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

$$\frac{\delta E}{\delta n(r)} = 0 \quad \text{equivalent to Kohn-Sham equations}$$

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

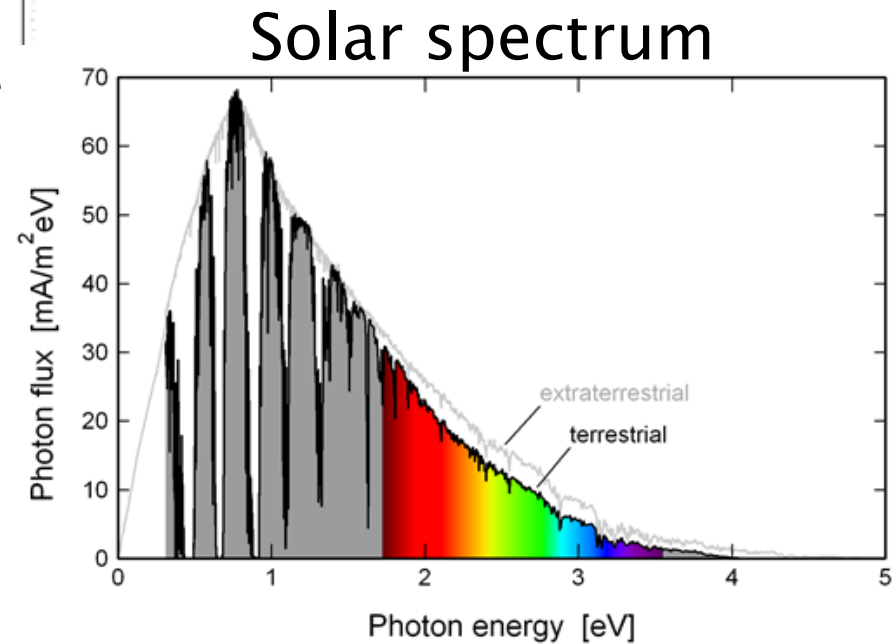
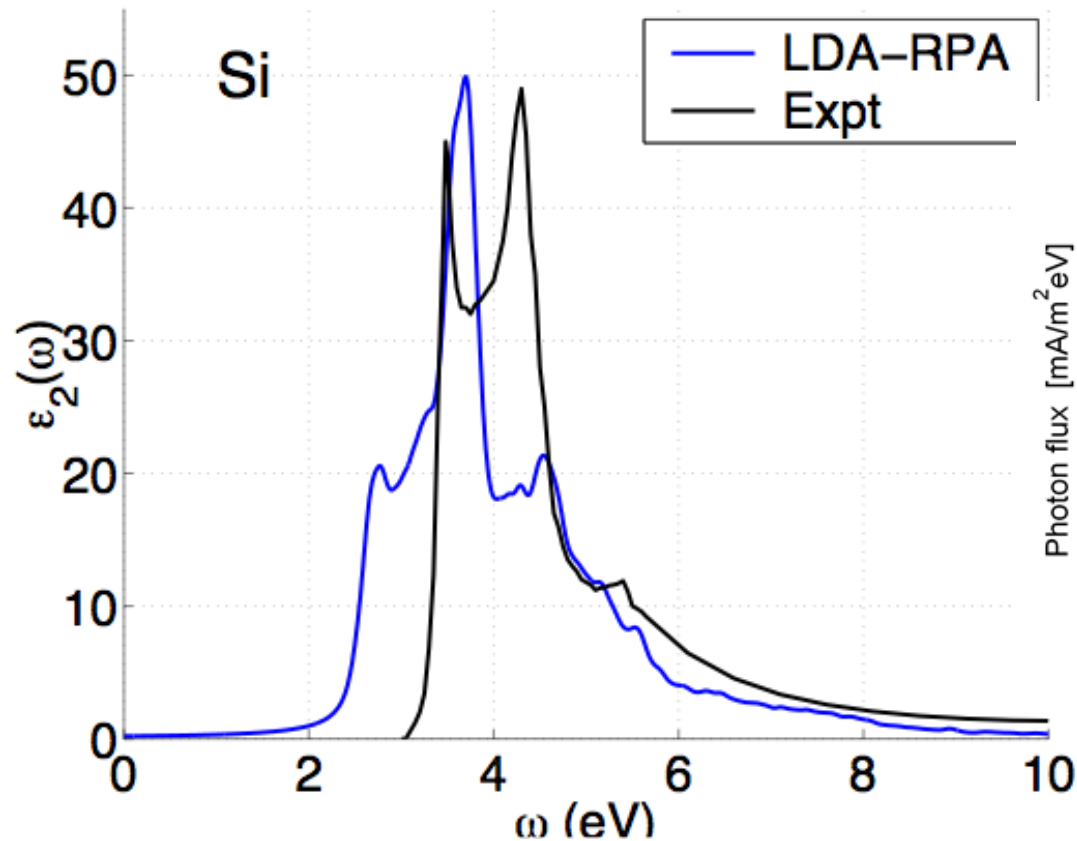
- LDA/GGA for E_{xc} : good geometries and total energies
- Bad band gaps and excitations

DFT: problems with excitations

Energy gaps (eV)

Material	LDA	Expt. [1]
Diamond	3.9	5.48
Si	0.5	1.17
LiCl	6.0	9.4

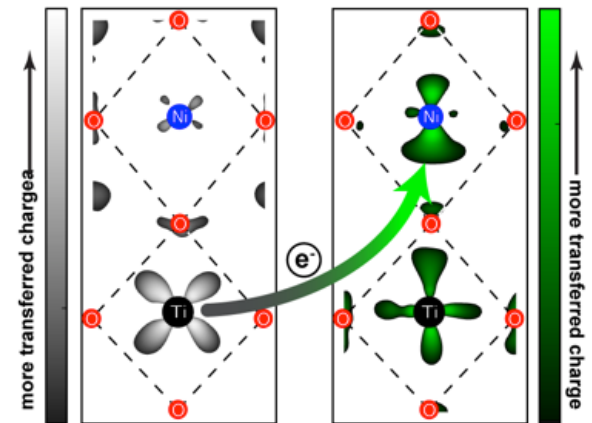
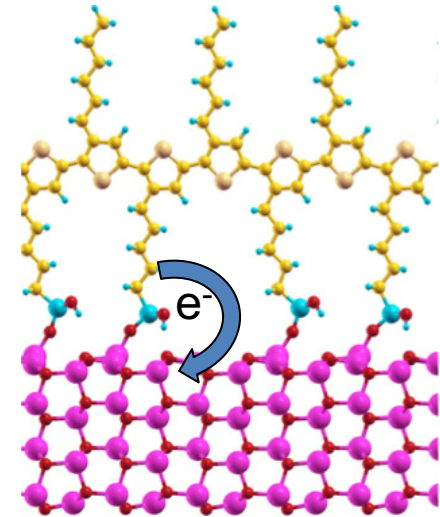
[1] Landolt-Bornstien, vol. III; Baldini & Bosacchi, *Phys. Stat. Solidi* (1970).



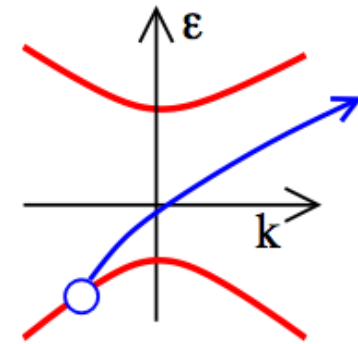
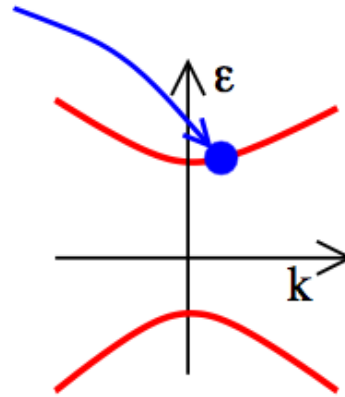
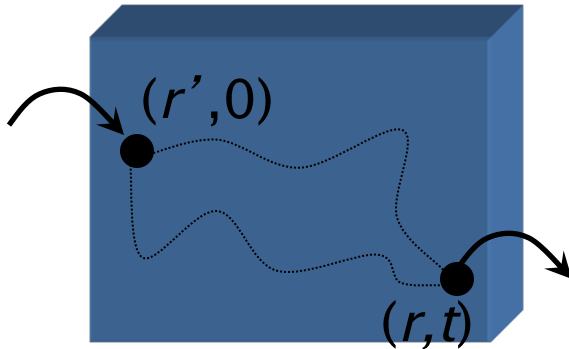
DFT: problems with energy alignment

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:

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

DFT:

$$\Sigma \approx iG_1 W \quad , \quad W = \epsilon^{-1}(\omega) * v_c \quad (RPA)$$

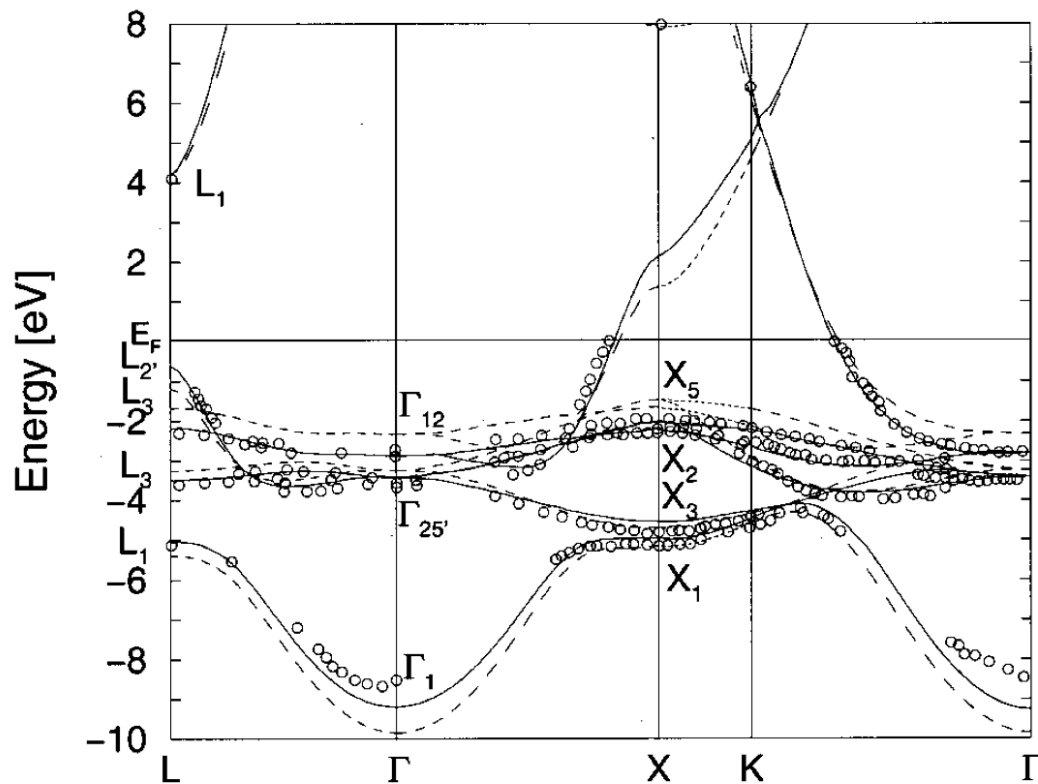
$$\left[-\frac{\nabla^2}{2} + V_{ion}(r) + V_H(r) + \underline{V_{xc}(r)} \right] \psi_j(r) = \epsilon_j \psi_j(r)$$

Green's functions successes

Quasiparticle gaps (eV)

Material	LDA	GW	Expt.
Diamond	3.9	5.6*	5.48
Si	0.5	1.3*	1.17
LiCl	6.0	9.1*	9.4
SrTiO ₃	2.0	3.4-3.8	3.25

* Hybertsen & Louie, *Phys. Rev. B* (1986)

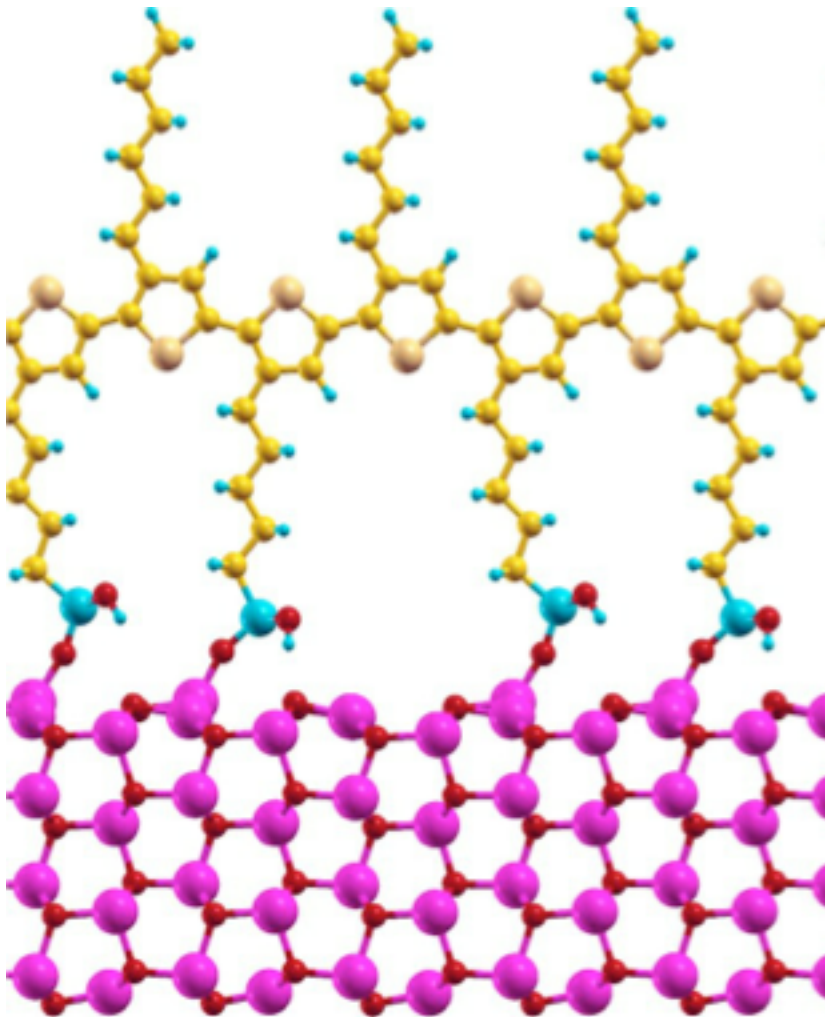


Band structure of Cu

Strokov *et al.*, PRL/PRB (1998/2001)

What is a big system for GW?

P3HT polymer



Zinc oxide nanowire

Band alignment for
this potential
photovoltaic system?

100s of atoms/unit cell

Not possible *routinely*
(with current software)

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

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

$P(r, r')$ = Response of electron density $n(r)$ at position r
to change of potential $V(r')$ at position r'

What is so expensive in GW?

One key element : response of electrons to perturbation

$$P(r, r') = \frac{\partial n(r)}{\partial V(r')} = -2 \sum_v^{\text{filled}} \sum_c^{\text{empty}} \frac{\psi_v(r)\psi_c(r)\psi_v(r')\psi_c(r')}{\epsilon_v - \epsilon_c}$$

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 : ϵ_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

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

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

Stage 5 : put together ϵ_i , $\psi_i(r)$ and $\epsilon^{-1}(\omega) \rightarrow$ self-energy $\Sigma(\omega)$

Steps for typical G_0W_0 calculation

Stage 1 : Run DFT calc. on structure \rightarrow output : ϵ_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 $\rightarrow P(G, G')$

Most expensive

Stage 3 : compute and invert dielectric screening function

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

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

Stage 5 : put together ϵ_i , $\psi_i(r)$ and $\epsilon^{-1}(\omega) \rightarrow$ self-energy $\Sigma(\omega)$

G versus R space P calculation

G-space:

$$P(G, G') = - \sum_{v,c} \langle c | e^{-iG \cdot r} | v \rangle \langle v | e^{iG' \cdot r} | c \rangle \frac{2}{\epsilon_v - \epsilon_c}$$

FFT [$\psi_c^*(r)\psi_v(r)$]

- 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

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

R-space:

$$P(r, r') = - \sum_{v,c} \psi_c^*(r) \psi_v(r) \psi_v^*(r') \psi_c(r') \frac{2}{\epsilon_v - \epsilon_c}$$

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

$$P(r, r')$$



FFT N_r rows

$$P(G, r')$$



FFT N_r columns

$$P(G, G')$$

N_r : # r grid

$N_r \approx 4N_c$

- $N_v + N_c + 8N_c$ FFTs needed
- Big $O(N^4)$ matrix multiply

Eric Mikida

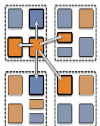


Parallel Implementation

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

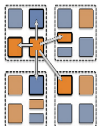
Basic Computation

$$f_{lm} = \psi_l \times \psi_m \text{ for all } l, m$$
$$P += f_{lm} f_{lm}^T \text{ for all } f$$

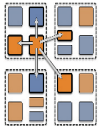
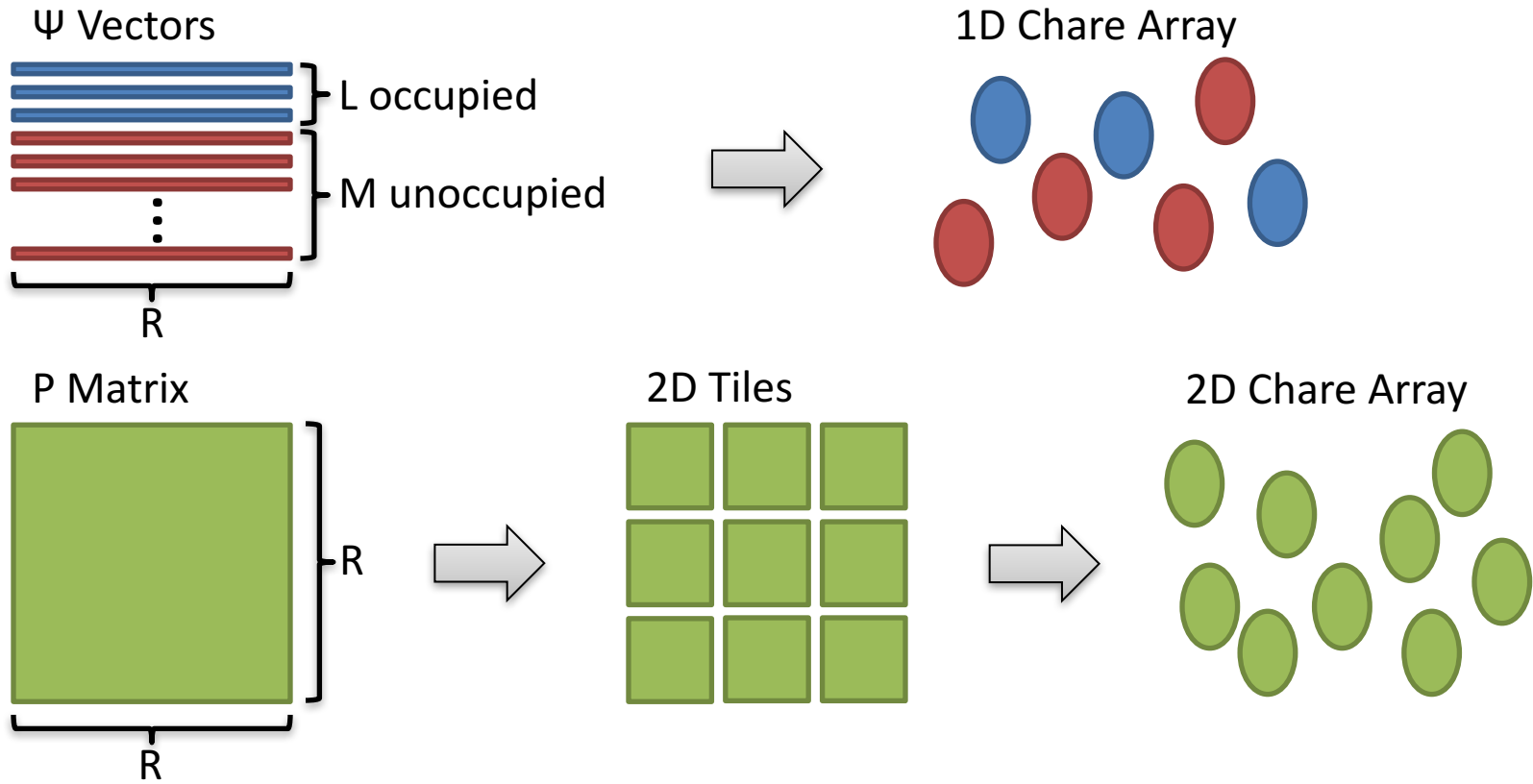


GW-BSE Memory Concerns

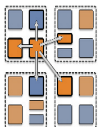
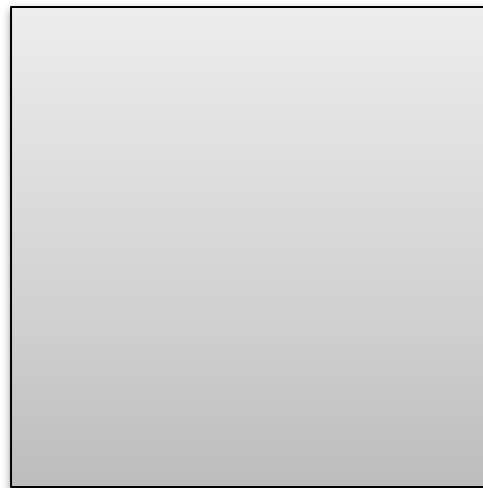
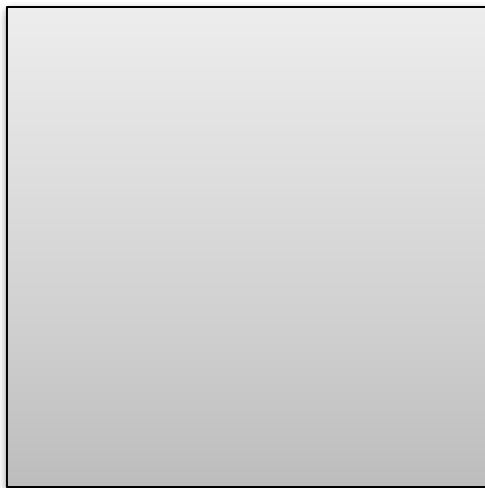
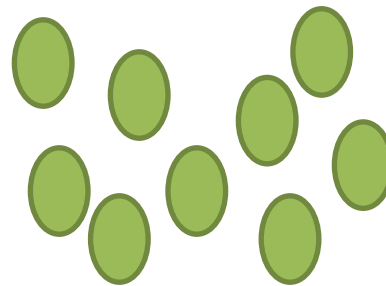
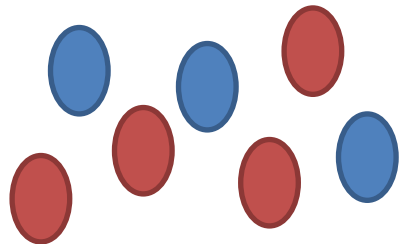
- 1 MB per state
- 10,000 total states per k-point
- 10 k-points
- 100 GB to store all states
- 1 TB to store P
- 90,000,000 f vectors (90 TB total)



Parallel Decomposition

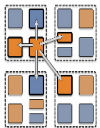
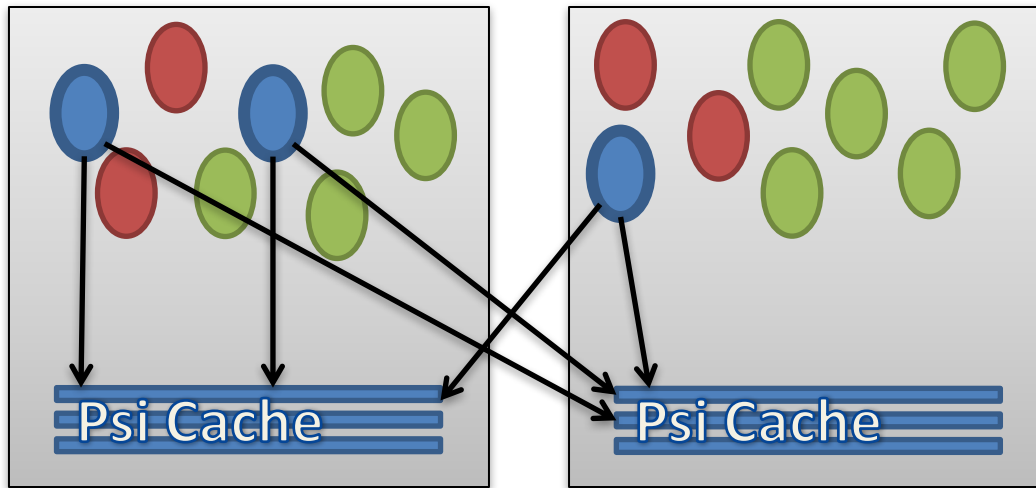


