QMC Workshop 2021

Solids

Week 5 // 2 November

Joshua P Townsend, jptowns@sandia.gov

https://github.com/QMCPACK/qmcpack_workshop_2021

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. Funding: U.S. Department of Energy, Office of Science, Basic Energy Sciences, Materials Sciences and Engineering Division, as part of the Computational Materials Sciences Program and Center for Predictive Simulation of Functional Materials. SAND2021-13825 C

CPSFM Center for Predictive Simulation of Functional Materials

Sandia National Laboratories

Argonne NATIONAL LABORATORY

NC STATE UNIVERSITY

BROWN

OAK RIDGE National Laboratory

Topics Covered

1. Boundary conditions for solids & twist-averaging
2. Many-body wave functions for solids
3. Finite size effects
4. Laboratory exercises

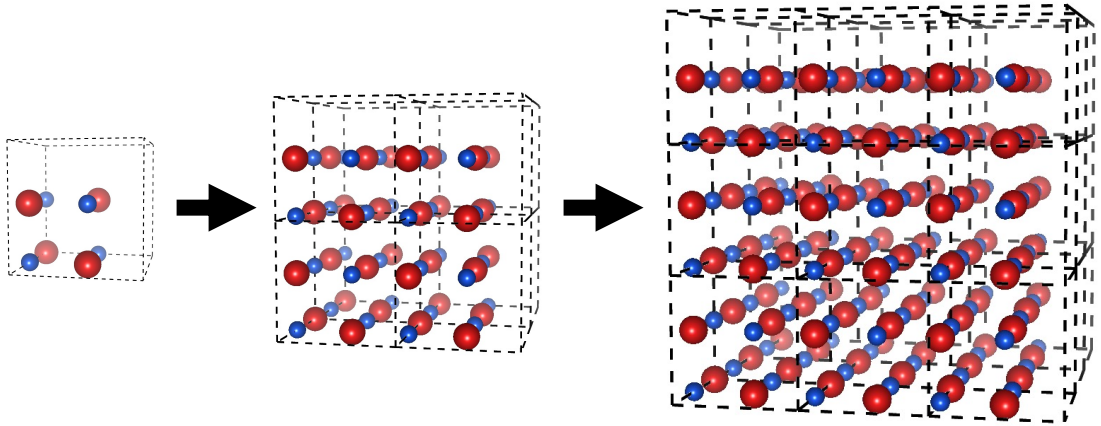
Topics Not Covered

1. QMC equilibration (covered in week 3)
2. DMC timestep extrapolation (covered in week 4)
3. DMC population bias (covered in week 4)
4. Mixed estimator bias (covered in week 6)
5. Pseudopotentials (covered in week 6)
6. Spins & magnetism (covered in week 8)
7. Systematic improvability (covered in week 4)

Topics Covered

1. **Boundary conditions for solids & twist-averaging**
2. Many-body wave functions for solids
3. Finite size effects
4. Laboratory exercises

From isolated molecules to bulk systems



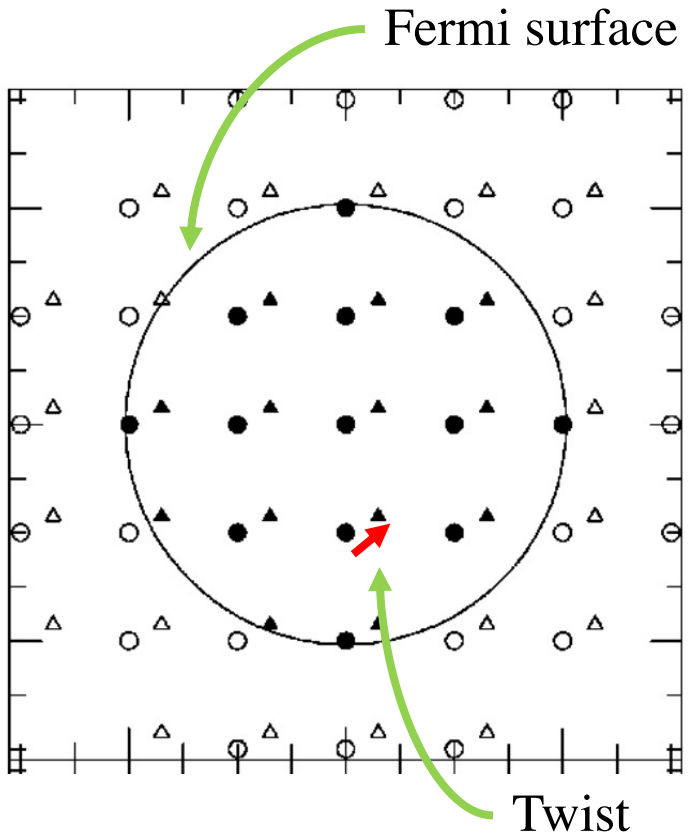
- In open BC's surface effects spoil convergence of, e.g. energy/atom
- Means that you need *many* atoms to get accurate bulk properties
- For solids we use **twisted boundary conditions**
- Eliminates surfaces, need fewer atoms, but how to handle LR interactions?