Parallel Decomposition



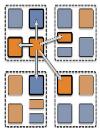
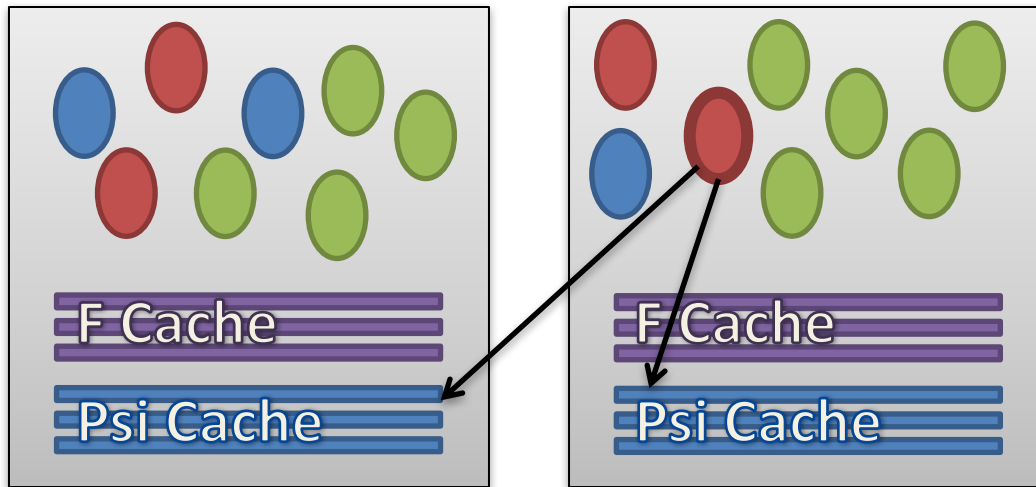
Parallel Decomposition

1. Duplicate occupied states on each node



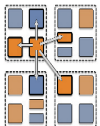
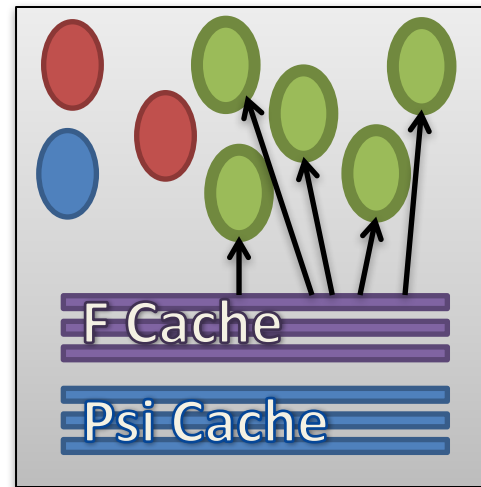
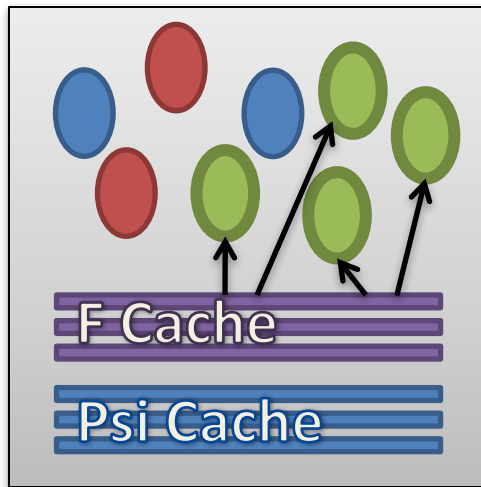
Parallel Decomposition

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



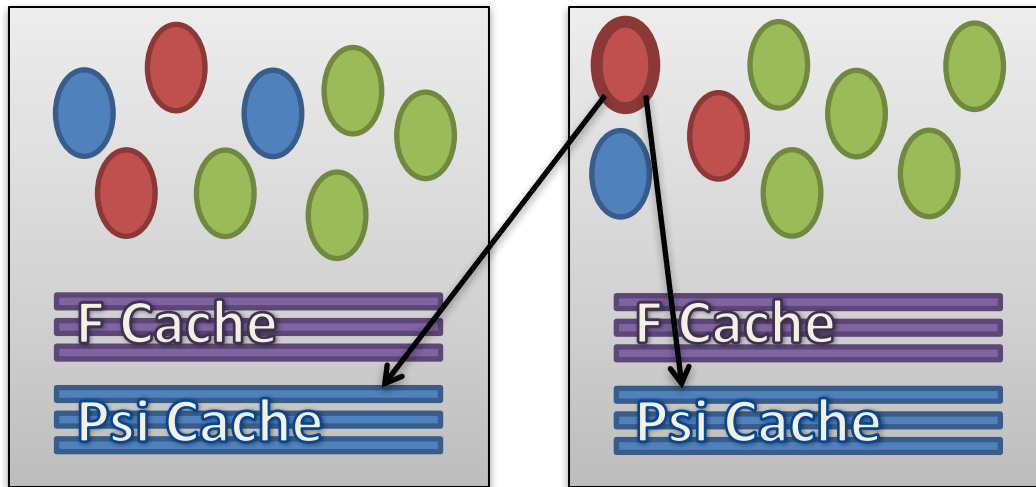
Parallel Decomposition

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**



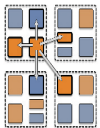
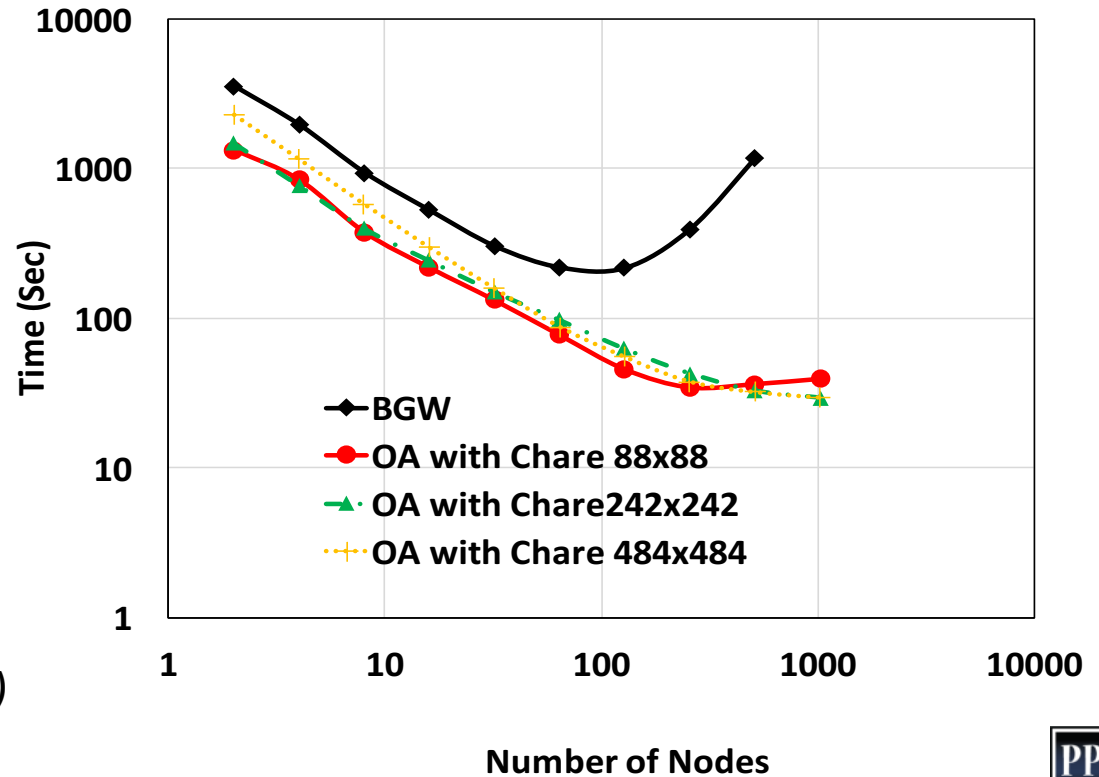
P Formation Scaling

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

~0.2MB per state

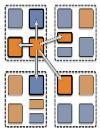
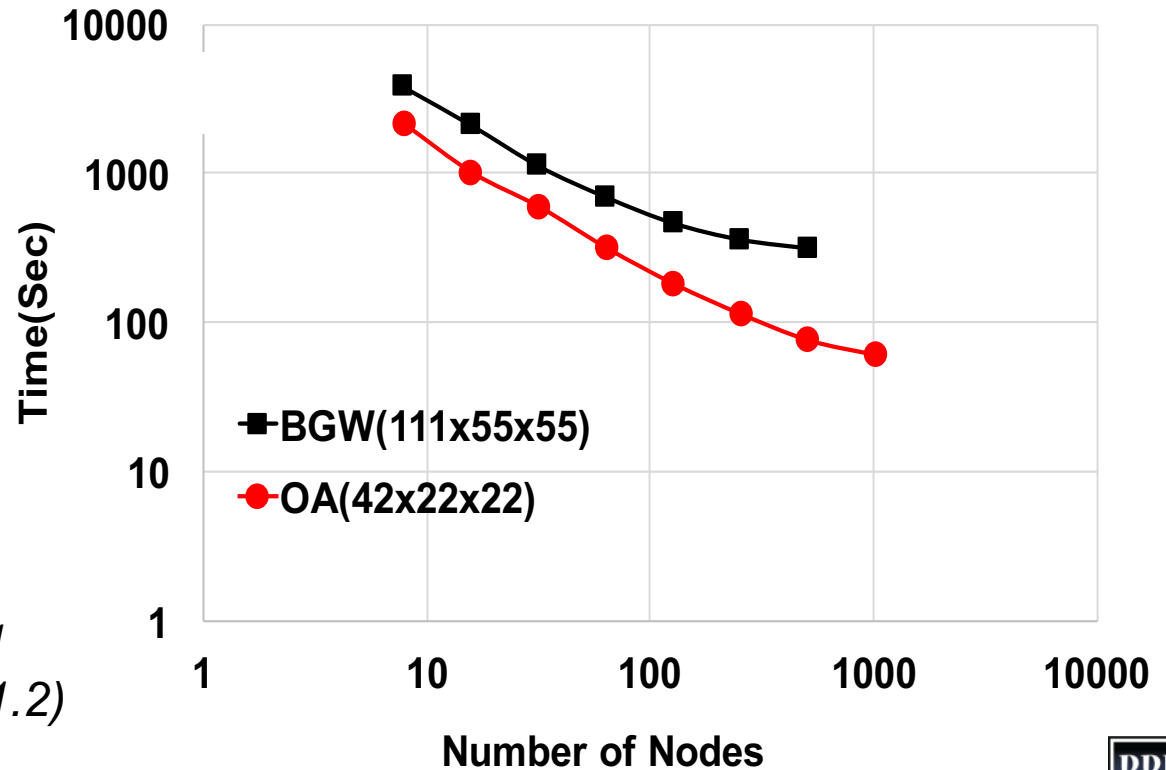
216 occupied

1832 unoccupied

1 k point

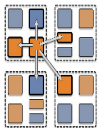
32 processors per node on
Vesta (IBM BG/Q @ ANL)

*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

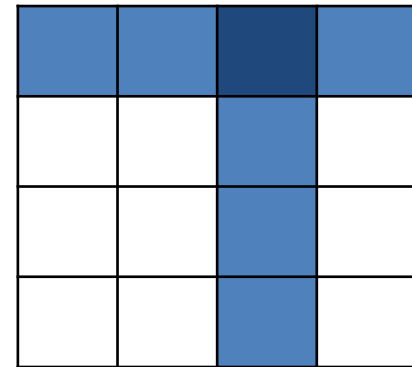


Epsilon Inverse

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

Basic Computation

Initial: $X = \varepsilon * \varepsilon^T$
Step 1: $M1 = 2I - A * X$
Step 2: $X1 = X * M1$
Converge on ($X = X1$)



CLA Matrix Algorithm - 2D



Self-Energy Calculation

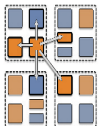
- Operation on pairs of f_{nl} where n is from an input set of state indices
- Bare Exchange and Screened Exchange

Basic Computation

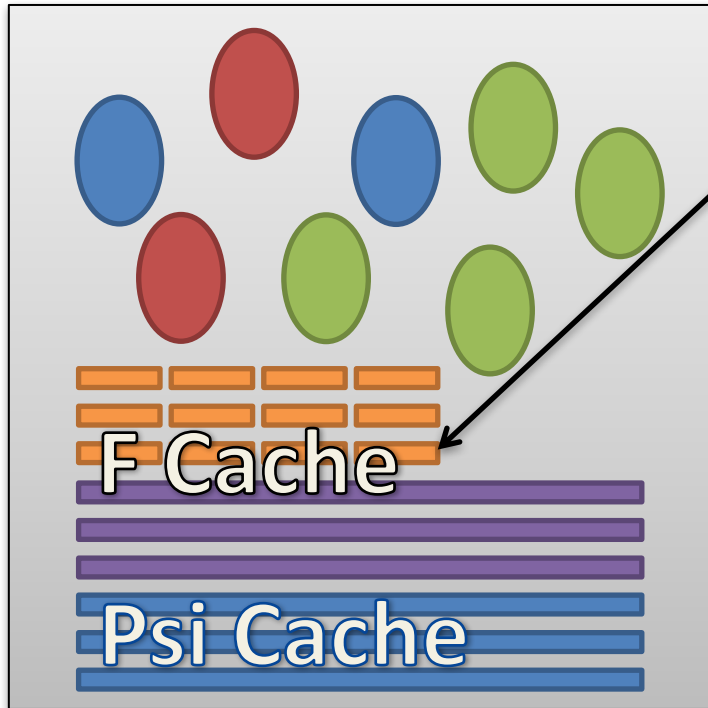
$$f_{nl} A f_{n'l}^T \text{ for all } n, l$$

$$\text{Screened: } A = \epsilon$$

$$\text{Bare: } A = v(g)$$

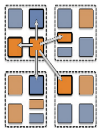


Parallel Decomposition



- 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

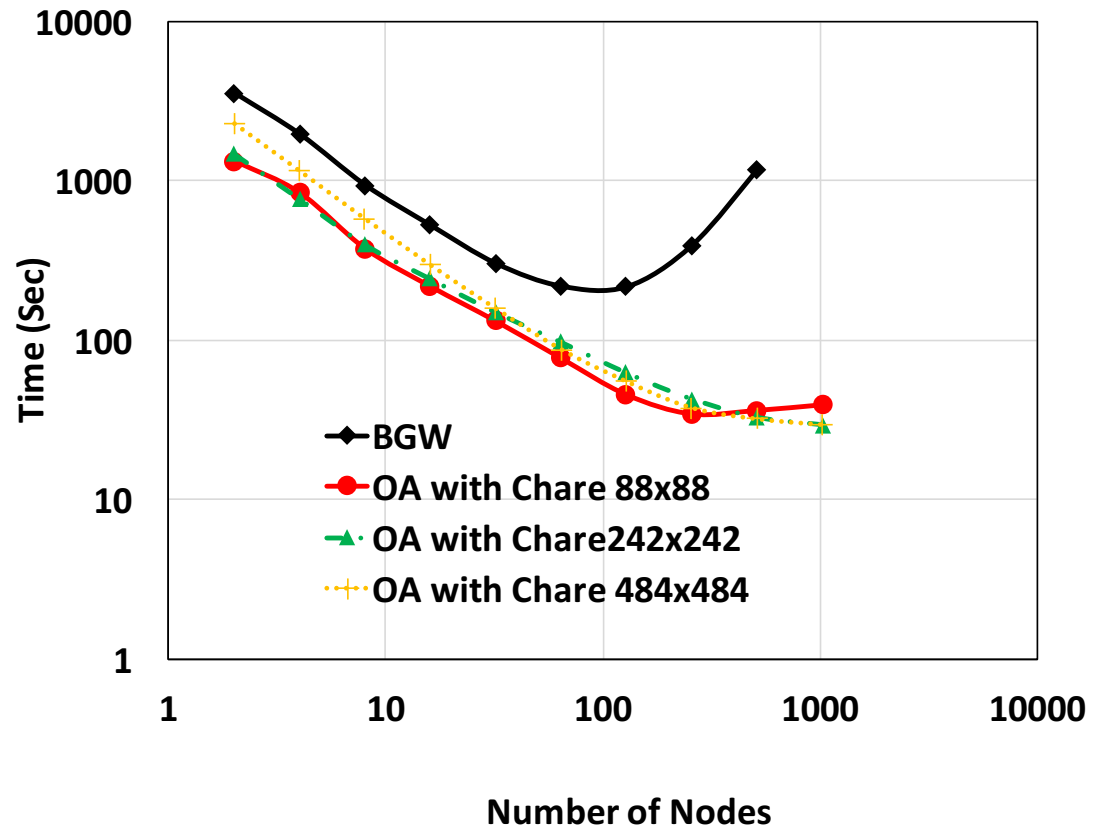


Sohrab Ismail-Beigi



Parallel performance: P calculation

54 atom bulk Si
108 occupied
1000 unoccupied
1 k point
32 processors per node

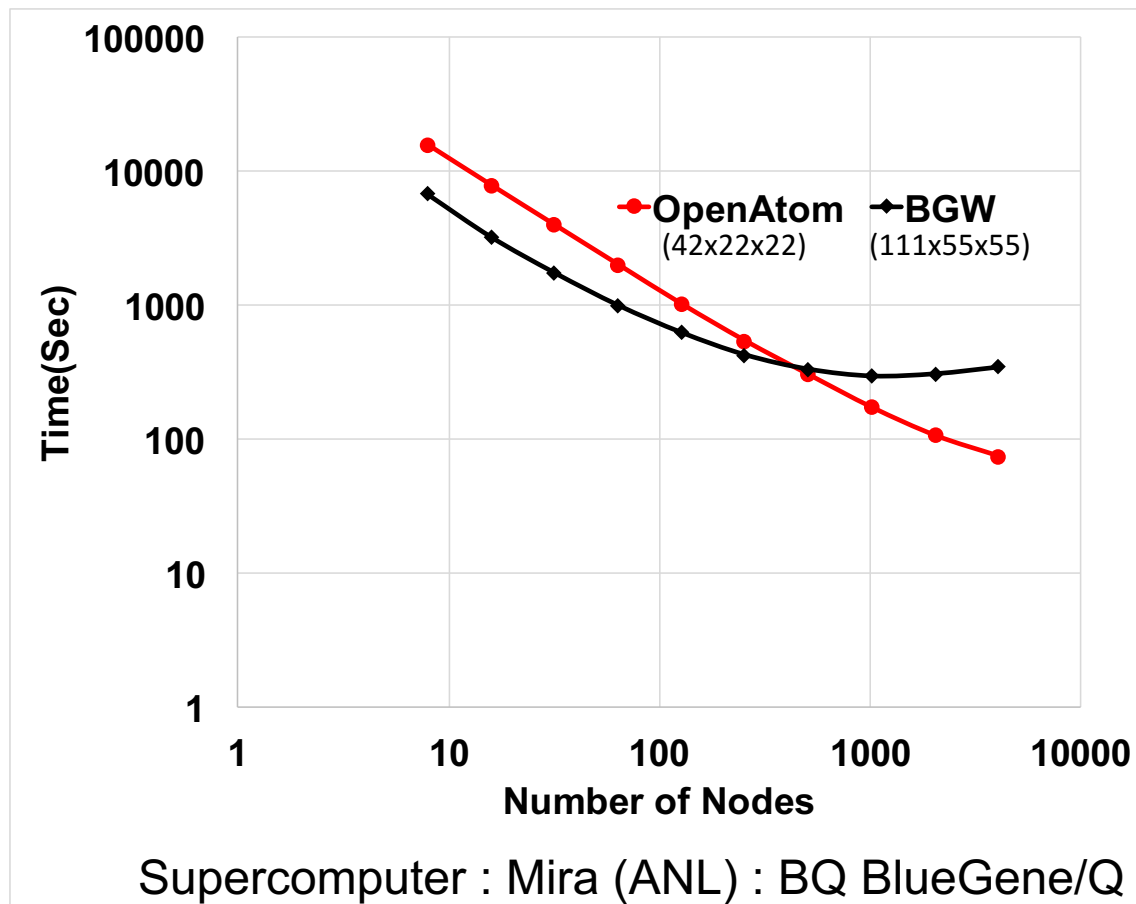


Supercomputer : Vesta (ANL) : BlueGene/Q

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

Parallel performance: P calculation

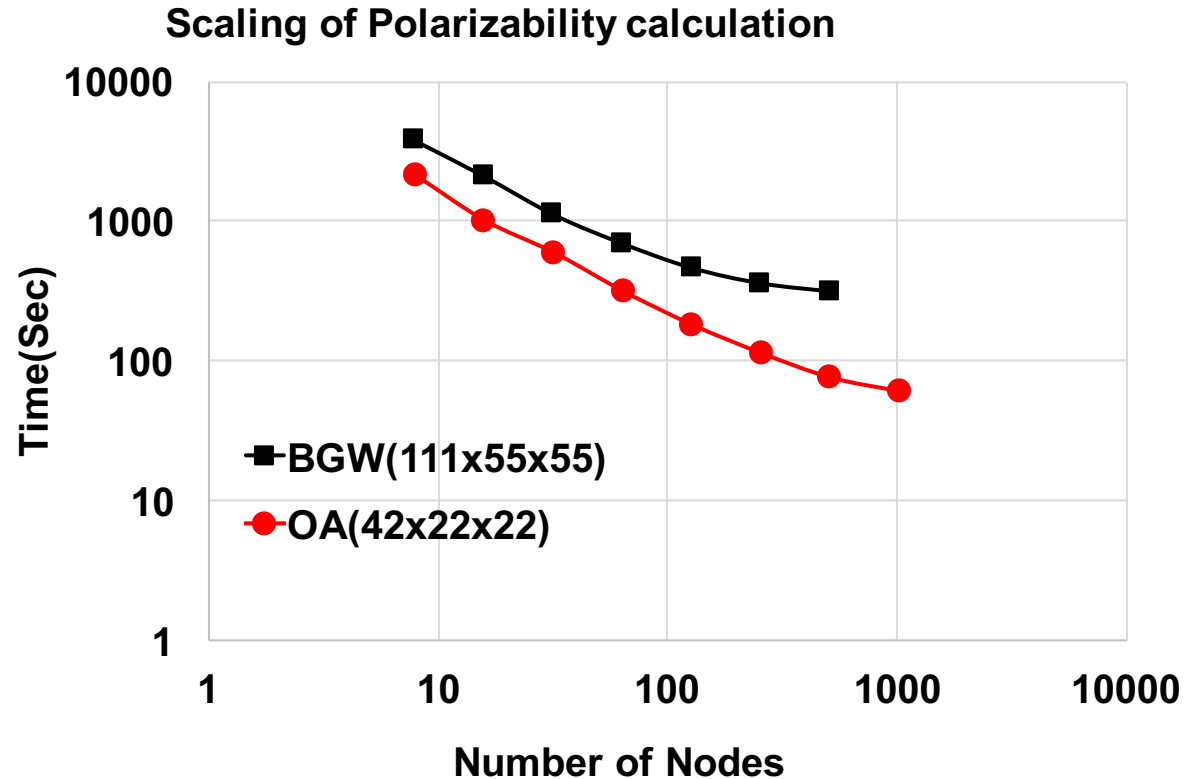
108 atom bulk Si
216 occupied
1832 unoccupied
1 k point
16 processors per node



*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

Phase		Serial	Parallel
1	Compute P in RSpace	Complete	Complete
2	FFT P to GSpace	Complete	Complete
3	Invert epsilon	Complete	Complete
4	Plasmon pole	Complete	In Progress
5	COHSEX self-energy	Complete	In Progress
6	Dynamic self-energy	In Progress	Future
7	Coulomb Truncation	Future	Future

Aim to release COHSEX version early summer 2017

Minjung Kim



Static polarizability calculations

$$P(r, r') = \sum_{v, c} \psi_c^*(r) \psi_v(r) \psi_v^*(r') \psi_c(r') \frac{2}{\epsilon_v - \epsilon_c}$$

$$N^4 \quad \Rightarrow \quad N^3 \quad ?$$

Giustino, Cohen, and Louie, *PRB* **81** (2010)

Wilson, Gygi, and Galli, *PRB* **78** (2008)

Liu, Kaltak, Klimes, and Kresse, *PRB* **94** (2016)

Cubic scaling algorithm – 1. Interpolation

$$P(r, r') = -2 \sum_{v,c} \frac{\psi_v(r) \psi_c^*(r) \psi_c(r') \psi_v^*(r')}{\epsilon_c - \epsilon_v}$$

$$N_r^2 N_c N_v \sim N^4$$

$$A(r, r'; z) = \sum_c \frac{\psi_c^*(r) \psi_c(r')}{\epsilon_c - z}$$

$$N_r^2 N_c N_z$$

1. Save values over some z grid

2. Interpolate

$$P(r, r') = -2 \sum_v \psi_v(r) \psi_v^*(r') A(r, r'; \epsilon_v)$$

$$N_r^2 N_v N_{int}$$

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

N_z : number of points to be evaluated

N_{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:
$$\int_0^\infty f(x) e^{-x} dx \approx \sum_k^{N_{GL}} \omega_k f(x_k)$$

w_k : weight
 x_k : node
 N_{GL} : # quadrature nodes

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

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

separable!

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

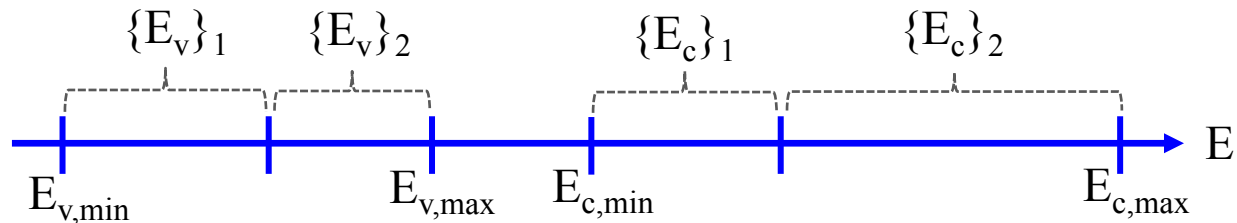
3. Windowing:

- Observation: N_{GL} depends on $\frac{E_{bw}}{E_{gap}}$ $N_r^2 N_{GL} (N_c + N_v)$ E_{bw} : band width ($E_{c,max} - E_{v,min}$)
 E_{gap} : band gap

$$P(r, r') = \sum_l^{N_{wv}} \sum_m^{N_{wc}} P_{lm}(r, r')$$

N_{wv} : # windows for E_v
 N_{wc} : # of windows for E_c

- Example: 2 by 2 windows $P = P_{11} + P_{21} + P_{12} + P_{22}$



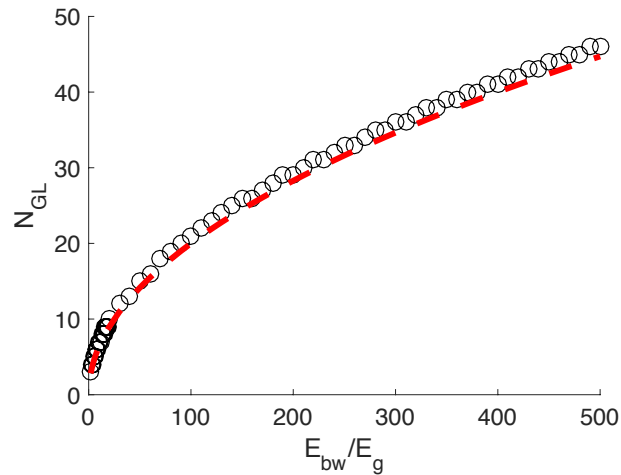
- Windowing can save computational costs
- Useful for materials with small band gap

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



- $Cost = N_r^2 N_{GL} (N_c + N_v)$

1. $N_{GL} \propto \sqrt{\frac{E_{bw}}{E_g}}$

2. $(N_c + N_v)/N \propto (E_{bw} - E_g)$

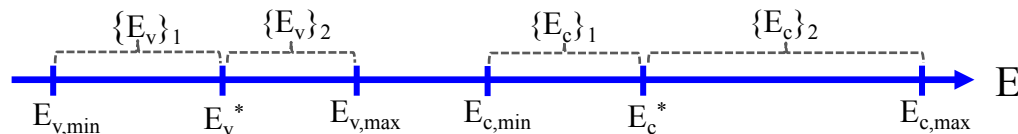


computation cost can be estimated by E_{bw} and E_g

$$C \propto \sum_l^{N_{vw}} \sum_m^{N_{cw}} \sqrt{\frac{E_{bw}^{lm}}{E_g^{lm}}} \left(\frac{E_{vl}^{max} - E_{vl}^{min}}{E_v^{max} - E_v^{min}} N_v - \frac{E_{cm}^{max} - E_{cm}^{min}}{E_c^{max} - E_c^{min}} N_c \right)$$

Estimate the computational costs

Example: 2x2 window

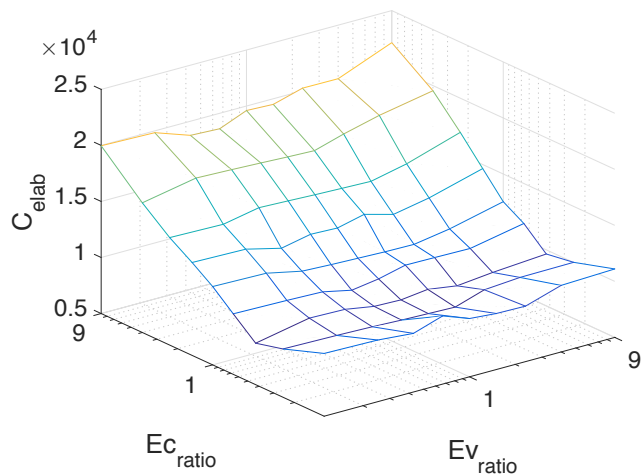


$$E_{\text{bw}} = 2 \text{ Hartree}$$

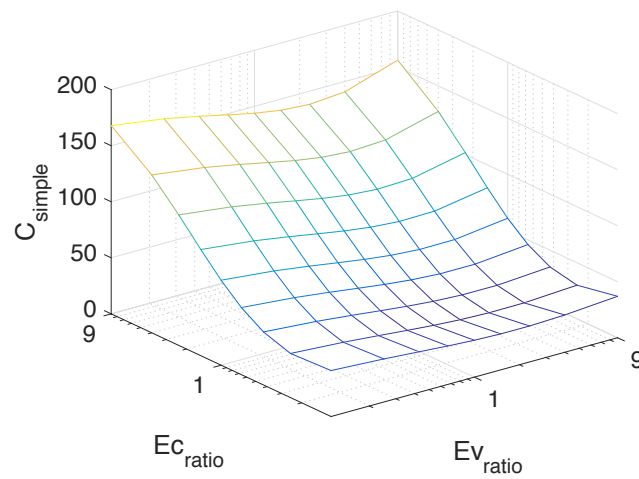
$$E_g = 0.02 \text{ Hartree}$$

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

Real computational costs



Estimated computational costs



Windowing

How many windows for occupied and unoccupied states?

$$E_{vmax} - E_{vmin} = 0.44 \text{ Ha}$$

$$E_{cmax} - E_{cmin} = 1.44 \text{ Ha}$$

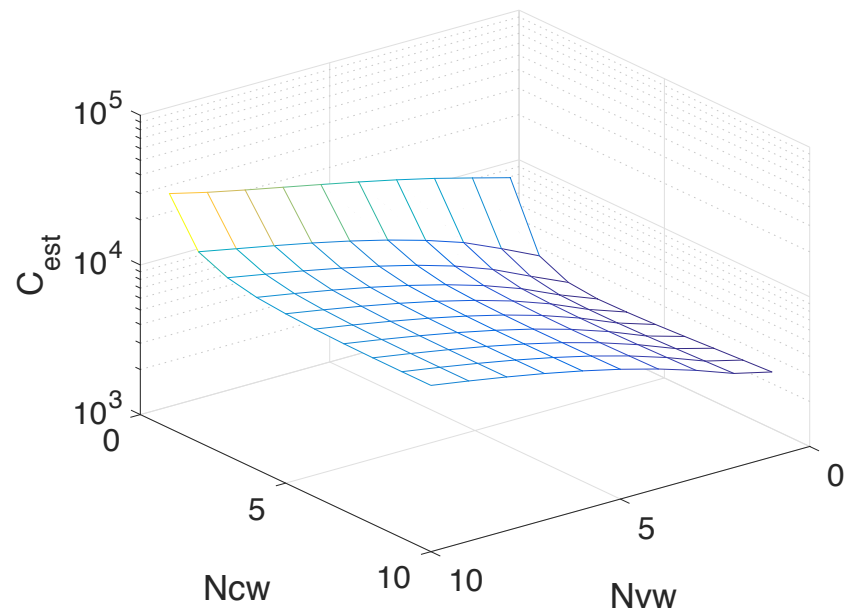
$$E_{bw} = 2 \text{ Ha}$$

$$E_g = 0.02 \text{ Ha}$$

Optimized number of windows:

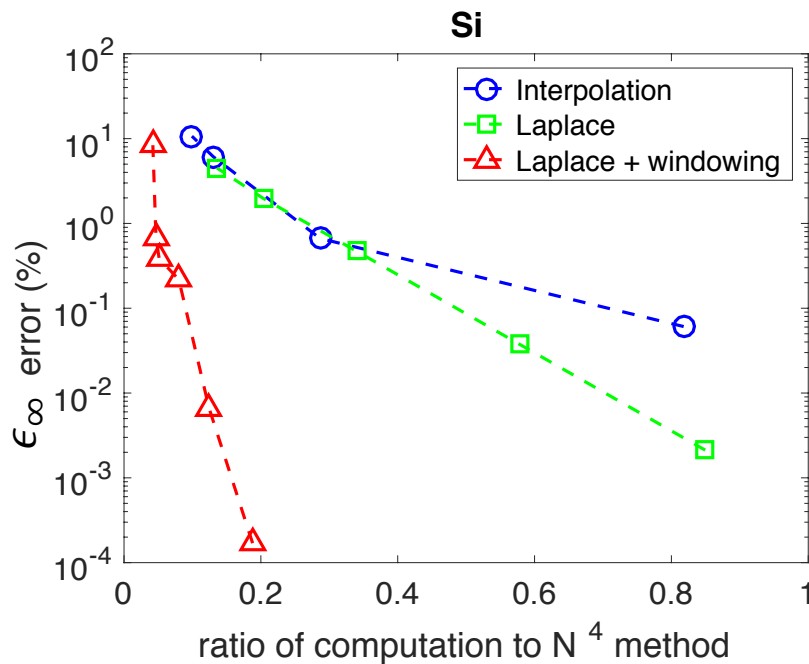
$$N_{vw} = 1$$

$$N_{cw} = 4$$



Results

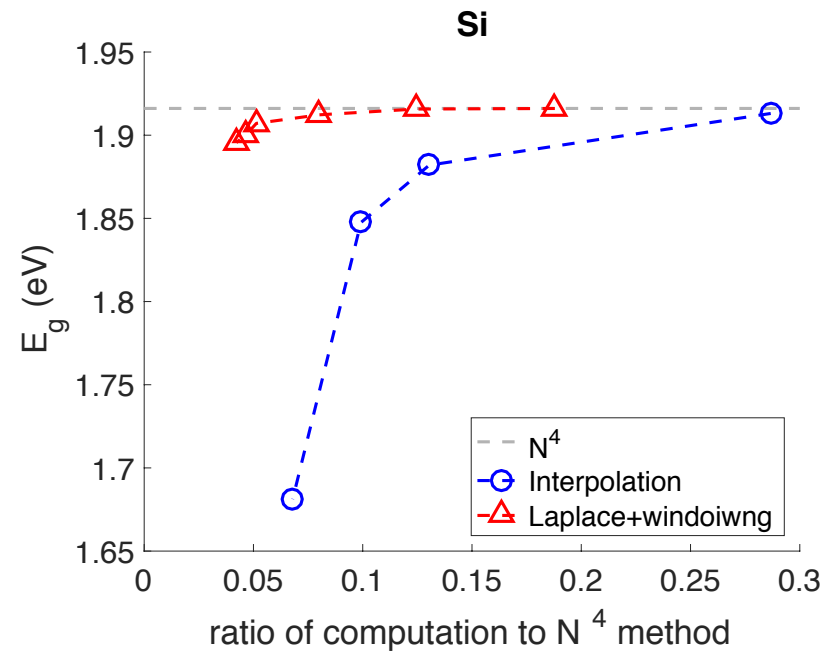
- Si crystal with 16 atoms
- Number of bands: 433
- Number of windows: 1 for N_v & 4 for N_c



Laplace method with windows wins!!

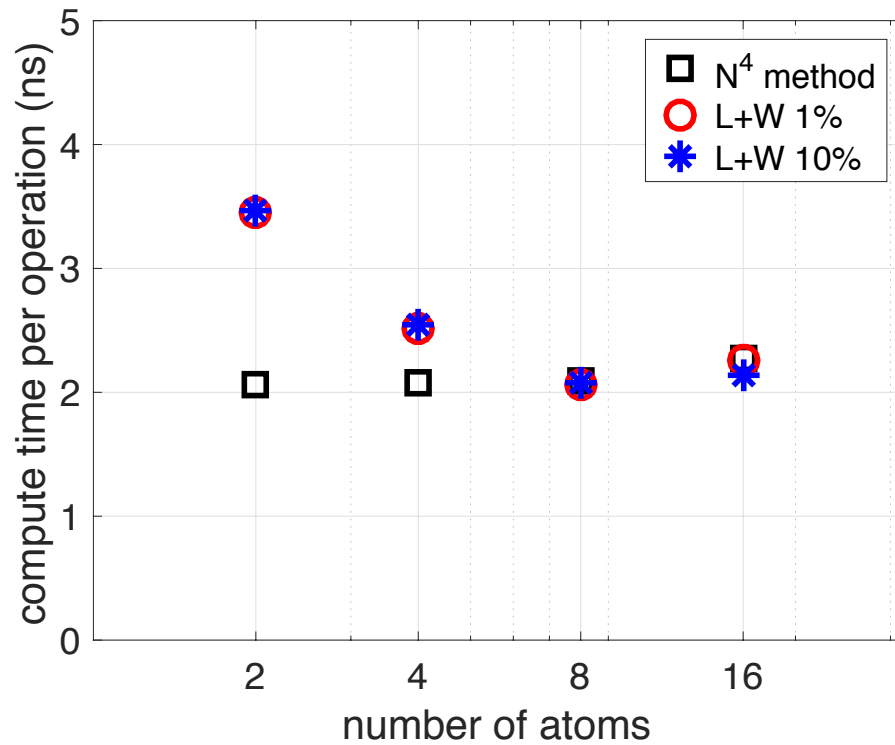
Maximum error of band gap

- Laplace + windowing: 0.02eV
- Interpolation: 0.23eV



Results

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



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

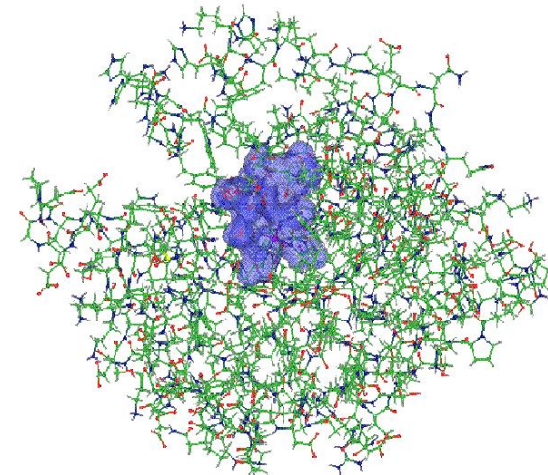
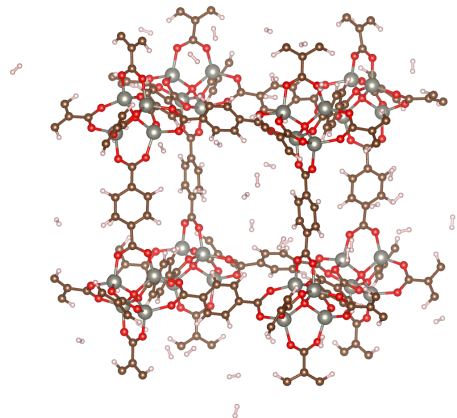
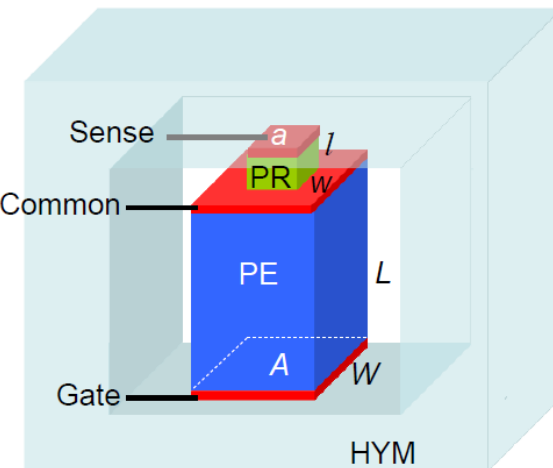
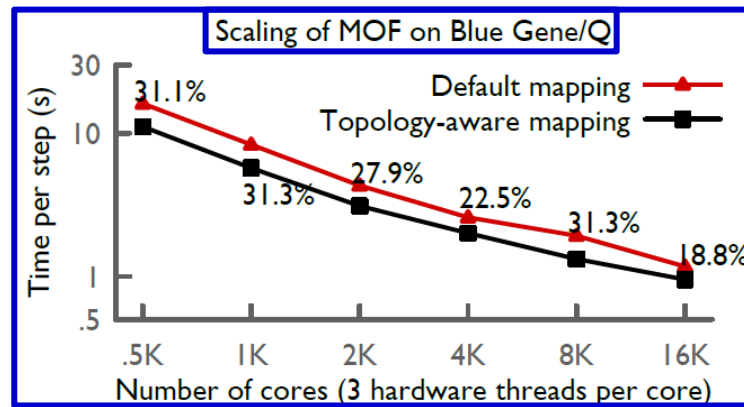
$$L + W : \frac{\text{Operation Time}}{\sum_l^{N_{wv}} \sum_m^{N_{wc}} N_{GL} (N_v + N_c) N_r^2}$$

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



Kohn-Sham Density Functional Theory (KS-DFT):

A workhorse of computational science.

- **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

$$\rho(\mathbf{r}, \mathbf{r}') = \sum_{I=1}^{N_{KS}} \psi_I(\mathbf{r}) \psi_I^*(\mathbf{r}'), \quad n(\mathbf{r}) = \rho(\mathbf{r}, \mathbf{r}), \quad N_{KS} = (\# \text{ electrons})/2$$

an *orthonormal set of KS states*, $\langle \psi_I | \psi_J \rangle = \delta_{IJ}$.



Walter Kohn,
Nobel Chemistry
1998

- **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*,

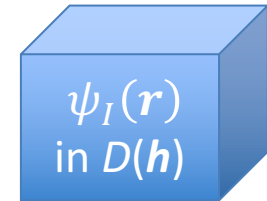
$$E[n(\mathbf{r})] = -\frac{\hbar^2}{2m_e} \int d\mathbf{r} (\nabla^2 \rho(\mathbf{r}, \mathbf{r}'))|_{\mathbf{r}'=\mathbf{r}} + \frac{e^2}{2} \int d\mathbf{r} d\mathbf{r}' \frac{n(\mathbf{r})n(\mathbf{r}')}{|\mathbf{r} - \mathbf{r}'|} \\ + e \int d\mathbf{r} n(\mathbf{r}) V_{ext}(\mathbf{r}; N) + E_{xc}[n(\mathbf{r})], \quad N = \# \text{ ions}, N_{KS} \sim N.$$

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

$$E_{xc}[n(\mathbf{r})] \approx \int d\mathbf{r} \varepsilon_{xc}(n(\mathbf{r}), \nabla n(\mathbf{r}))$$

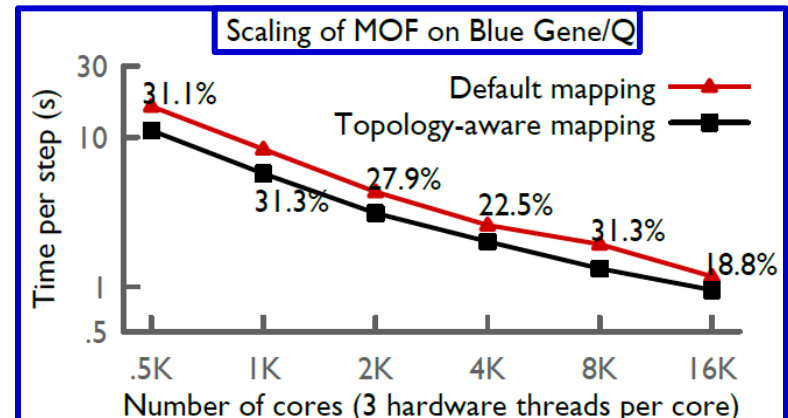
KS-DFT in OpenAtom

- **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.

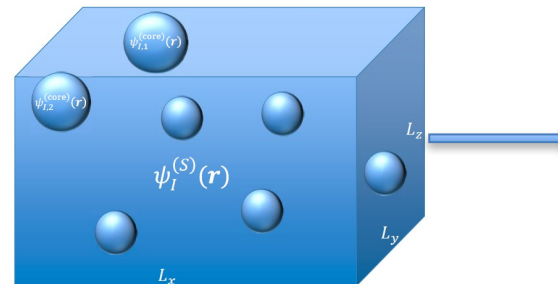


- **PW-KS-DFT in OpenAtom - Disadvantages:**

- *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*.



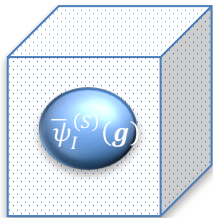
Goal: Implement $N^2 \log N$ EES-based PAW with high parallel efficiency in OpenAtom.