Twist-Averaged Boundary Conditions

- Many-body version of Bloch's Theorem:

$$\Psi(\mathbf{r}_1, \mathbf{r}_2 + \mathbf{L}, \ldots, \mathbf{r}_N) = e^{i\mathbf{L}\cdot\mathbf{k}}\Psi(\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_N)$$

- The "twist", $\theta \equiv \mathbf{L} \cdot \mathbf{k}$
- Regular PBC's $\rightarrow \theta = 0$
- TABC = Many-body version of BZ integration
- Reduces finite size effects[1] (more on this later)
- Faster convergence to thermodynamic limit[1]



[1]Lin & Ceperley, PRE **64**, 016702 (2001)

Twist-Averaged Boundary Conditions

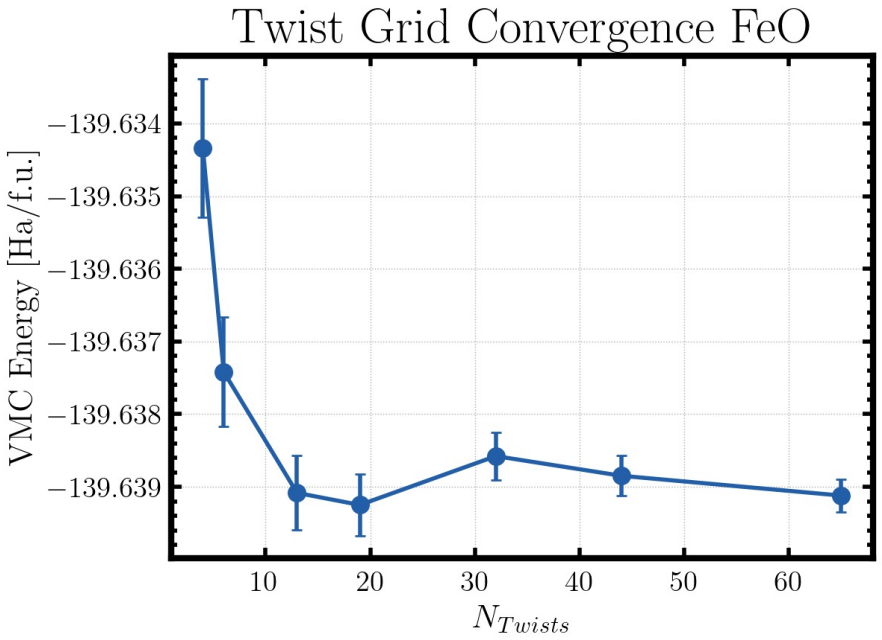
- Must apply TABC to all observables, not just energy!
- In general for TABC:

$$\langle A \rangle = \frac{1}{N_\theta} \sum_\theta A_\theta w_n$$

- A_θ is the usual MC average of A for a particular twist
- w_θ is the corresponding weight (just like a k-point mesh in DFT)

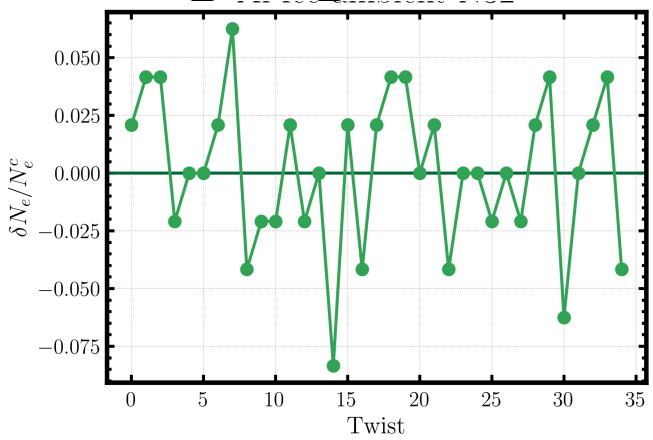
Twist Averaging – In Practice

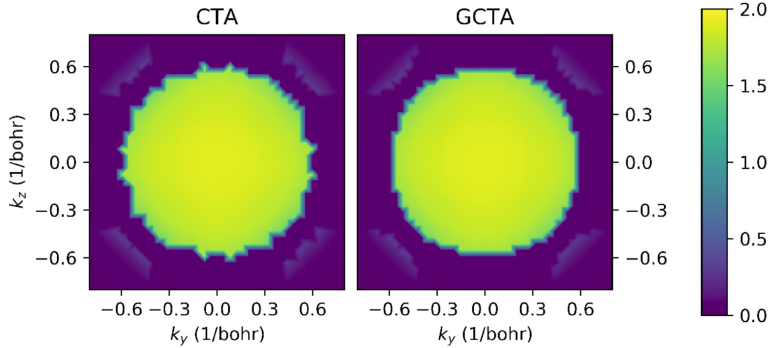
- Good news! Twists are independent
- Error reduction $\propto \frac{1}{\sqrt{N_{Twists}}}$
- How many twists are enough?
  - Depends on size, metallic vs insulating
  - You must check this yourself!
- Nexus can generate primitive cell k-point to supercell twist grid mapping for you



Grand Canonical Twist-Averaging

- In metals, N of occupied states depends on $\boldsymbol{k}$ due to bands crossing the Fermi surface
- Canonical TABC = Constant N for all twists
- GCTABC = N varies with twist
- Smaller finite size effects, but not always clear how to choose N in practice[1,2,3]

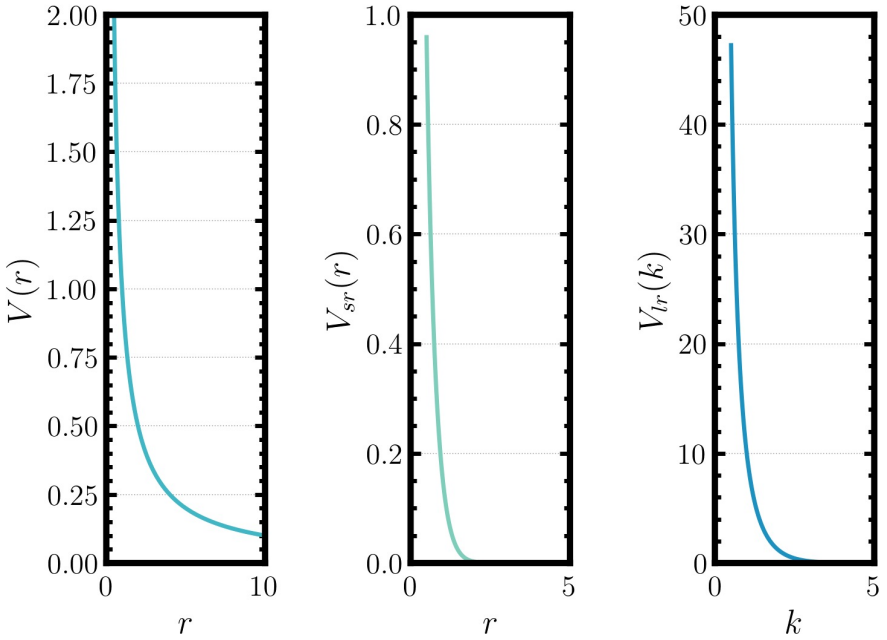




[1]Lin & Ceperley, PRE **64**, 016702 (2001)
[2]Azadi & Foulkes, PRB **100**, 245142 (2019)
[3]Yang et al, PRB **101**, 165125 (2020)

Ewald Summation

- How to compute Coulomb interaction?
- Conditionally convergent in PBC's (!)
- Ewald summation provides rigorous value via breakup into LR and SR parts:



Short-range r-space

Long-range k-space

$$V(\mathbf{r}) = \frac{1}{|\mathbf{r}|} = \sum_{\mathbf{L}} \frac{1}{|\mathbf{L}-\mathbf{r}|}\operatorname{erfc}(G|\mathbf{L}-\mathbf{r}|) + \frac{4\pi}{\Omega}\sum_{\mathbf{k}} \frac{e^{-|\mathbf{k}|^2/4G^2}}{|\mathbf{k}|^2} e^{i\mathbf{k}\cdot\mathbf{r}}$$

Ewald Summation

- How many times do we do Ewald in a typical QMC calculation?
  (Typical means ~100 electrons, ~1000 walkers, ~1000 steps)
  - For each MC step, propose a move for ~100 electrons
  - For each proposed move, evaluate ~100 interactions
  - Typically have O(1000) walkers
  - Typically take O(1000) steps
- Interaction calculated $\sim 10^{10}$ times!
- QMCPACK uses "optimized breakup" to maximize speed and minimize error[1]
  - Controlled via **`LR_DIM_CUTOFF`** in the input file.
  - Breakup is checked before calculation starts

[1]Natoli & Ceperley, JCP **117**, 171 (1995).