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:

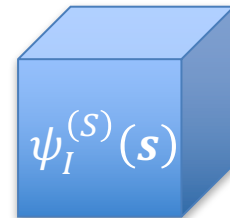
$$\psi_I(\mathbf{r}) = \psi_I^{(S)}(\mathbf{r}) + \sum_{J=1}^N \psi_{IJ}^{(\text{core})}(\mathbf{r}), \quad \psi_{IJ}^{(\text{core})}(\mathbf{r}) = 0, |\mathbf{r} - \mathbf{R}_J| > R_{pc}$$

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



$$\psi_I^{(S)}(\mathbf{s}) = \frac{1}{\sqrt{V}} \sum_{|\mathbf{g}| < G_c/2} \bar{\psi}_I^{(S)}(\mathbf{g}) \exp(i\hat{\mathbf{g}}\mathbf{s})$$

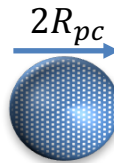
$\mathbf{r} = \mathbf{h}\mathbf{s}, V = \det \mathbf{h}, \mathbf{g} = 2\pi\mathbf{h}^{-1}\hat{\mathbf{g}}, \hat{\mathbf{g}} \in \text{integer}$



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

$$\psi_{IJ}^{(\text{core})}(\mathbf{r}) = \Delta p(\mathbf{r} - \mathbf{R}_J) Z_{IJ}^{(S)}, \quad \Delta p(\mathbf{r} - \mathbf{R}_J) = 0, |\mathbf{r} - \mathbf{R}_J| > R_{pc}$$

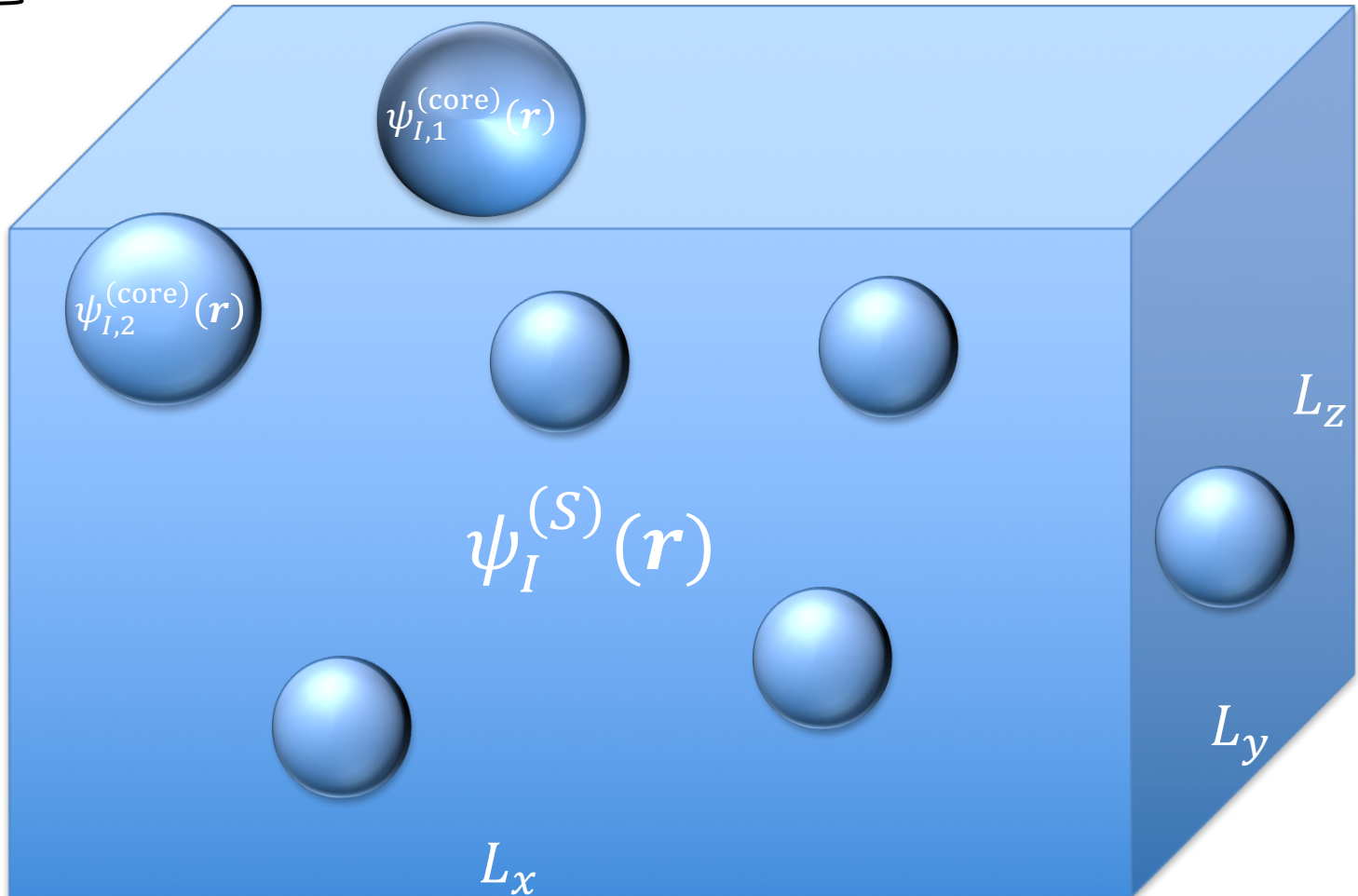
$$Z_{IJ}^{(S)} = \langle p_J^{(S)} | \psi_I^{(S)} \rangle = \int d\mathbf{r}^3 p^{(S)}(\mathbf{r} - \mathbf{R}_J) \psi_I^{(S)}(\mathbf{r}), \quad p^{(S)}(\mathbf{r} - \mathbf{R}_J) = 0, |\mathbf{r} - \mathbf{R}_J| > R_{pc}$$



PAW Basics: Example KS state

$$\mathbf{h} = \begin{bmatrix} L_x & 0 & 0 \\ 0 & L_y & 0 \\ 0 & 0 & L_z \end{bmatrix}$$

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 $D(\mathbf{h})$.

PAW Basics: KS-DFT within LDA under

periodic boundary conditions at Γ

The whole enchilada:

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

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

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

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

$$E_{ext} = - \int_{D(\mathbf{h})} d\mathbf{r} \sum_J \sum_m \frac{eQ_J n(\mathbf{r})}{|\mathbf{r} - \mathbf{r}_J + \mathbf{m}\mathbf{h}|}$$

Non-interacting electron kinetic energy: Smooth and core terms

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

$$E_{NIKE}^{(S)} = -\frac{\hbar^2}{2m_e} \int_{D(\mathbf{h})} d\mathbf{r} \sum_I \langle \psi_I^{(S)} | \nabla^2 | \psi_I^{(S)} \rangle, \quad E_{NIKE}^{(core1)} = -\frac{\hbar^2}{2m_e} \sum_{IJ} Z_{IJ}^{(S)} Z_{IJ}^{(\nabla^2 S, \Delta)}, \quad E_{NIKE}^{(core2)} = -\frac{\hbar^2}{2m_e} \sum_J Z_J^{(S,2)} \langle \Delta p | \nabla^2 | \Delta p \rangle$$

Exchange Correlation energy: Smooth and core terms

$$E_{xc} = E_{xc}^{(S)} + E_{xc}^{(core)} = \int_{D(\mathbf{h})} d\mathbf{r} \varepsilon_{xc}(n^{(S)}(\mathbf{r})) + \sum_J \int_{D(R_{pc})}^{core} d\mathbf{r} \left[\varepsilon_{xc}(n_J(\mathbf{r})) - \varepsilon_{xc}(n_J^{(S)}(\mathbf{r})) \right]$$

$$n^{(S)}(\mathbf{r}) = \sum_I |\psi_I^{(S)}(\mathbf{r})|^2, \quad n_J(\mathbf{r}) = n^{(S)}(\mathbf{r} - \mathbf{R}_J) + n^{(core1)}(\mathbf{r} - \mathbf{R}_J) + n^{(core2)}(\mathbf{r} - \mathbf{R}_J), \quad n_J^{(S)}(\mathbf{r}) = n^{(S)}(\mathbf{r} - \mathbf{R}_J)$$

$\forall \mathbf{r} \text{ in } D(\mathbf{h})$
 $\forall |\mathbf{r} - \mathbf{R}_J| < R_{pc}$
 $\forall |\mathbf{r} - \mathbf{R}_J| < R_{pc}$

PAW Basics: KS-DFT long/short-range decomposition

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 **\mathbf{g} -space** or **\mathbf{r} -space** alone.

$$E_H = \frac{e^2}{2} \int_{D(\mathbf{h})} d\mathbf{r} \int_{D(\mathbf{h})} d\mathbf{r}' \sum_{\mathbf{m}} \frac{n(\mathbf{r})n(\mathbf{r}')}{|\mathbf{r} - \mathbf{r}' + \mathbf{m}\mathbf{h}|} \quad , \quad E_{ext} = - \int_{D(\mathbf{h})} d\mathbf{r} n(\mathbf{r}) \sum_J \sum_{\mathbf{m}} \frac{eQ_J}{|\mathbf{r} - \mathbf{R}_J + \mathbf{m}\mathbf{h}|}$$

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

$$E_H = E_H^{(\text{short})} + E_H^{(\text{long})} \quad E_{ext} = E_H^{(\text{short})} + E_H^{(\text{long})}$$

$$E_H^{(\text{short})} = \frac{e^2}{2} \int_{D(\mathbf{h})} d\mathbf{r} \int_{D(\mathbf{h})} d\mathbf{r}' \frac{n(\mathbf{r})n(\mathbf{r}') \operatorname{erfc}(\alpha|\mathbf{r} - \mathbf{r}'|)}{|\mathbf{r} - \mathbf{r}'|} \quad E_{ext}^{(\text{short})} = -e \int_{D(\mathbf{h})} d\mathbf{r} n(\mathbf{r}) \sum_J \frac{\operatorname{erfc}(\alpha|\mathbf{r} - \mathbf{R}_J|)}{|\mathbf{r} - \mathbf{R}_J|}$$

$$E_H^{(\text{long})} = \frac{e^2}{2V} \sum_{\mathbf{g} \neq 0}^{|g|^2 < G_c} \frac{4\pi}{|\mathbf{g}|^2} \exp\left(-\frac{|\mathbf{g}|^2}{4\alpha^2}\right) |\bar{n}(\mathbf{g})|^2 - \frac{\pi e^2 |\bar{n}(0)|^2}{2V\alpha^2} \quad E_{ext}^{(\text{long})} = -\frac{e}{V} \sum_{\mathbf{g} \neq 0}^{|g|^2 < G_c} \frac{4\pi}{|\mathbf{g}|^2} \exp\left(-\frac{|\mathbf{g}|^2}{4\alpha^2}\right) \bar{n}(\mathbf{g}) \bar{S}(\mathbf{g}) + \frac{\pi e \bar{n}(0) \bar{S}(0)}{V\alpha^2}$$

$$\bar{S}(\mathbf{g}) = \sum_J Q_J \exp(-i\mathbf{g} \cdot \mathbf{R}_J)$$

Choose α , such that the **\mathbf{g} -space cutoff** = $G_c =$ *pw density cutoff*.

Ensure **\mathbf{r} -space cutoff**, $R_c = (3.5 / \alpha) > R_{pc}$, confines the **\mathbf{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: $\frac{G_c^2}{4} \approx \frac{\gamma^4}{R_c^2}$, $\gamma = \alpha R_c$

	$R_c = 4$ bohr	
PW cutoff: $(\hbar^2 G_c^2 / 8me)$ Ryd	$\gamma = \alpha R_c$	erfc(γ)
5.1	3.0	2.21e-05
9.4	3.5	7.43e-07
16	4.0	1.54e-08

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

- Using FFTs, the $\bar{f}^m(\mathbf{g})$, can be computed exactly in $N \log N$ as:

$$f(\mathbf{s}) = \frac{1}{V} \text{FFT}^{(m,+)}[\bar{f}(\mathbf{g}), G_c], \quad \bar{f}^m(\mathbf{g}) = \frac{V}{N_{\text{FFT}}} \text{FFT}^{(m,-)}[f^m(\mathbf{s}), mG_c], \quad V = \det \mathbf{h}$$

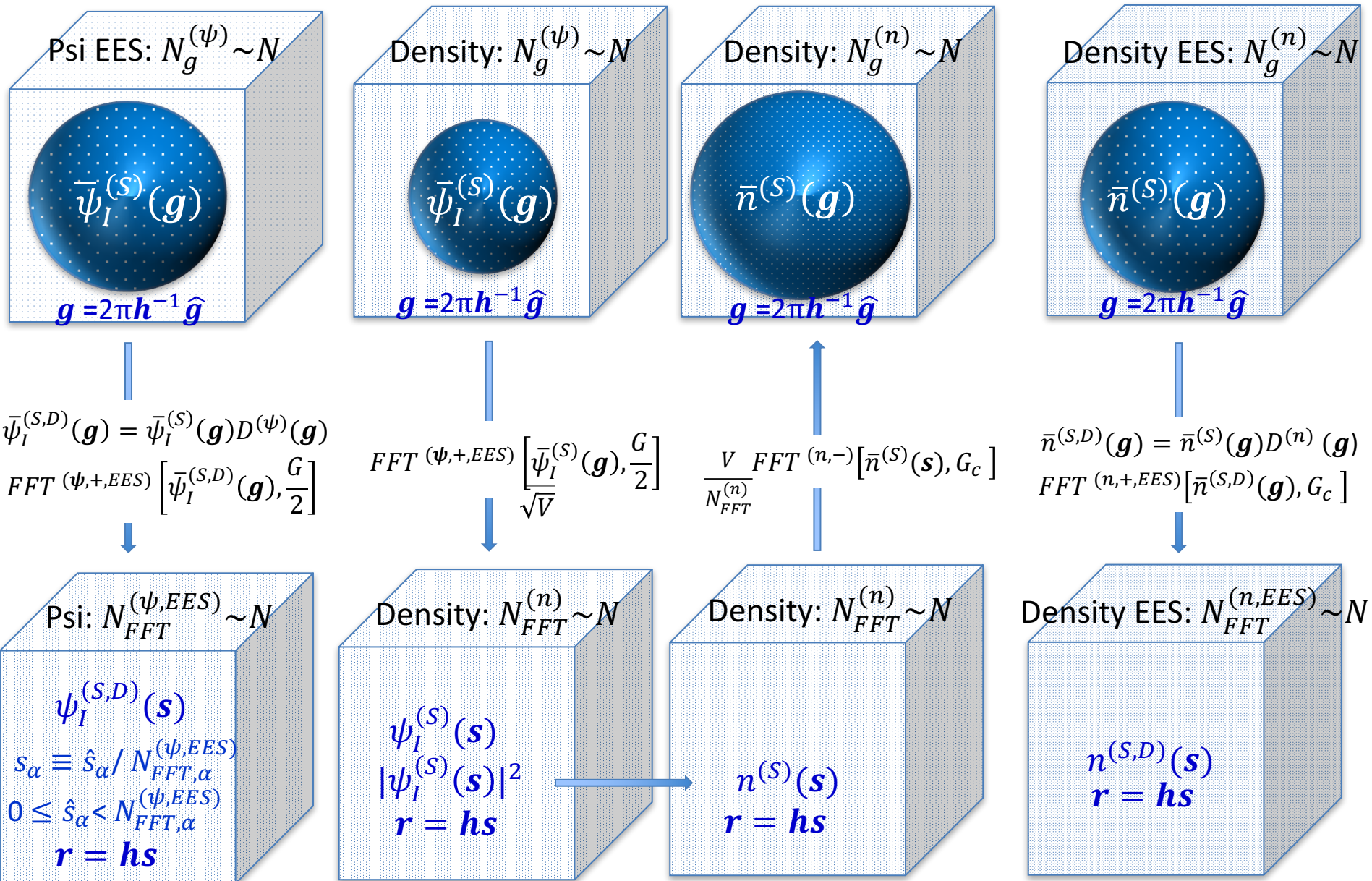
2. Euler Exponential Spline Interpolation and FFTs

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

$$e^{2\pi i \hat{\mathbf{g}} \cdot \mathbf{s}} = D_p(\hat{\mathbf{g}}, N_{\text{FFT}}) \sum_{\hat{s}=0}^{N_{\text{FFT}}} \sum_{j=1}^p M_p(u - \hat{s}) e^{\frac{2\pi i \hat{\mathbf{g}} \hat{s}}{N_{\text{FFT}}}} \delta_{\hat{s}, l-j} + \mathcal{O}\left(\frac{2\hat{g}}{N_{\text{FFT}}}\right)^p, \quad \begin{array}{l} M_p \text{ has compact supp.} \\ u = s N_{\text{FFT}} \quad l = \text{int } u \\ N_{\text{FFT}} > 2\hat{g}_{\text{max}} \approx 2.8\hat{g}_{\text{max}} \end{array}$$

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

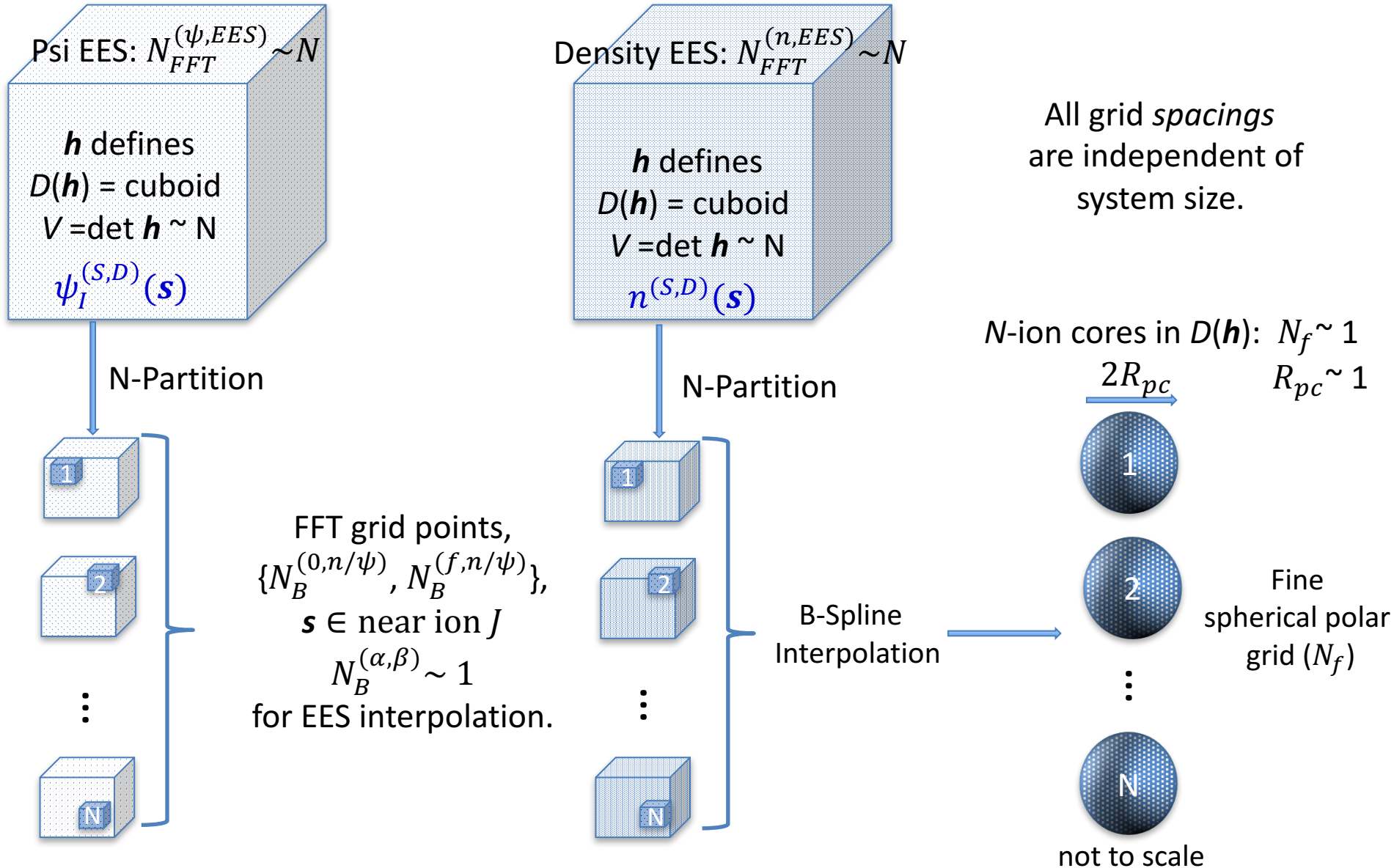
PAW Basics: g -space to s -space and back



The $D^{(\tau)}(\mathbf{g}) = \prod_{\alpha} D_p \left(\hat{g}_{\alpha}, N_{FFT, \alpha}^{(\tau, EES)} \right)$ enables B-spline interpolation

PAW Basics: r -space interpolation

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



Creating the \mathbf{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:

$$n(\mathbf{r}) = n^{(S)}(\mathbf{r}) + \sum_J \left[n_J^{(\text{core } 1)}(\mathbf{r}_f) + n_J^{(\text{core } 2)}(\mathbf{r}_f) \right], \quad n_J^{(S)}(\mathbf{r}_f)$$

- (1) Create the smooth KS states in real space, $\psi_I^{(S)}(\mathbf{s}): N^2 \log N$.
- (2) Create the smooth density in real space, $n^{(S)}(\mathbf{s}): N^2$.
- (3) *Create the smooth density in the ion cores, $n_J^{(S)}(\mathbf{r}_f): N \log N$.
- (4) Create the smooth Z-matrix, $Z_{IJ}^{(S)} : N^2 \log N$.
- (5) *Create the core-2 densities, $n_J^{(\text{core } 2)}(\mathbf{r}_f): N^2$.
- (6) *Create the core-1 densities, $n_J^{(\text{core } 1)}(\mathbf{r}_f): N^2 \log N$.

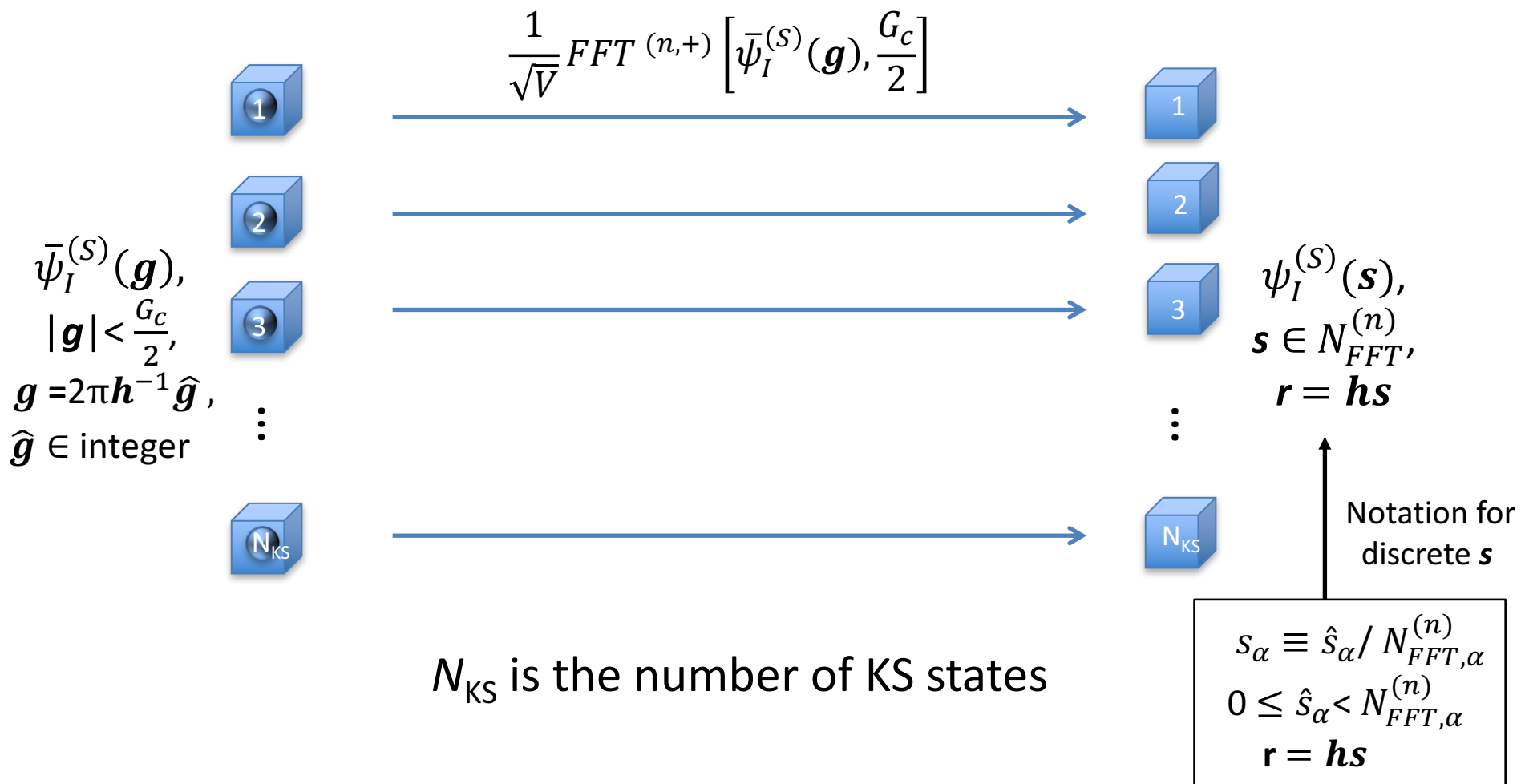
* New terms.

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, $\psi_I^{(S)}(\mathbf{s})$, on the density \mathbf{s} -space FFT grid, $\mathbf{s} \in N_{FFT}^{(n)}$

Smooth part
of the KS states
in \mathbf{g} -space

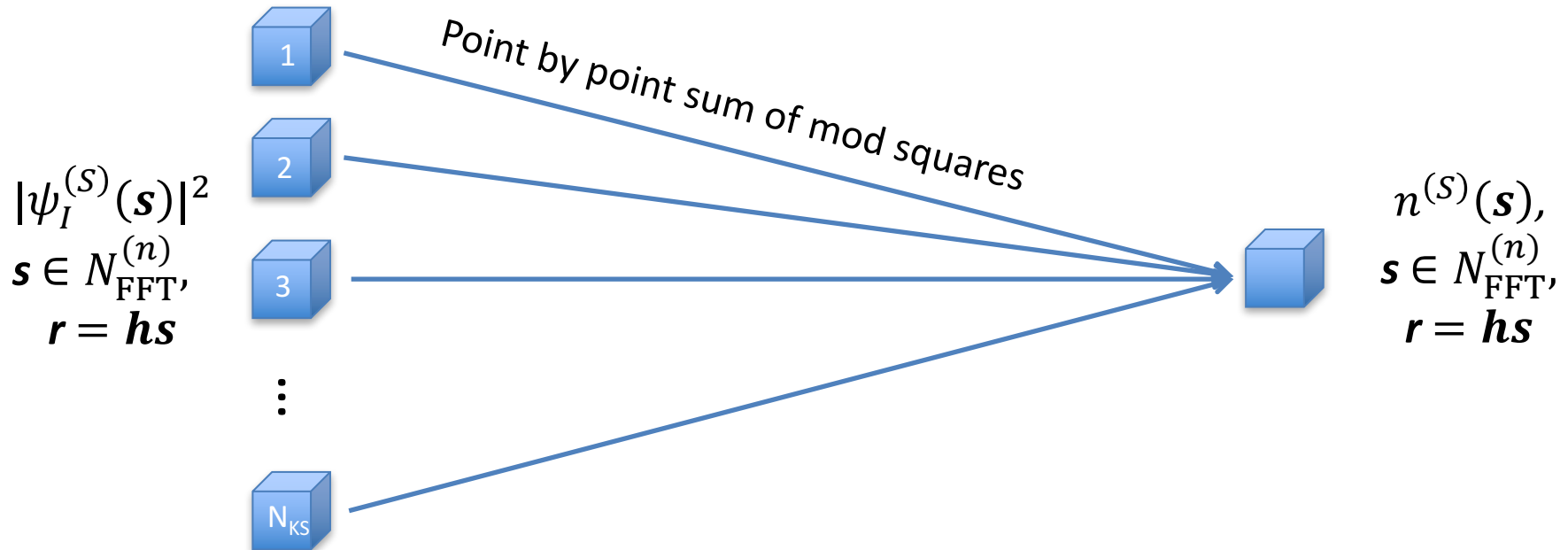
Smooth part
of the KS states
on discrete \mathbf{s} -space



2. Creating the smooth density, $n^{(s)}(\mathbf{s})$ on the density \mathbf{s} -space FFT grid, $\mathbf{s} \in N_{\text{FFT}}^{(n)}$

Smooth part
of the KS states
on discrete \mathbf{s} -space

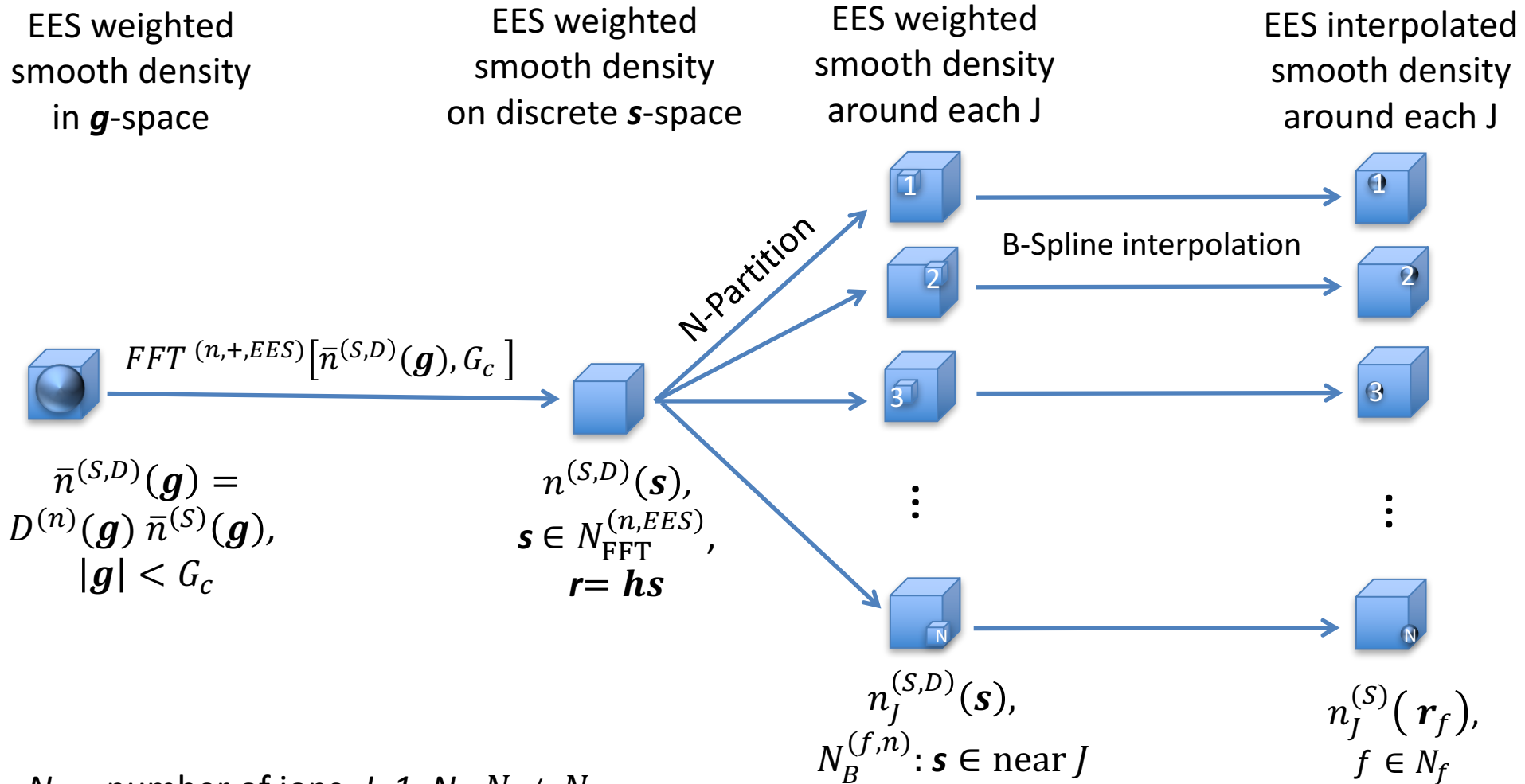
Smooth part
of the density
on discrete \mathbf{s} -space



Point by point sum of mod squares:

$$n^{(s)}(\mathbf{s}) = \sum_I |\psi_I^{(s)}(\mathbf{s})|^2 \quad \forall \mathbf{s} \in N_{\text{FFT}}^{(n)}$$

3. Creating the smooth density, $n_J^{(S)}(\mathbf{r}_f)$ around each ion J , on the fine grid, $f \in N_f$

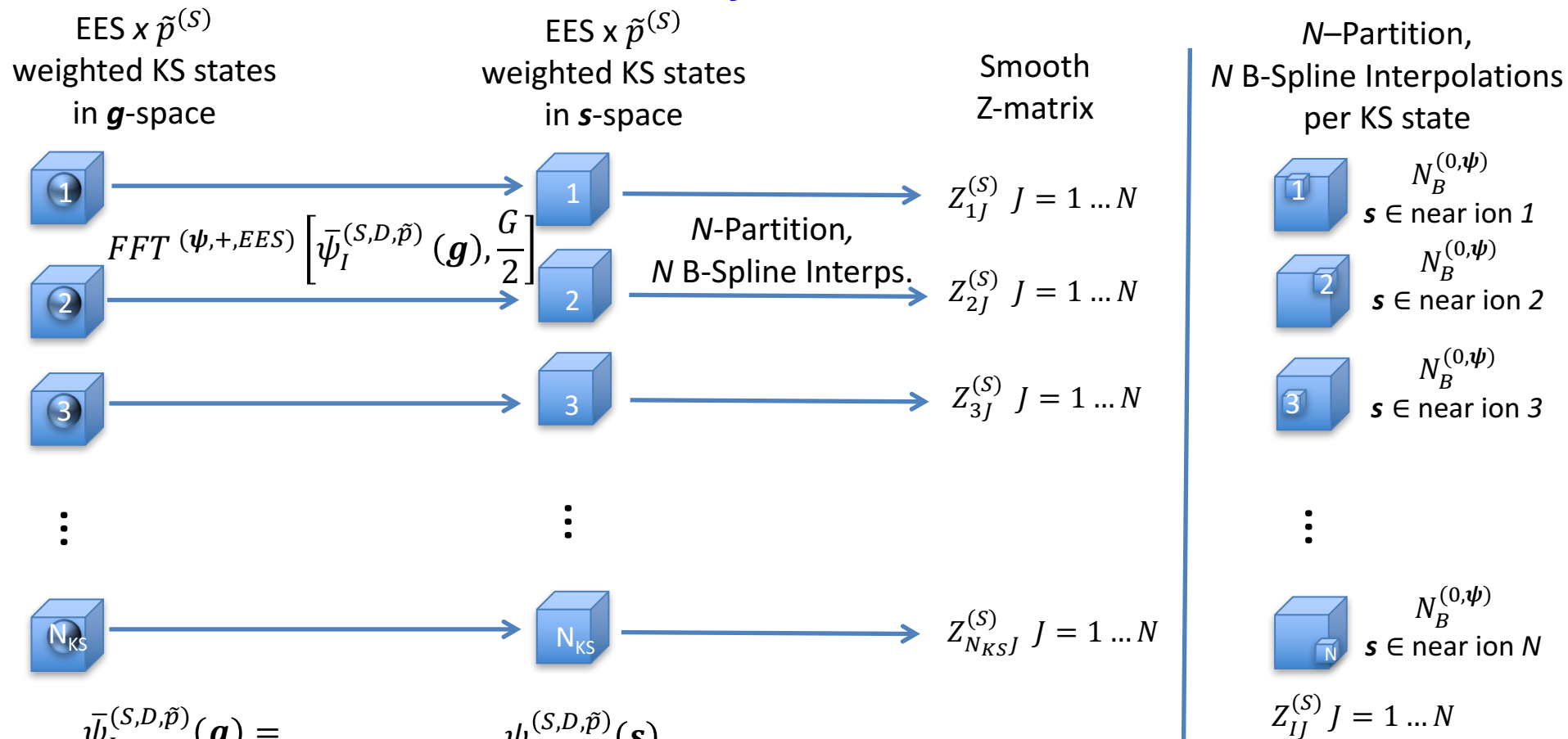


N = number of ions, $J=1..N$, $N \neq N_{KS}$

N_f = number points on spherical-polar grid around each ion.

N_f and $N_B^{(f,n)}(\mathbf{s} \in \text{near } J)$ independent system size .

4. Creating the $Z_{IJ}^{(S)}$ the matrix elements



$$\bar{\psi}_I^{(S,D,\tilde{p})}(\mathbf{g}) = D^{(\psi)}(\mathbf{g}) \bar{\psi}_I^{(S)}(\mathbf{g}) \tilde{p}^{(S)}(\mathbf{g}), \quad |\mathbf{g}| < G_c/2$$

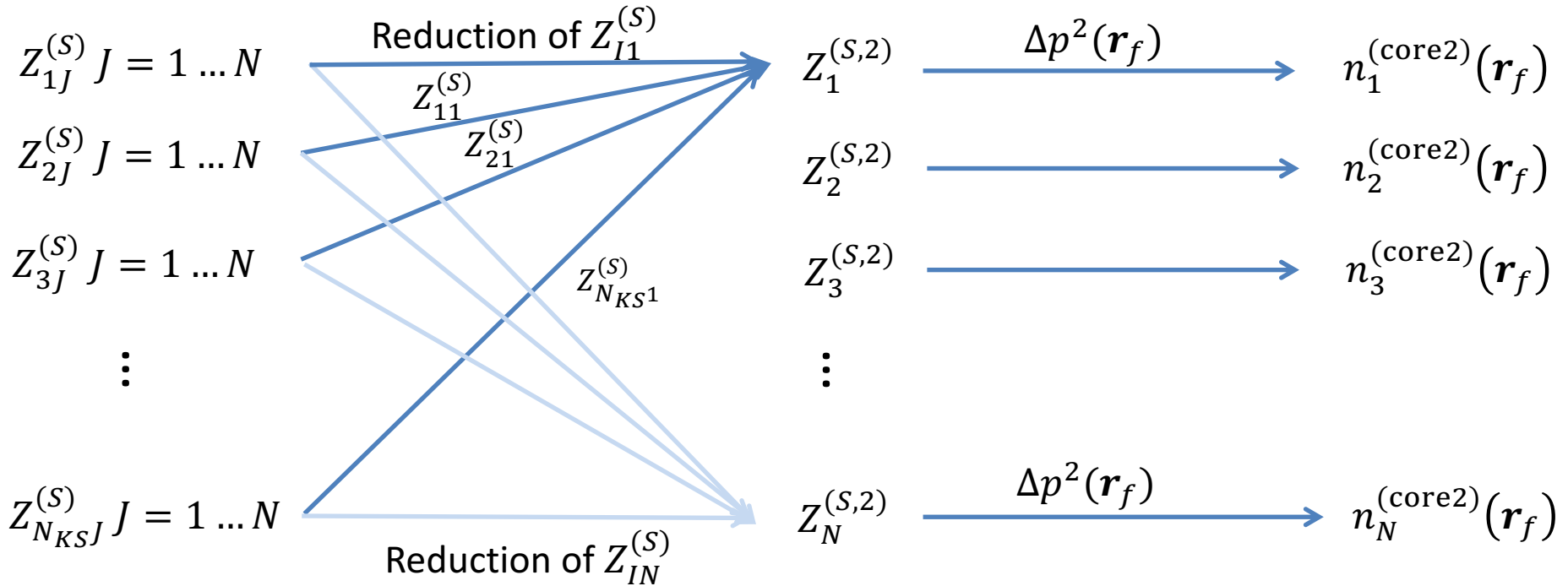
$$\psi_I^{(S,D,\tilde{p})}(\mathbf{s}), \quad \mathbf{s} \in N_{FFT}^{(\psi, EES)}$$

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

$$Z_{IJ}^{(S)} = \sum_{\mathbf{s} \in \text{near } J}^{N_B^{(0,\psi)}} \psi_I^{(S,D,\tilde{p})}(\mathbf{s}) M_{J,p}^{(3)}(\mathbf{s})$$

$\mathbf{s} \in \text{near ion } J$ independent of I as are B-splines, $M_{J,p}^{(3)}(\mathbf{s})$

5. Creating the core density component, $n_J^{(\text{core2})}(\mathbf{r}_f)$, 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

$$n_J^{(\text{core2})}(\mathbf{r}_f) = Z_J^{(S,2)} \Delta p^2(\mathbf{r}_f) \quad \forall f \in N_f$$

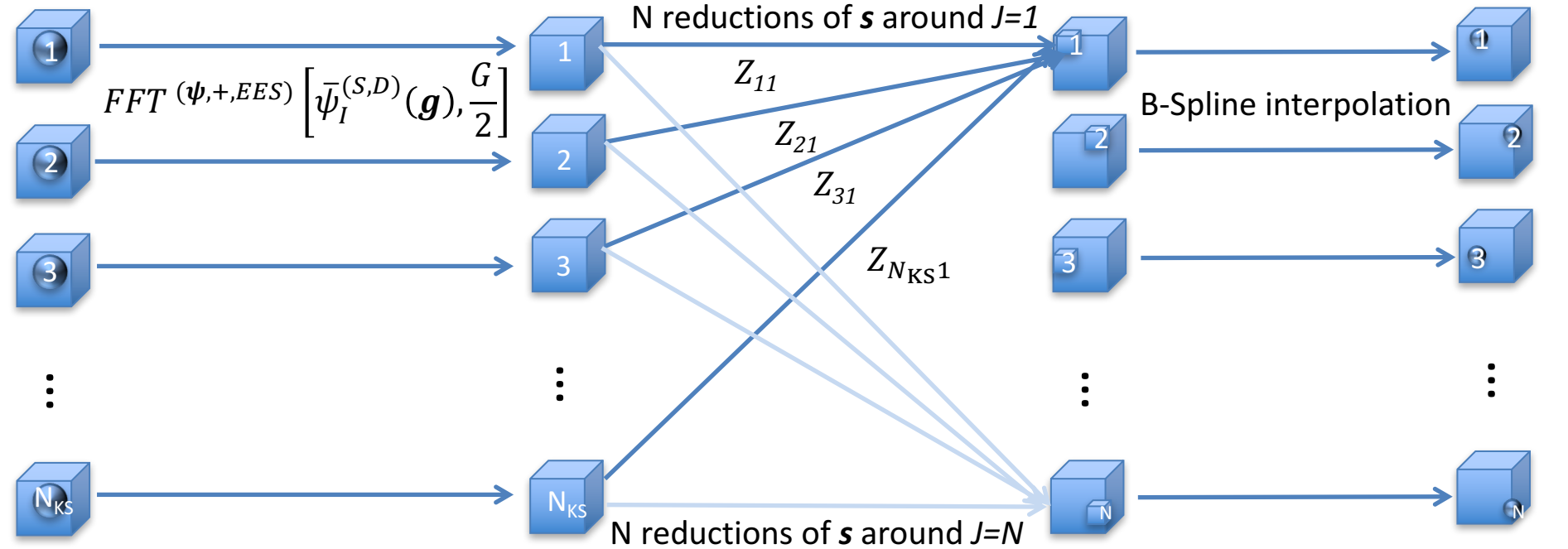
6. Creating the core density component, $n_J^{(\text{core1})}(\mathbf{r}_f)$, around each ion J , on the fine grid, $f \in N_f$

EES weighted
KS states
in \mathbf{g} -space

EES weighted
KS states
in \mathbf{s} -space

EES weighted
KS states
around J

EES interpolated
PAW 1 density
around each J



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

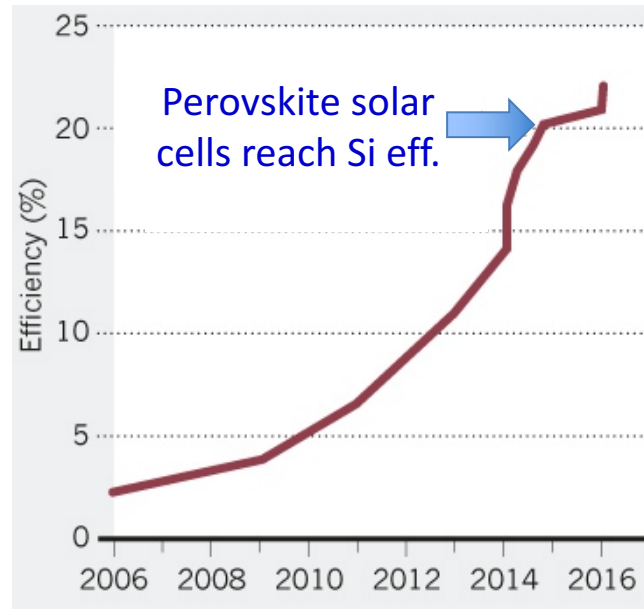
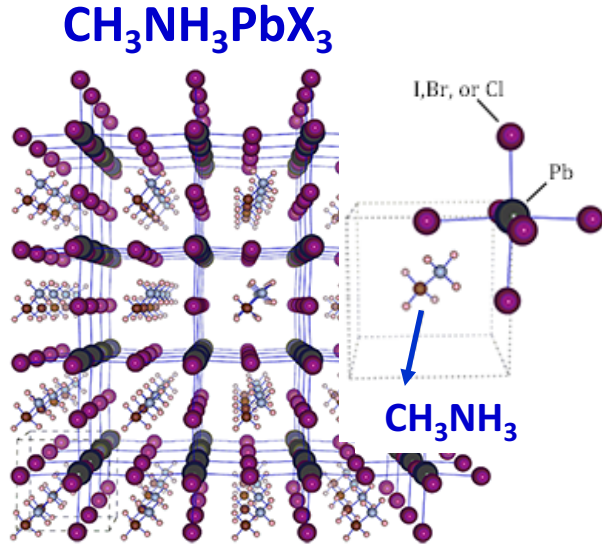
$$\psi_I^{(S,D)}(\mathbf{s}), \quad \mathbf{s} \in N_{FFT}^{(\psi, EES)}$$

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

$$n_J^{(\text{core1})}(\mathbf{r}_f), \quad f \in N_f$$

Each KS state contributes to N unique reductions
 $\psi_J^{(S,D,Z)}(\mathbf{s}) = \sum_I Z_{IJ} \psi_I^{(S,D)}(\mathbf{s}) \quad \forall \mathbf{s} \in \text{near } J: N_B^{(f, \psi)}$
 Z_{IJ} = weight for points $\mathbf{s} \in \text{near } J$ from KS state, I .

Grand Challenge Application: Perovskite solar cells



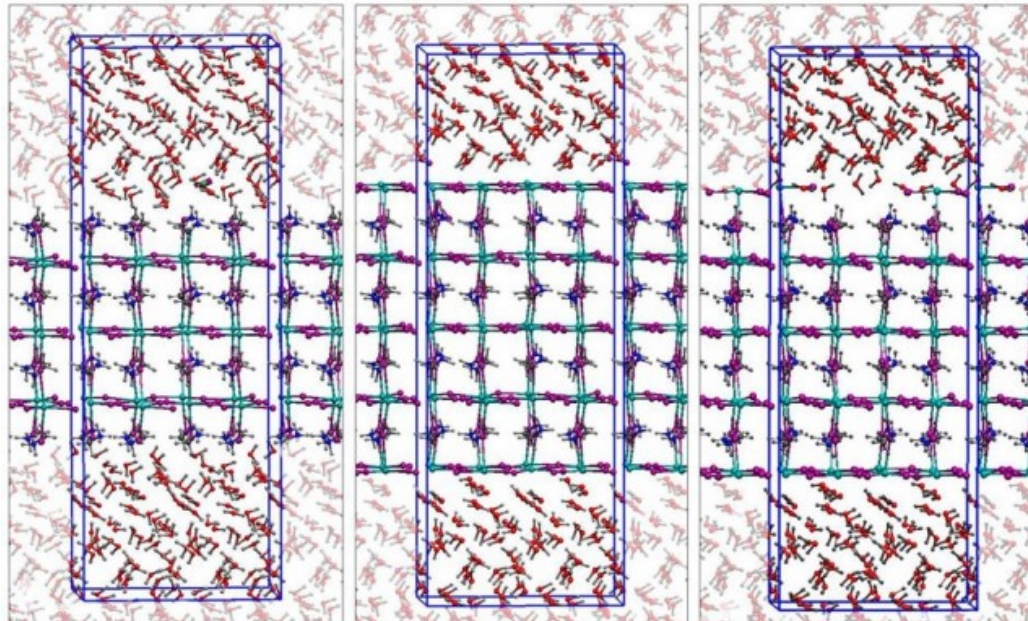
- **Pros:**
High eff., low cost, tunable band gap (ABX_3)
- **Cons:**
Instability: water, air, light, interface ... & toxic compounds.

PAW in
OpenAtom

MAI-term.

PbI_2 -term.

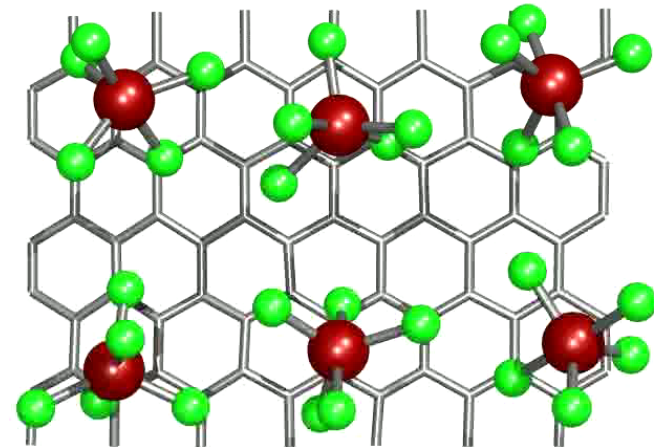
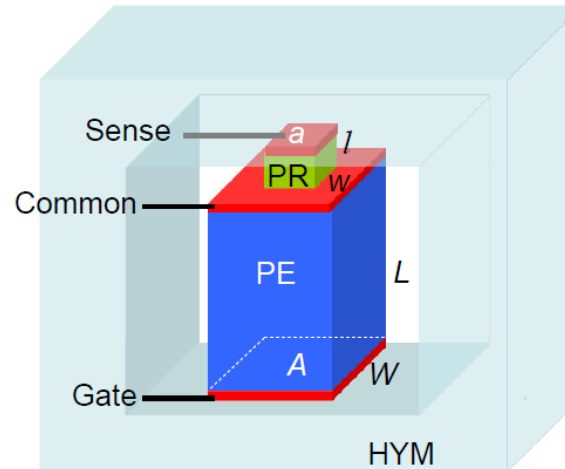
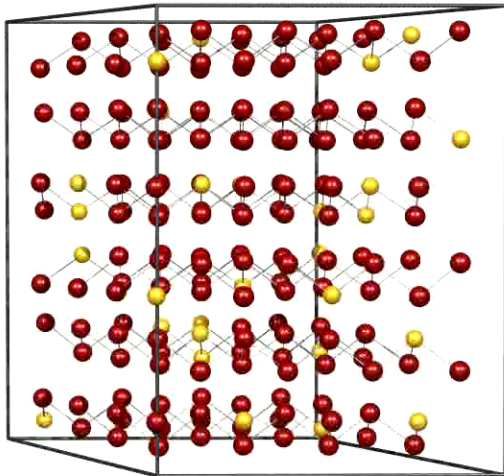
PbI_2 -defect.



- **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 ($4 \times 4 \times 2$ $\text{MAPbI}_3 + 128$ 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 \mathbf{g} -space representation of the e-density

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

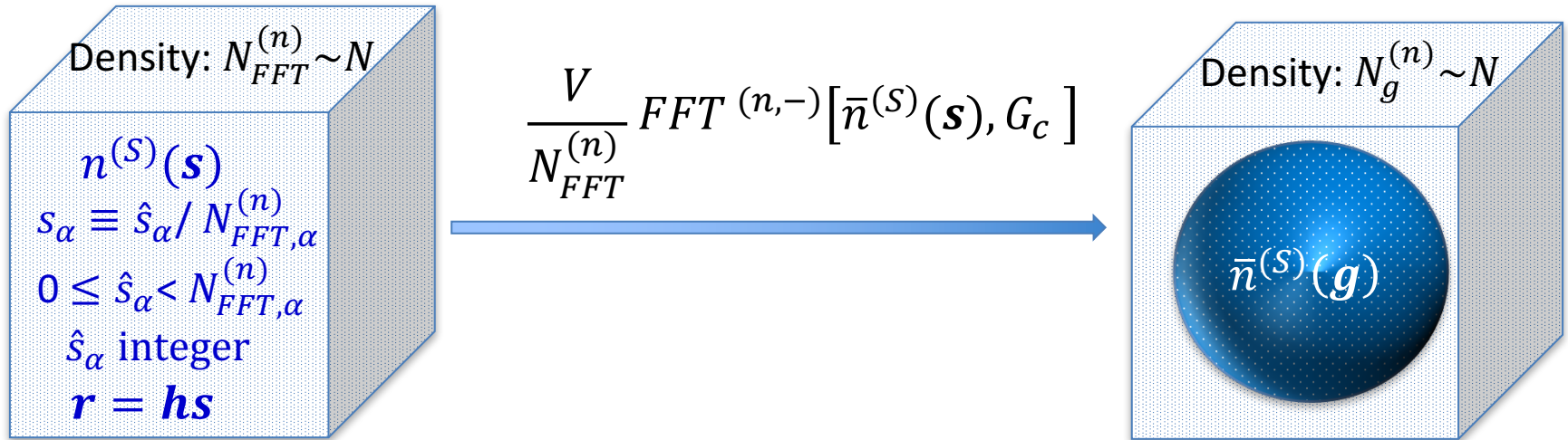
- $\bar{n}^{(S)}(\mathbf{g})$: Sampling theorem from $n^{(S)}(\mathbf{s})$
 $\bar{n}^{(\text{core 1})}(\mathbf{g})$: Numerical integration over core 1 density + EES
 $\bar{n}^{(\text{core 2})}(\mathbf{g})$: Bessel transform (precompute) + EES

- (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*

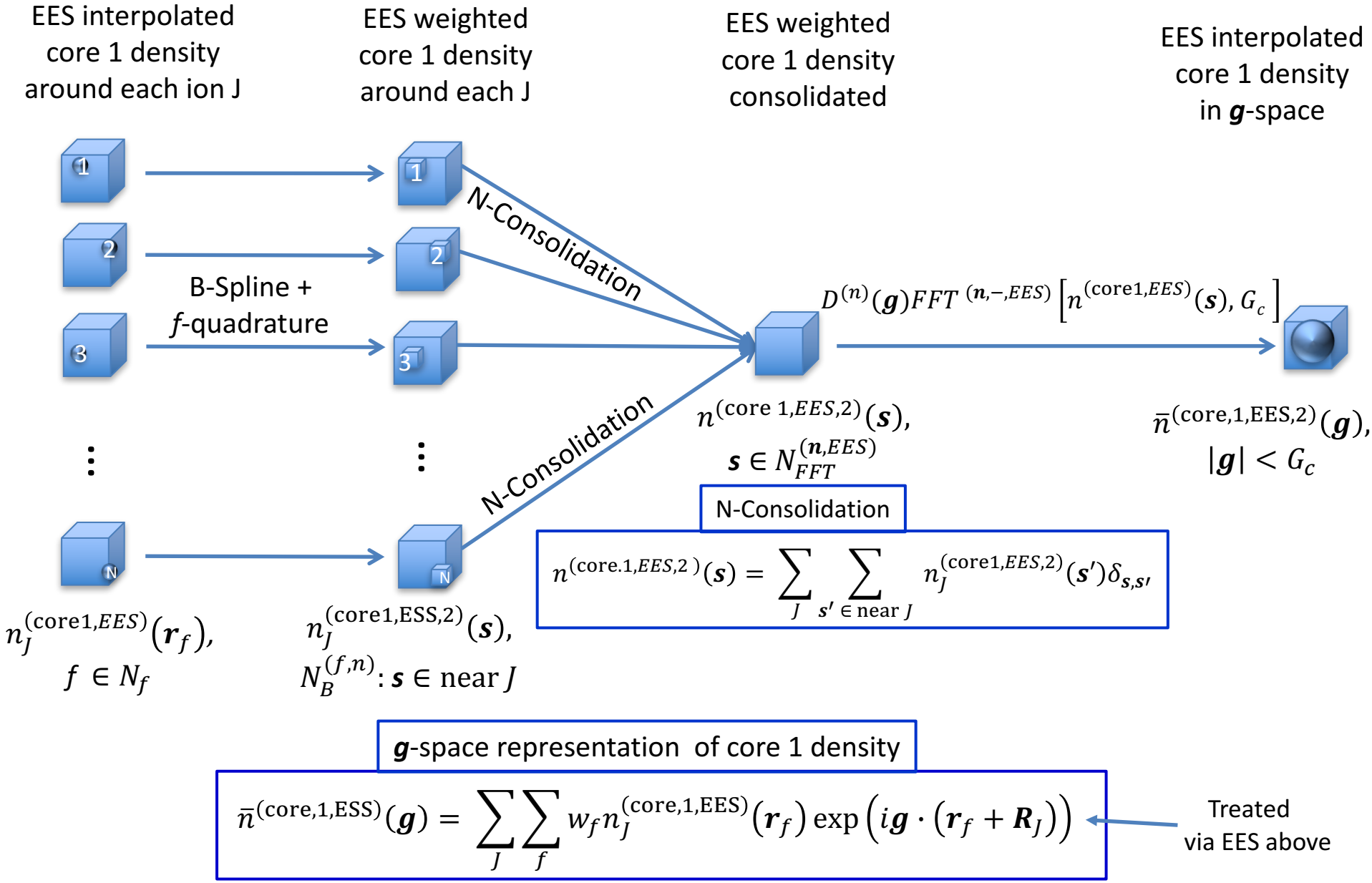
- $\bar{n}^{(S)}(\mathbf{g})$: $N \log N$.
 $\bar{n}^{(\text{core 1})}(\mathbf{g})$: $N \log N$
 $\bar{n}^{(\text{core 2})}(\mathbf{g})$: $N \log N$

1. Creating the \mathbf{g} -space representation of smooth density

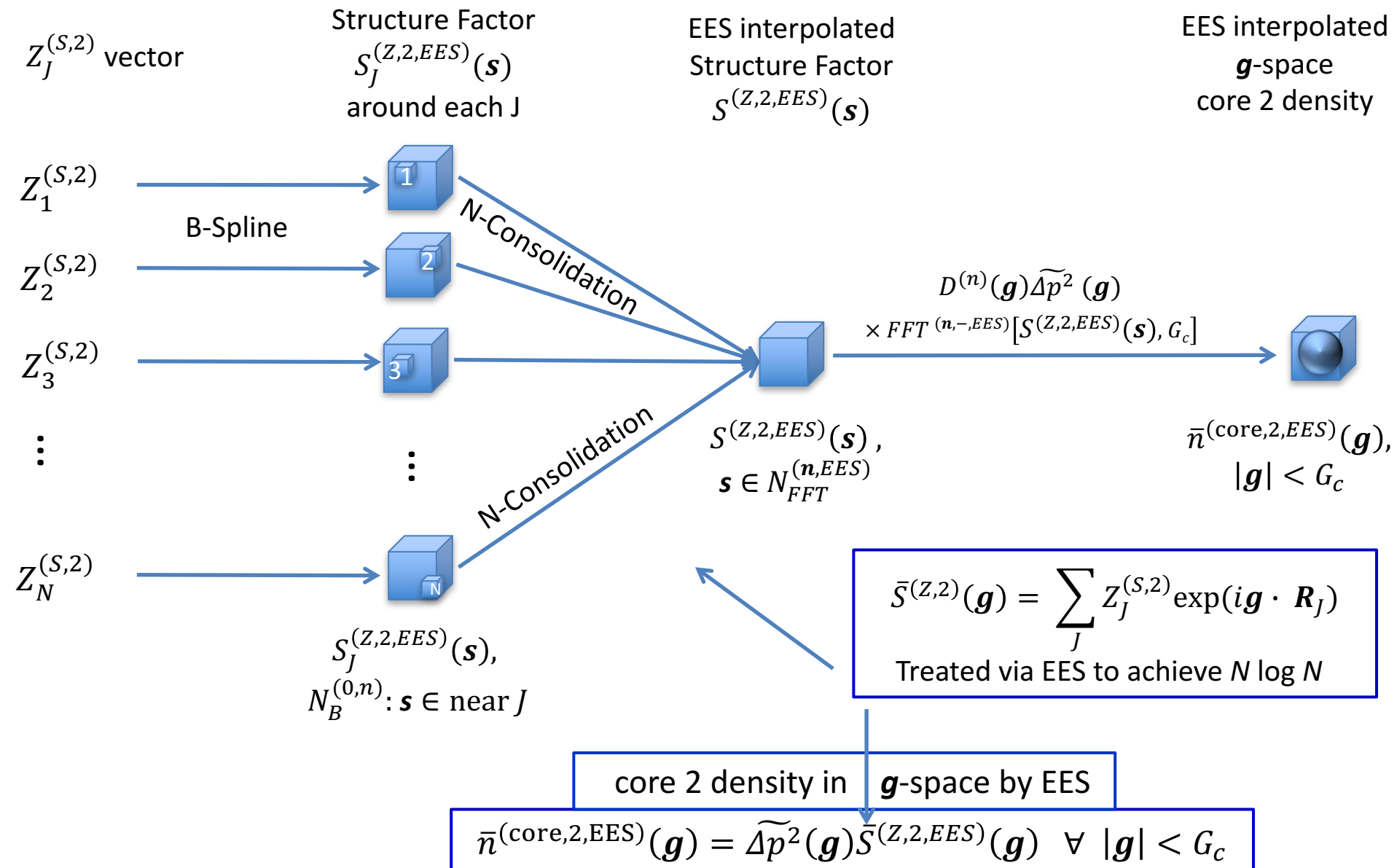


Density Fourier coefficients $\bar{n}^{(s)}(\mathbf{g})$, $|\mathbf{g}| < G_c$ are exact, through intermediate discrete \mathbf{s} -space – Theorem 1.
 $N \log N$ method given $n^{(s)}(\mathbf{s})$.

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



3. Creating the \mathbf{g} -space core 2 density, $\bar{n}^{(\text{core } 2, \text{EES})}(\mathbf{g})$



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

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

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

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

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

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

*new terms

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

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

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

$$\bar{\psi}_I^{(\nabla^2 S, D, \Delta p)}(\mathbf{g}) = D^{(\psi)}(\mathbf{g}) g^2 \bar{\psi}_I^{(S)}(\mathbf{g}) \widetilde{\Delta p}(\mathbf{g})$$

Accuracy of long/short decomposition

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

	$R_c = 4$ bohr	
PW cutoff: ($\hbar^2 G_c^2 / 8me$) Ryd	$\gamma = \alpha R_c$	erfc(γ)
5.1	3.0	2.21e-05
9.4	3.5	7.43e-07
16	4.0	1.54e-08

	$R_c = 2$ bohr	
PW cutoff: ($\hbar^2 G_c^2 / 8me$) Ryd	$\gamma = \alpha R_c$	erfc(γ)
20.3	3.0	2.21e-05
37.5	3.5	7.43e-07
64.0	4.0	1.54e-08

High accuracy can be obtained with both R_c and G_c small !

Eric Bohm



UNIVERSITY OF ILLINOIS
AT URBANA-CHAMPAIGN

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

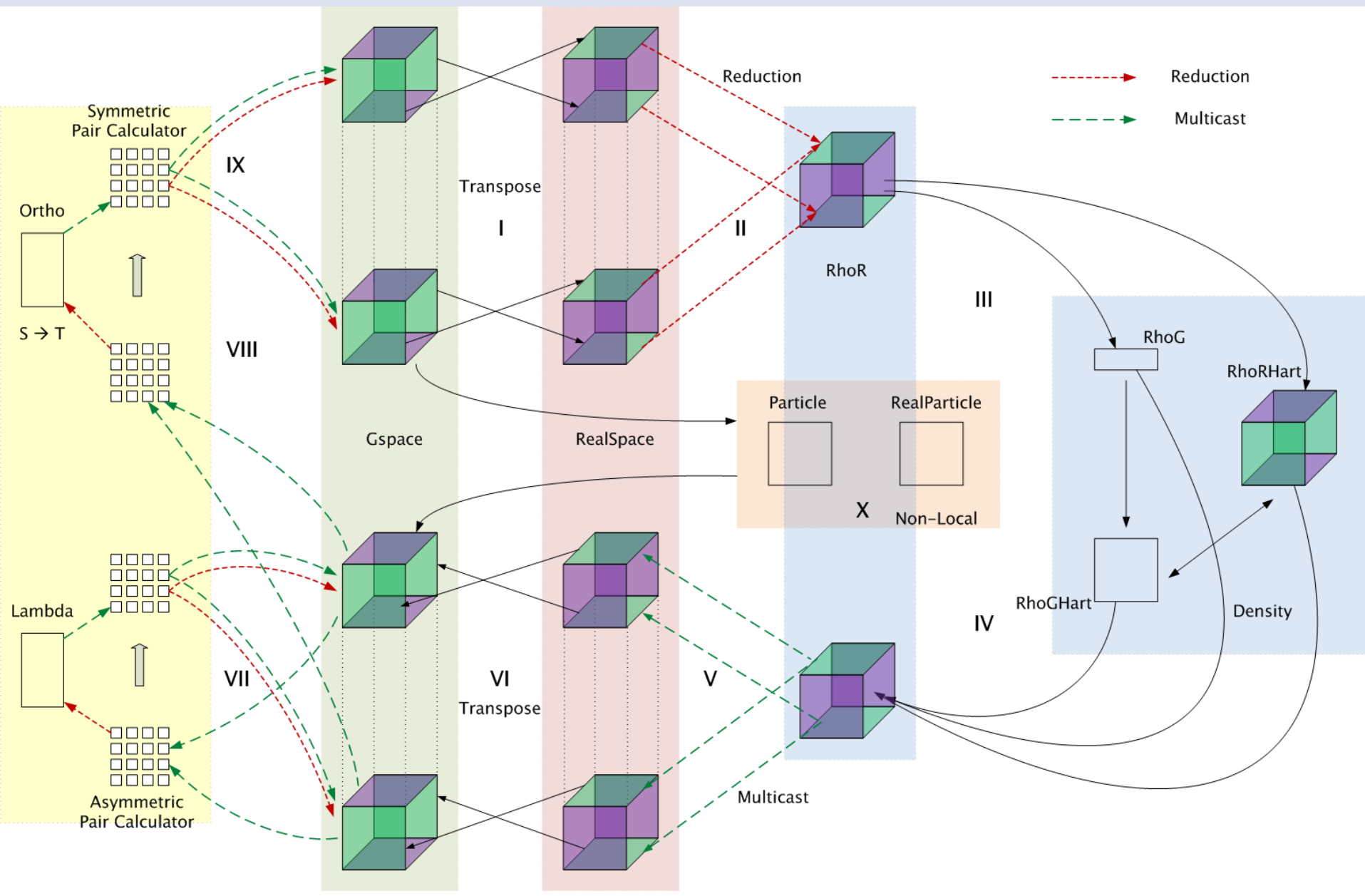


Ground State Feature Status

Feature	Minimization Status	Dynamics Status	Test Integration
CPAIMD Dynamics	NA	Production	Automated 
Path Integrals	Production	Production	Automated 
K-Points	Production	Needs Verification	Automated 
Spin Orbitals	Production	Production	Automated 
Tempering	NA	Production	Automated 
Born Oppenheimer Dynamics	NA	Production	Automated 
Band Generation	Being Evaluated	Being Evaluated	Manual 



Control flow in OpenAtom (PW-DFT)

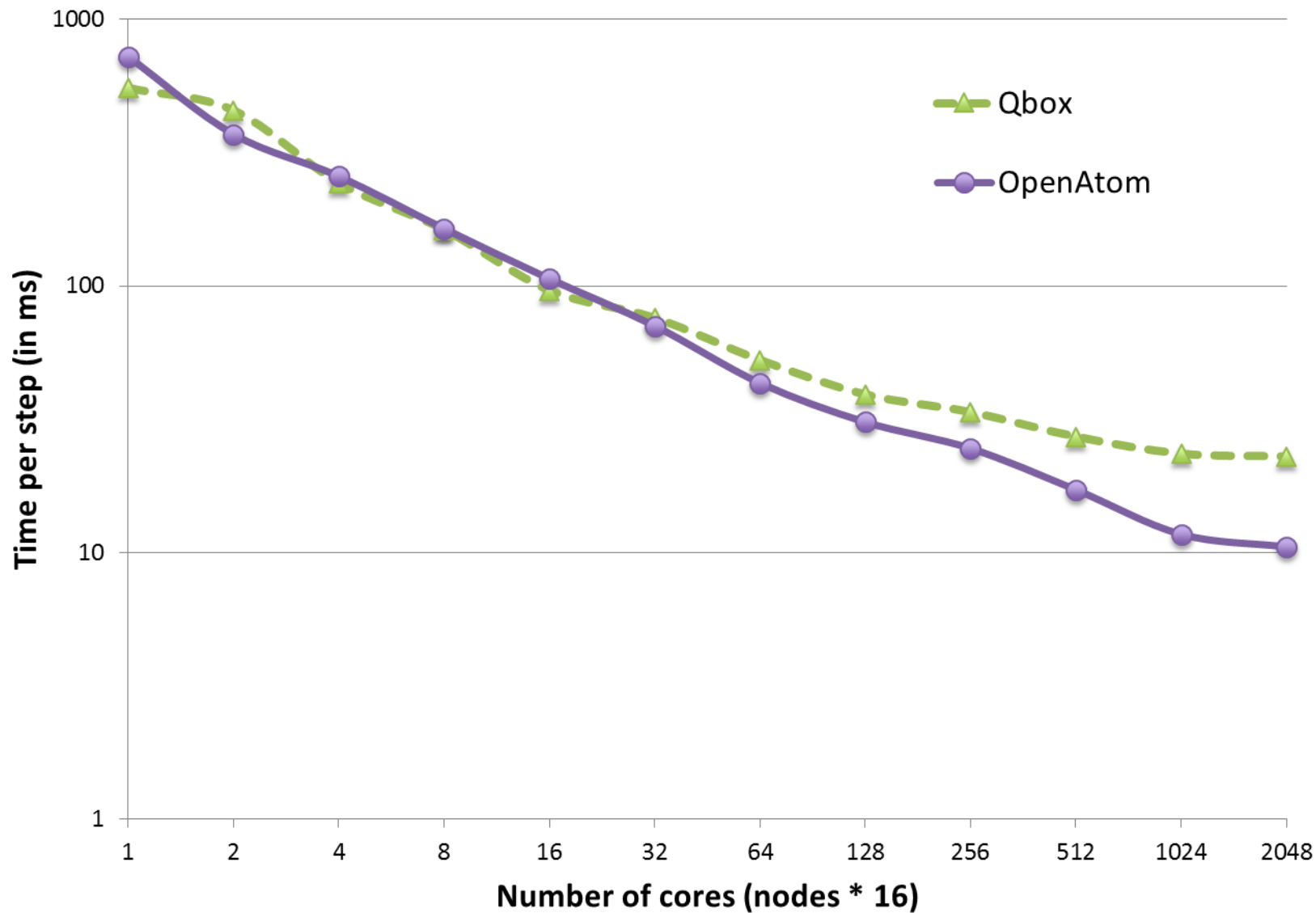


Nikhil Jain

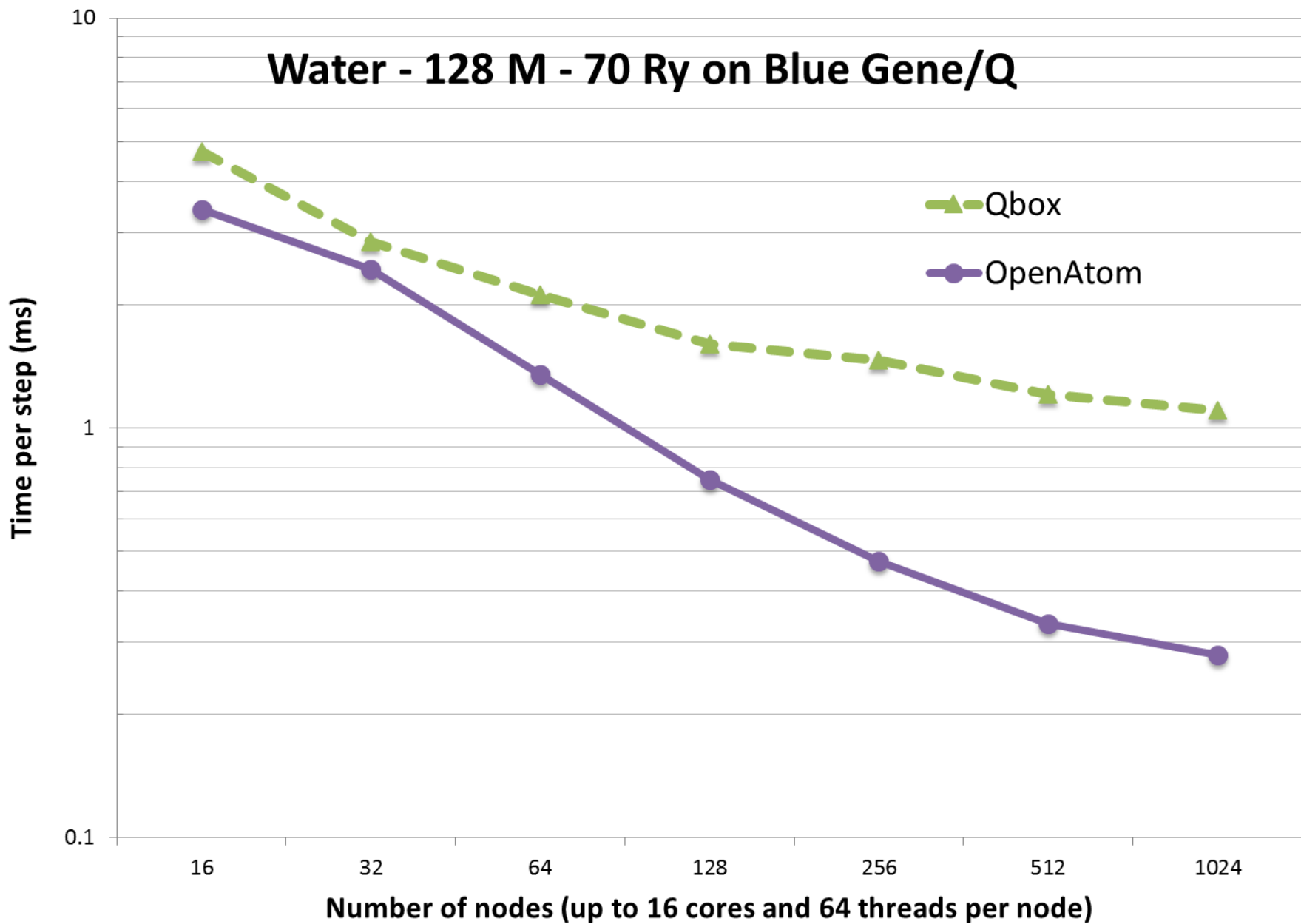
QBOX COMPARISON



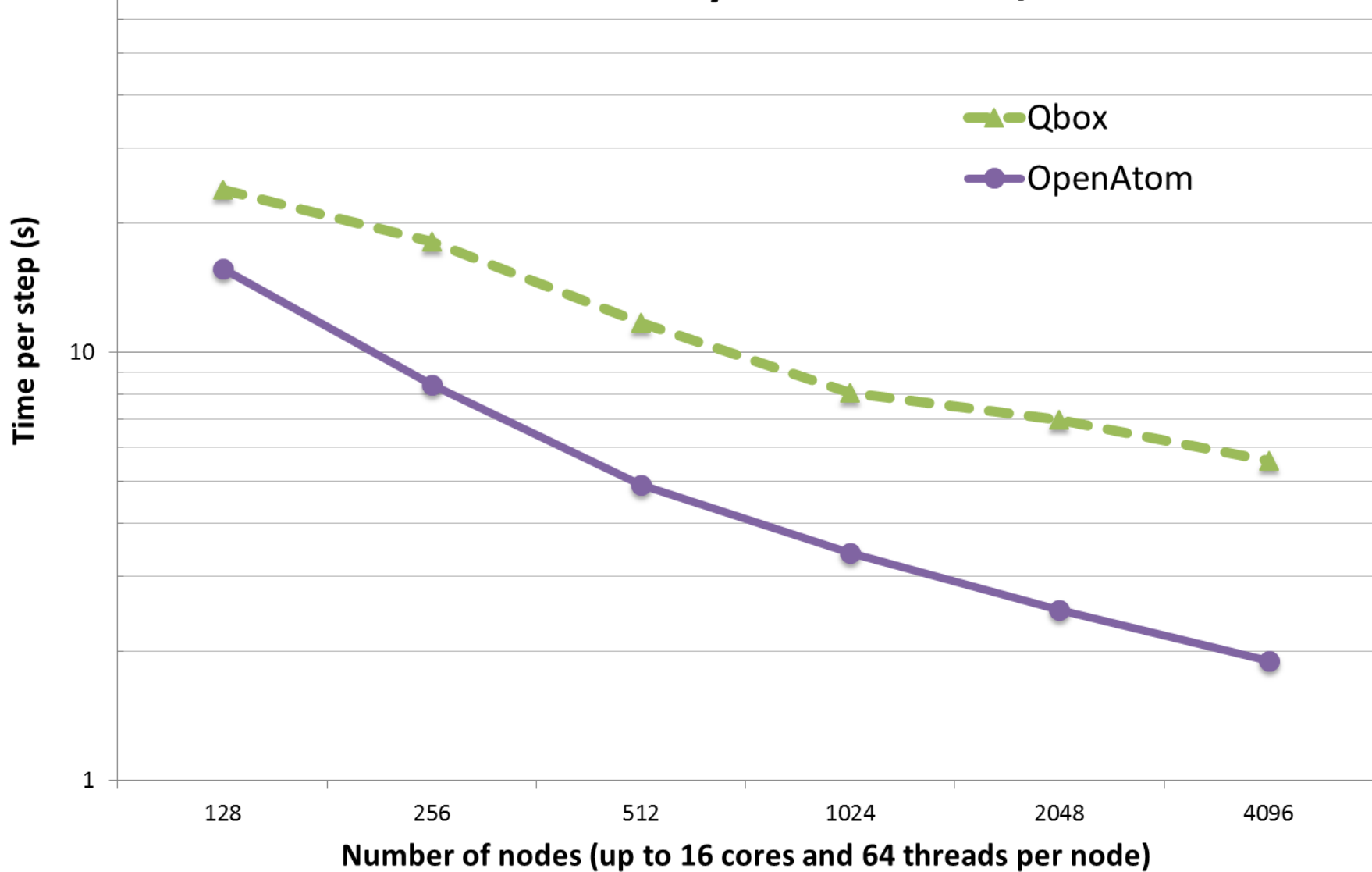
Performance comparison for Water-32M-10Ry



Water - 128 M - 70 Ry on Blue Gene/Q



Water - 512 M - 70 Ry on Blue Gene/Q

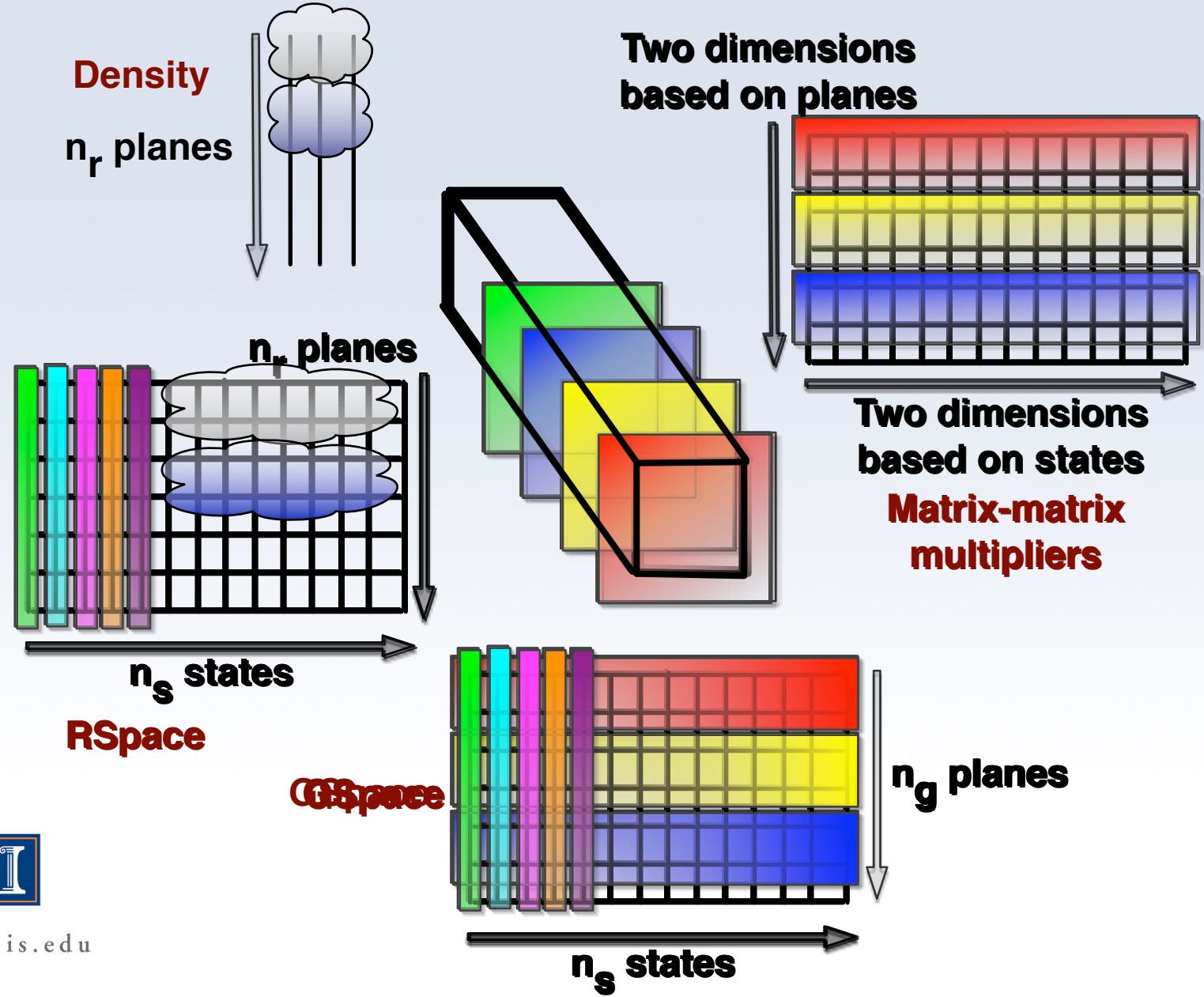


Nikhil Jain

OBJECT PLACEMENT



Topology aware mapping



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

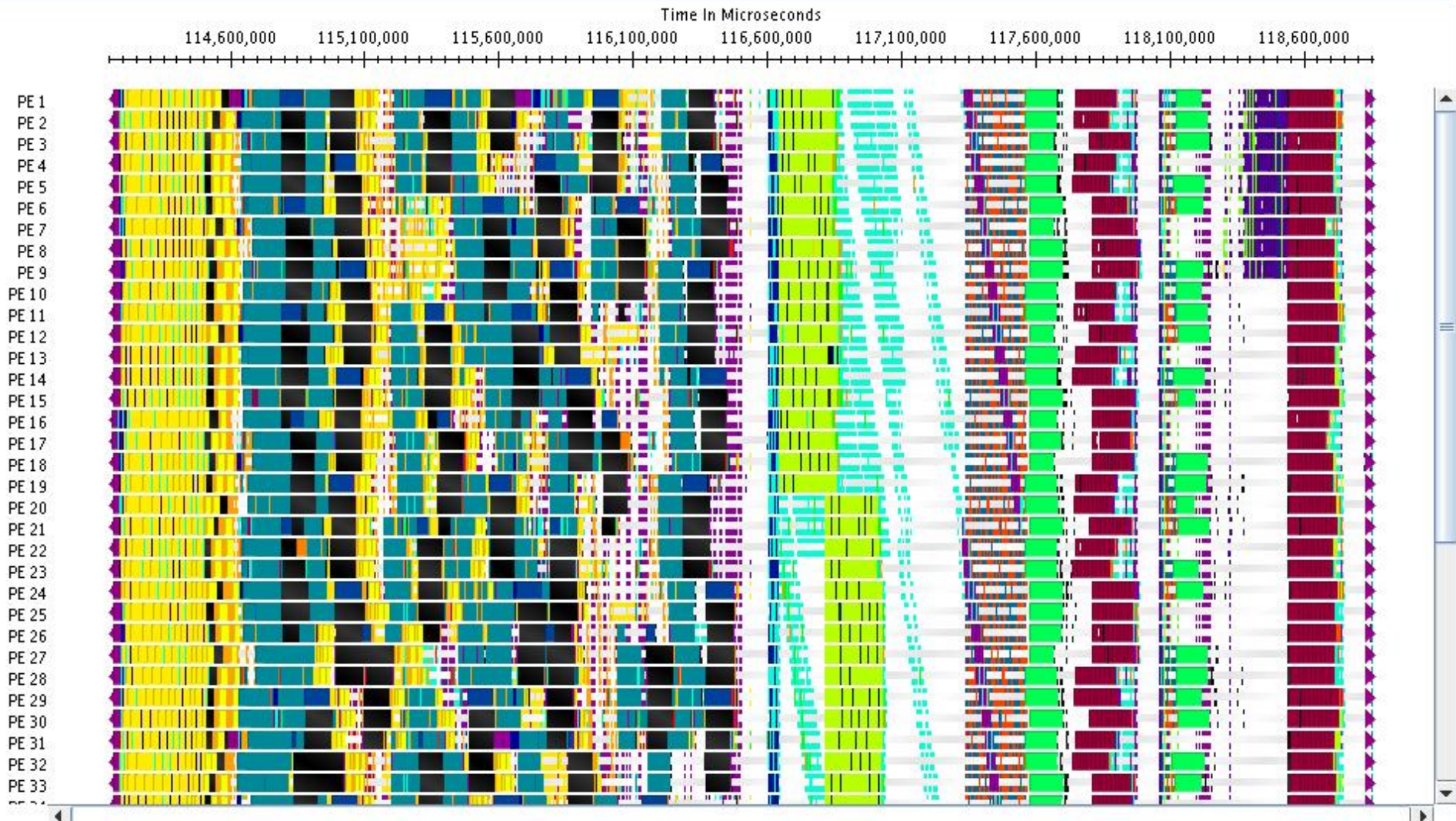
(barriers introduced for clarity)

States: Many
G -> R FFTs

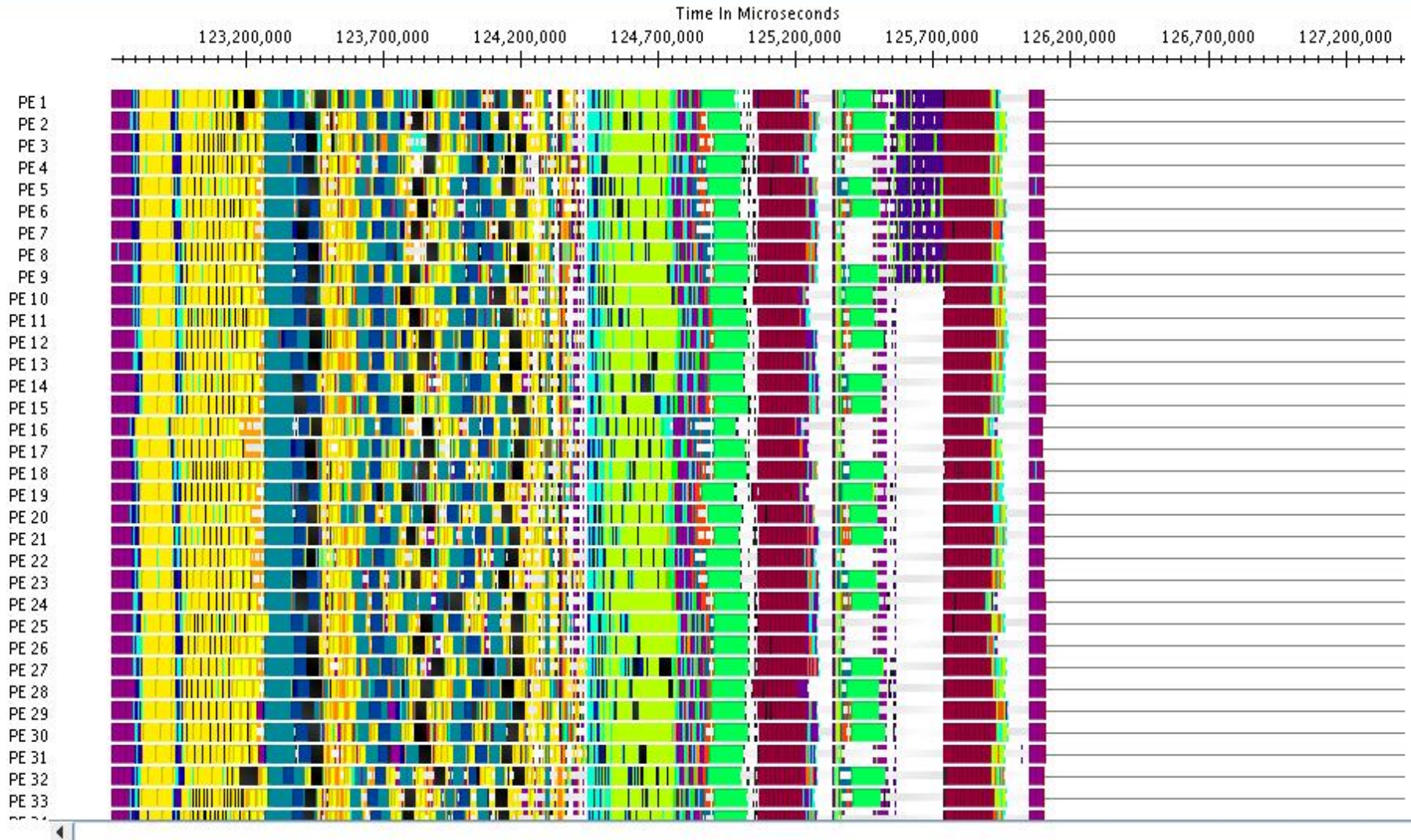
Density: G -> R -> G
Non-local G-> R -> G

Density to States
States R-> G

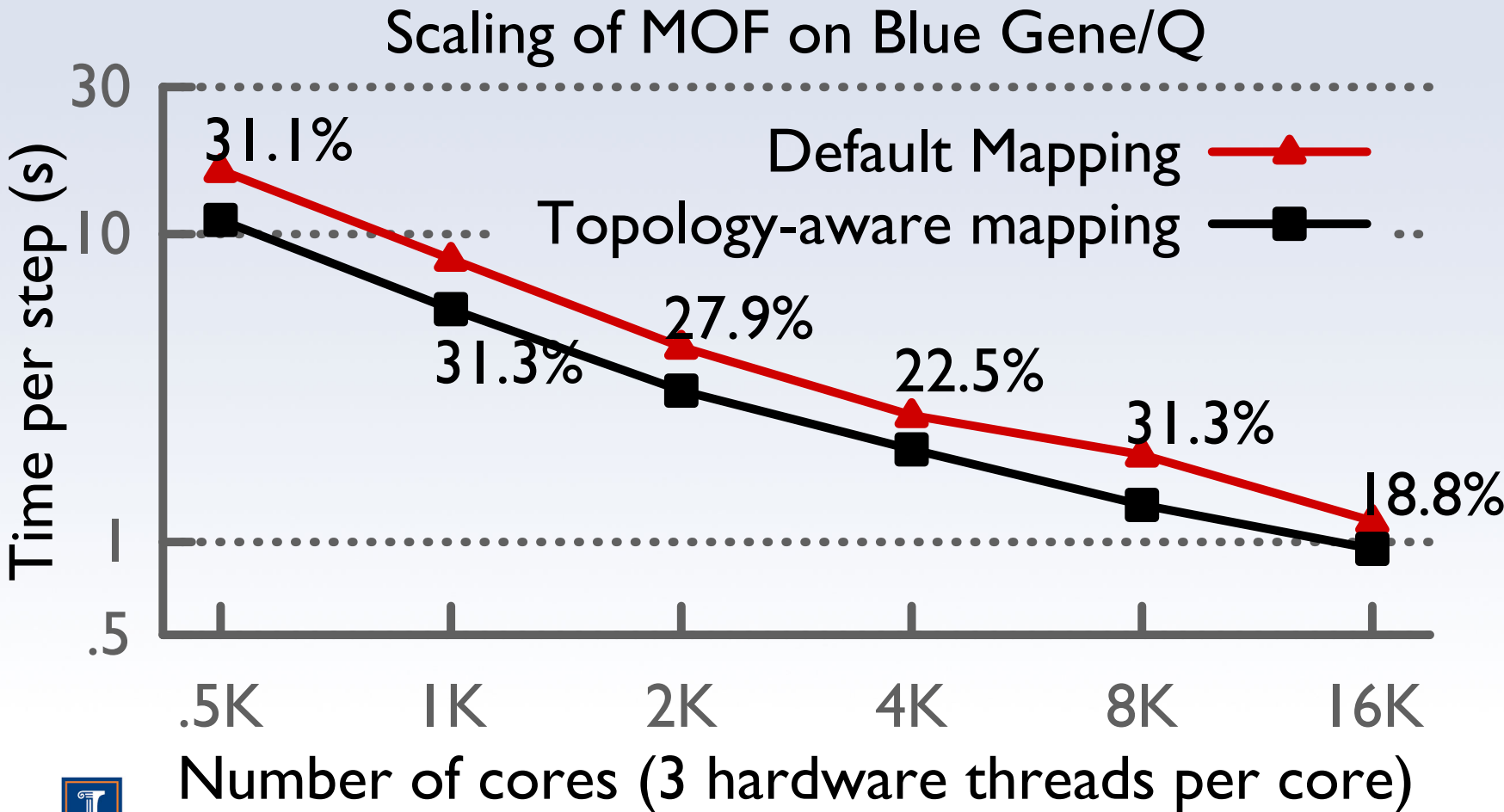
Force correction
Ortho-normalization



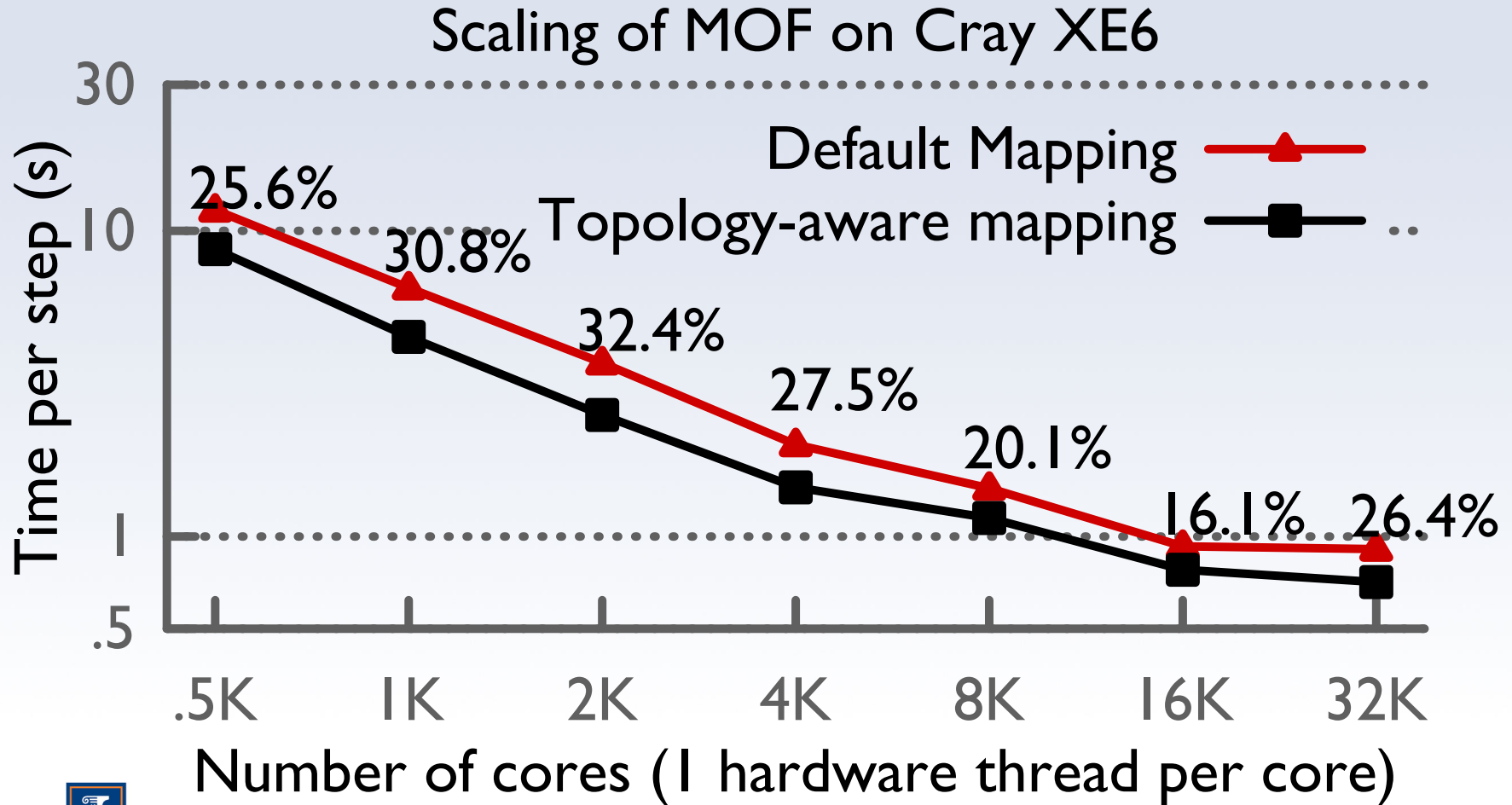
System utilization with mapping



Impact of mapping on Blue Gene/Q: up to 30% improvement



Impact of mapping on Blue Waters: up to 32% improvement



Eric Bohm, Glenn Martyna

UBERS : MULTI-INSTANCE METHODS



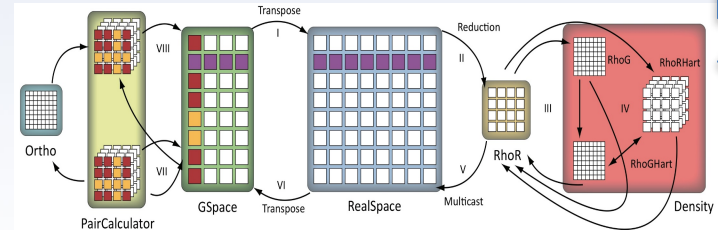
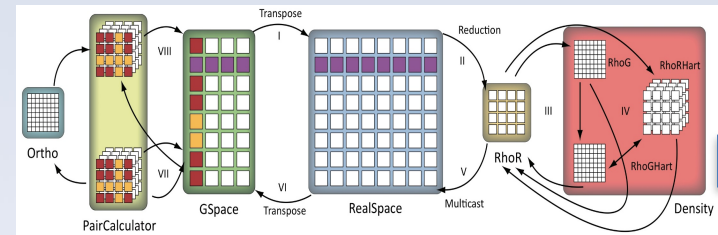
Multi Instance Methods

- 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



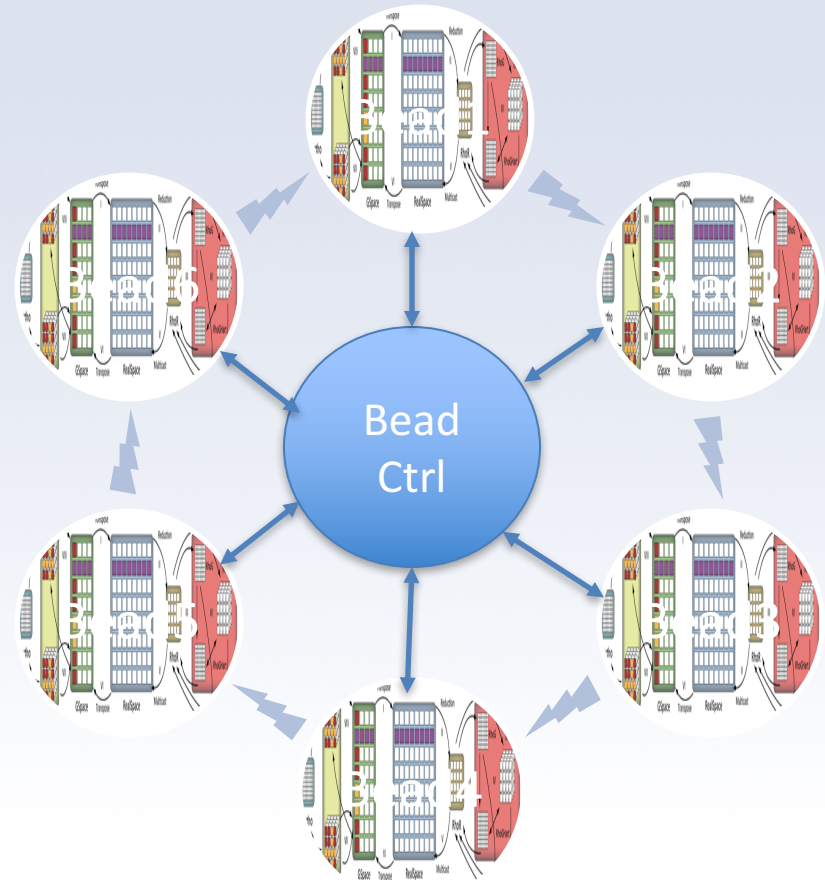
Spin Orbitals (LSDA)

- Each Spin shares : atom and energy chares
- 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 chares



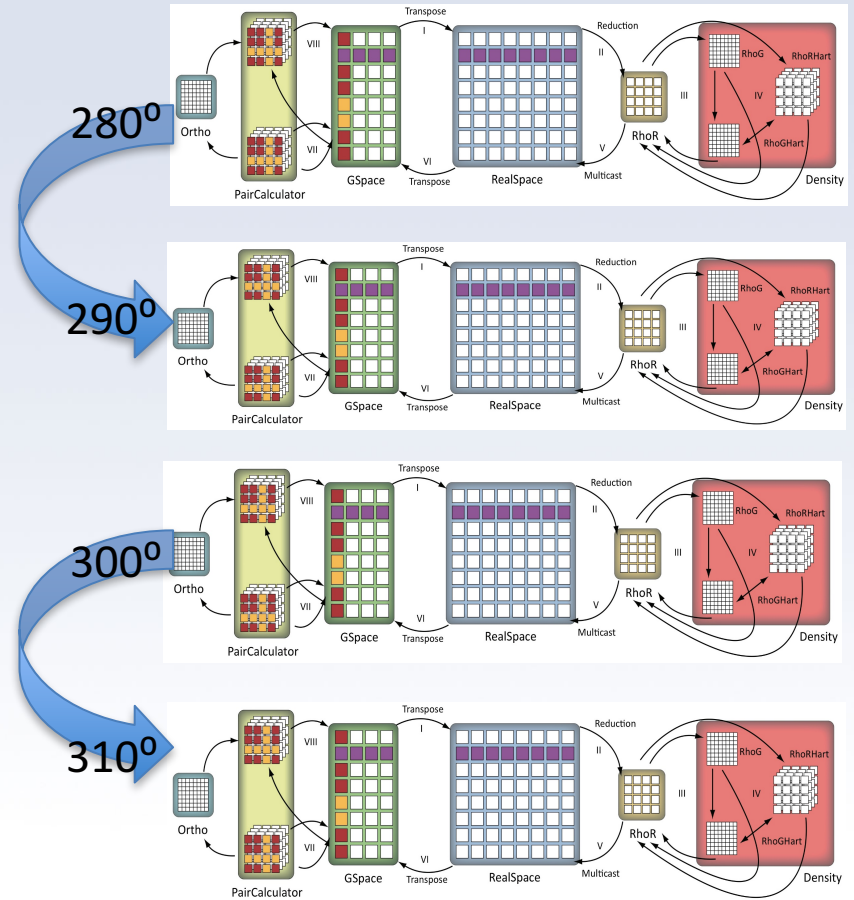
Path Integral Beads

- 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 chares



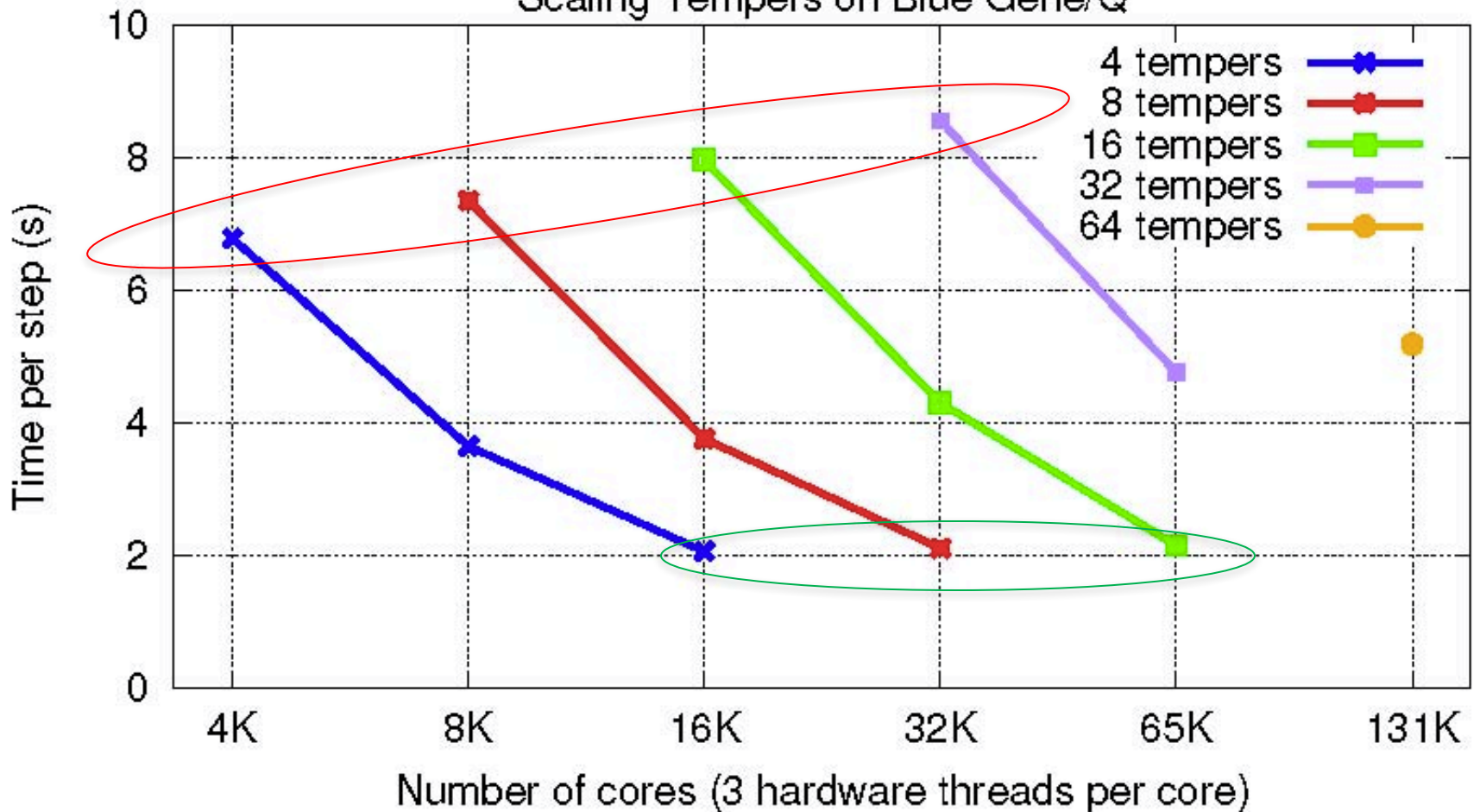
Tempers

- 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 chares

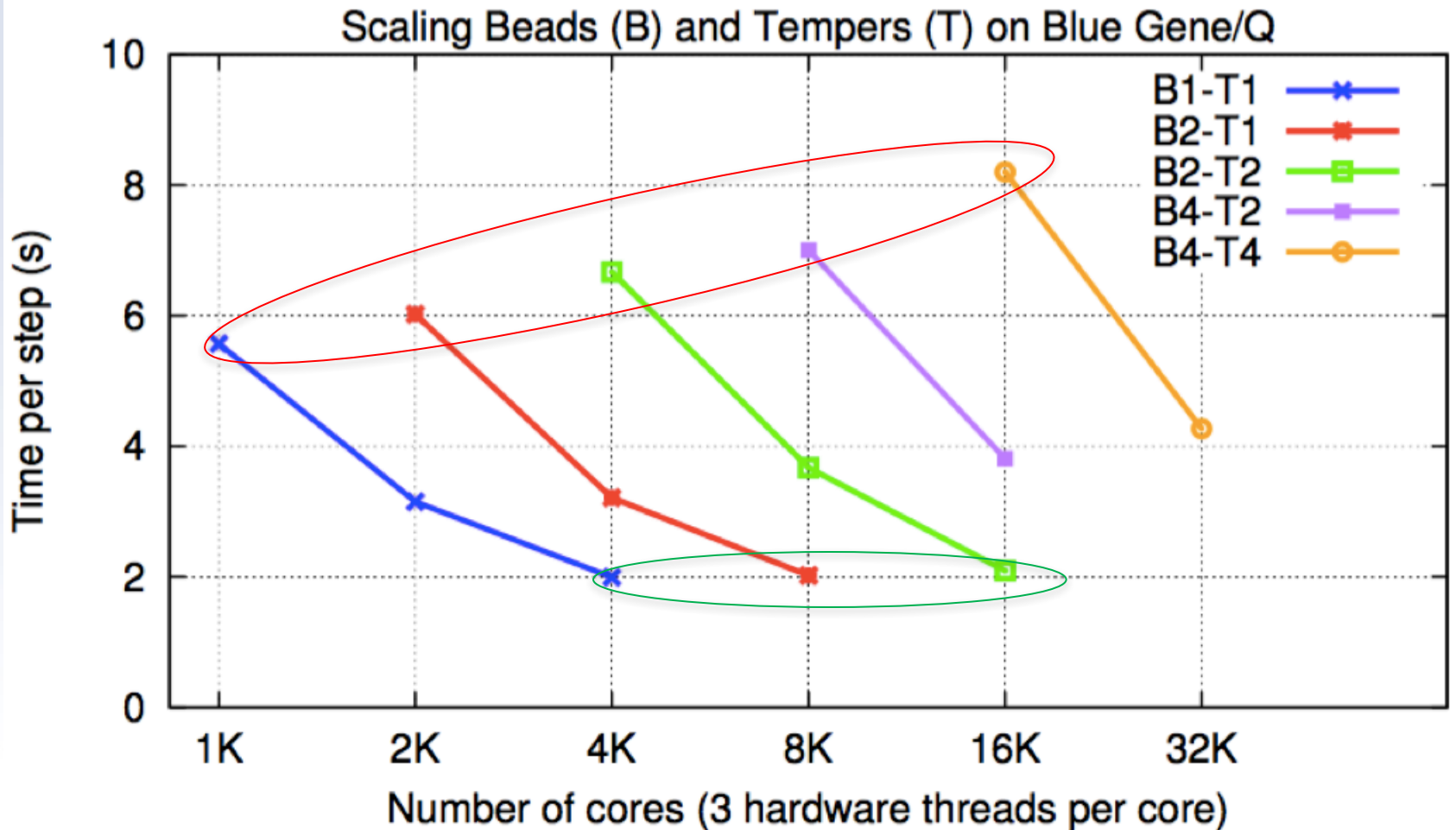


Temper Performance

Scaling Tempers on Blue Gene/Q



Combined Performance



Please Refer to : Heterogeneous Computing in Charm++

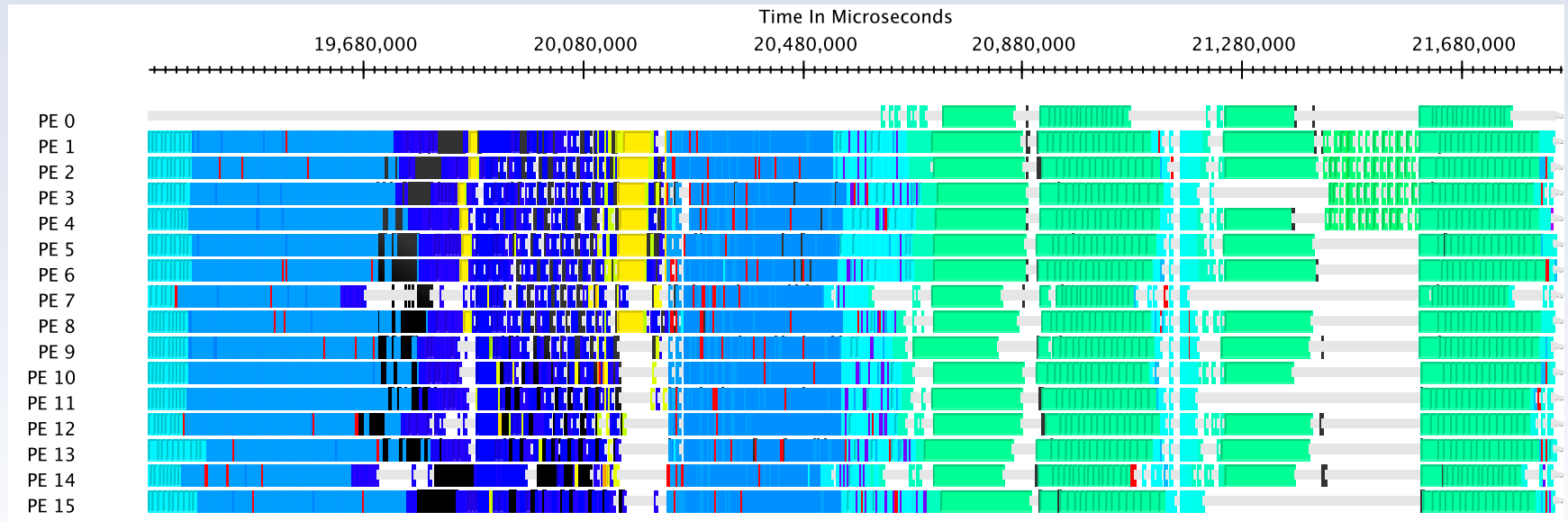
Michael Robson

GPGPU



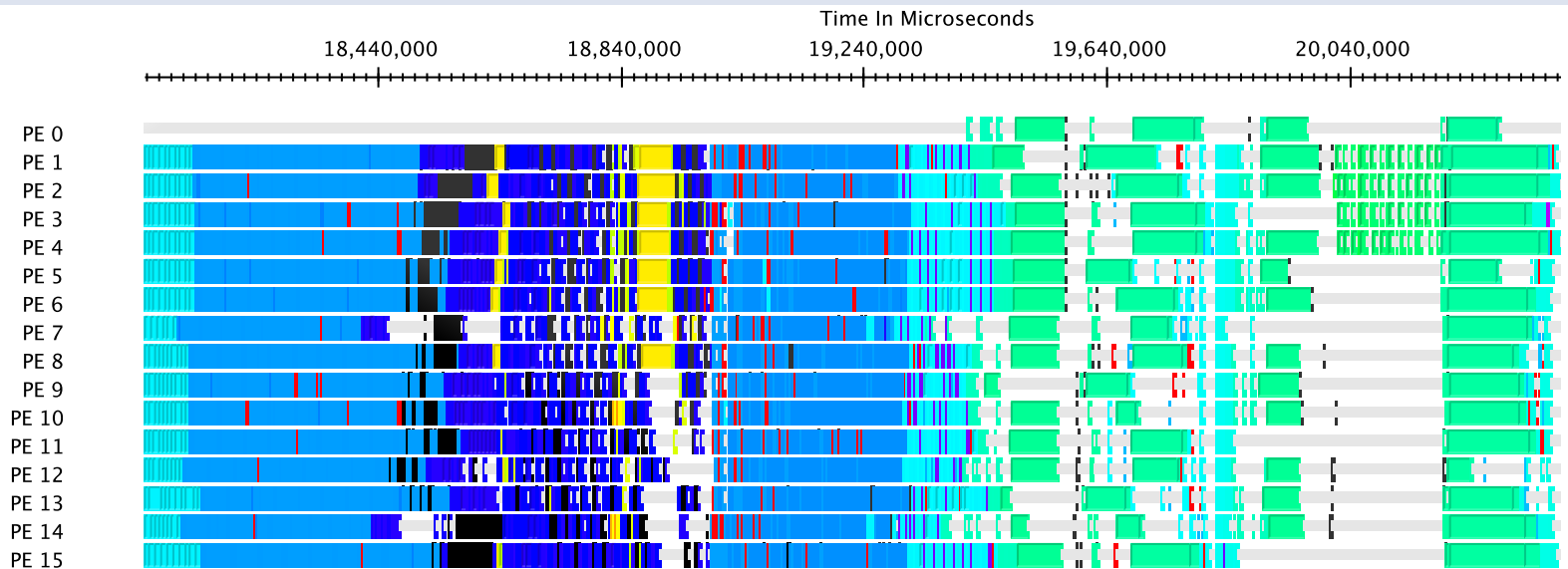
OpenAtom CPU Performance

Water 256M_70Ry 64 Nodes XK



OpenAtom GPU Performance

Water 256M_70Ry 64 Nodes XK



Eric Bohm, Qi Li, Glenn Martyna

PAW PARALLELIZATION



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
 - $\text{num_coretype} * \text{num_channel} * \text{num_projector}$
 - core_1 core_2



Parallelization Design

- Control flow dependencies introduced by PAW
 - PAW elements
 - f_{grid}
 - Bsplines
 - core zmatrices
 - Interactions with existing data structures
 - ρ^{s}
 - PW ρ is now the smooth part of ρ
 - ψ^{s}
 - PW ψ is now the smooth part of ψ
 - 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: SI2-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

$$P(G, G') = \sum_{v,c} \langle c | e^{-iG \cdot r} | v \rangle \langle v | e^{iG' \cdot r} | c \rangle \frac{2}{\epsilon_v - \epsilon_c}$$

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

$$P(r, r') = \sum_{v,c} \psi_c^*(r) \psi_v(r) \psi_v^*(r') \psi_c(r') \frac{2}{\epsilon_v - \epsilon_c}$$

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

- 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)

* Wilson, Gygi, Galli, *PRB* **78**, (2008); Govoni, Galli, *J. Chem. Th. Comp.*, **11** (2015)

* Gao, Xia, Gao, Zhang, *Sci. Rep.* **6** (2016)

† Liu, Kaltak, Klimes, and Kresse, *PRB* **94**, (2016)

Steps for typical G_0W_0 calculation

Stage 1 : Run DFT calc. on structure \rightarrow output : ϵ_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

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

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

Stage 5 : put together ϵ_i , $\psi_i(r)$ and $\epsilon^{-1}(\omega) \rightarrow$ self-energy $\Sigma(\omega)$

Inverting epsilon

$$\varepsilon(G, G') \rightarrow \varepsilon^{-1}(G, G')$$

Iterative matrix inversion for Hermitian matrix A:

A. Ben-Israel and D. Cohen, *SIAM J. Numer. Anal.*, 3:410-419, 1966

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

$$X_{n+1} = X_n (2I - AX_n)$$

We just “borrow” the pre-existing OpenAtom+charmm fast parallel matrix multiplication

Steps for typical G_0W_0 calculation

Stage 1 : Run DFT calc. on structure \rightarrow output : ϵ_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

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

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

Stage 5 : put together ϵ_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

Band Gaps (eV)					
System	Experiment	GW (full)	COHSEX	Corrected COHSEX*	DFT-LDA
Diamond	5.48	5.70	6.99	5.93	4.2
Si	1.17	1.29	1.70	1.18	0.49

$$\Sigma(r, r') = \Sigma^X(r, r') + \Sigma^{SEX}(r, r') + \Sigma^{COH}(r, r')$$

$$\Sigma^X(r, r') = - \sum_v \psi_v(r) \psi_v(r')^* \frac{1}{|r - r'|}$$

$$\Sigma^{SEX}(r, r') = - \sum_v \psi_v(r) \psi_v(r')^* [W(r, r') - 1/|r - r'|]$$

$$\Sigma^{COH}(r, r') = \frac{1}{2} \delta(r - r') [W(r, r') - 1/|r - r'|]$$

GW-Static Self-Energy (COHSEX)

Interestingly, direct real space method is not best here
Wave vector (Fourier) space is better computationally
Serial version written and correctness tested

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

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

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

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

$$\langle n | \Sigma^{COH} | n' \rangle = \frac{1}{2} \sum_{G,G'} S_{G,G'} \times f^{nn'}(G - G')$$

GW: some math details

1. Calculate RPA polarizability P

$$P_{q,q'}(\omega) = \sum_{\substack{c,v \\ c,v}} \frac{2(\epsilon_c - \epsilon_v)}{\omega^2 - (\epsilon_c - \epsilon_v)^2} \cdot \langle v | e^{iq \cdot \hat{r}} | c \rangle \langle v | e^{iq' \cdot \hat{r}} | c \rangle^*$$

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 Σ for each state n

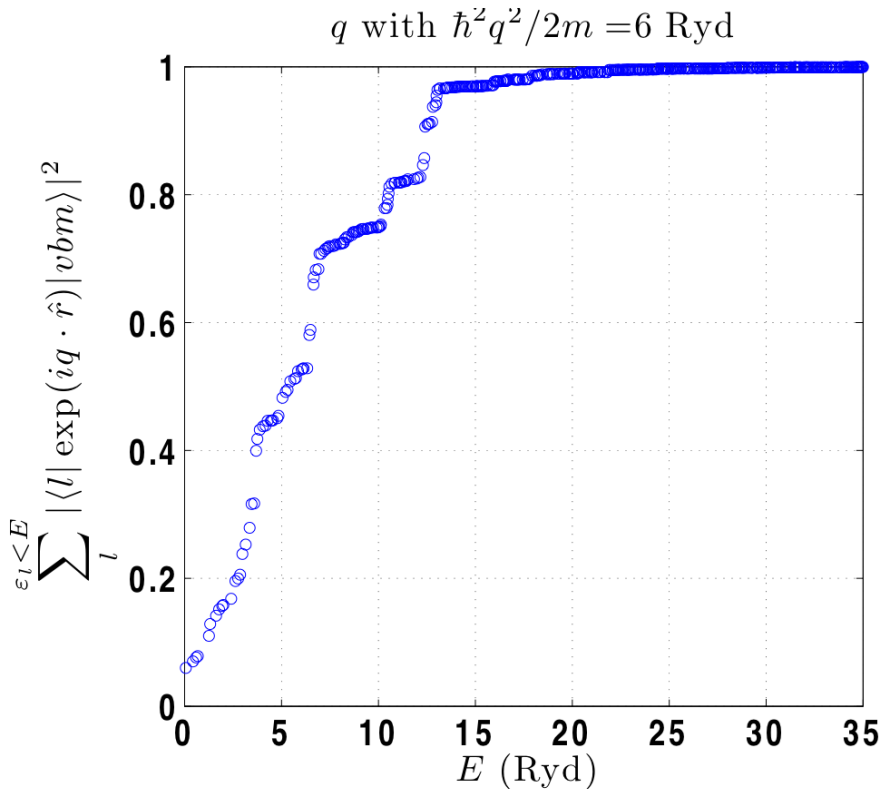
$$\begin{aligned} \langle n | \Sigma(\epsilon_n) | n \rangle &= \sum_{q,q'} \sum_l \left(\sum_p \frac{B_{q,q'}^p}{\epsilon_n - \epsilon_l - \omega_p} \right) \cdot \langle n | e^{iq \cdot \hat{r}} | l \rangle \langle n | e^{iq' \cdot \hat{r}} | l \rangle^* \\ &\quad - \sum_{q,q'} \sum_v W_{q,q'}(\epsilon_n - \epsilon_v) \cdot \langle n | e^{iq \cdot \hat{r}} | v \rangle \langle n | e^{iq' \cdot \hat{r}} | v \rangle^* \end{aligned}$$

GW: matrix elements

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

Simple sum rule

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



\therefore Need $\epsilon_l \gtrsim E_q$ to converge

Why?

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

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

$$P(r, r', \omega) = \sum_{c,v} \psi_c(r) \psi_v^*(r) \psi_c^*(r') \psi_v(r') \times \left[\frac{1}{\omega - (\epsilon_c - \epsilon_v)} - \frac{1}{\omega + (\epsilon_c - \epsilon_v)} \right]$$

GW: details 2

Solving Dyson's equation: write as perturbation on DFT

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

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 $|\psi_j\rangle \approx |\psi_j^{DFT}\rangle$ so system already diagonal

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

- Evaluate Σ and $d\Sigma/d\epsilon$ at ϵ_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

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

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

Typical: Local Density Approximation (LDA)

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