Topics Covered

1. Boundary conditions for solids & twist-averaging
2. **Many-body wave functions for solids**
3. Finite size effects
4. Laboratory exercises

Q&A break

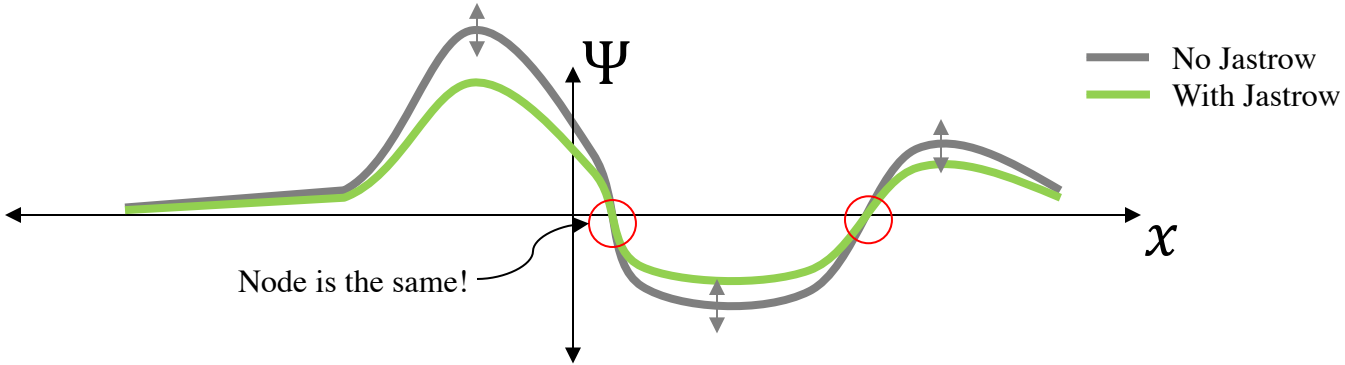
QMC Wave Functions for Solids

- Slater-Jastrow (SJ) is the simplest, most-compact, and most common many-body wave function ansatz:

$$\Psi^{SJ}(X) = e^{J(X)} \det \begin{vmatrix} \phi_1(\mathbf{x}_1) & \phi_1(\mathbf{x}_2) & \cdots & \phi_1(\mathbf{x}_N) \\ \phi_2(\mathbf{x}_1) & \phi_2(\mathbf{x}_2) & \cdots & \phi_2(\mathbf{x}_N) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_N(\mathbf{x}_1) & \phi_N(\mathbf{x}_2) & \cdots & \phi_N(\mathbf{x}_N) \end{vmatrix}$$

- Explicitly incorporates (some but not all) electron correlations
- The determinant piece comes from outside QMC
  - E.g. DFT orbitals from Quantum ESPRESSO, RMG, PySCF…
- The Jastrow is constructed within QMC by VMC optimization

QMC Wave Functions for Solids – Jastrow

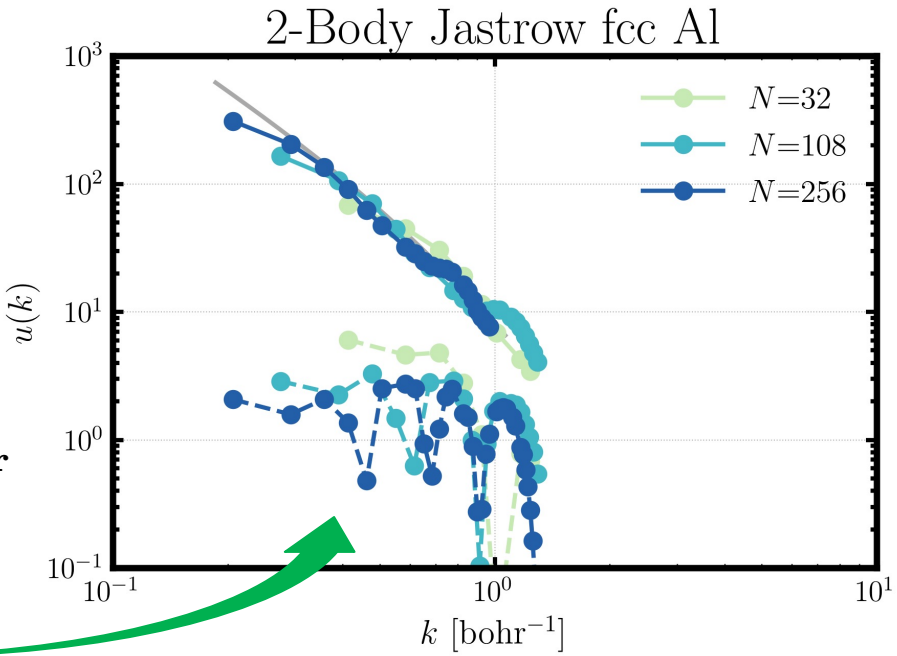


- 1B: Correlations between electron and atomic nucleus
- 2B: (Most important) Correlations between pairs of electrons
- 3B: (high Z semi-core states) Correlations between pairs of electrons and atomic nucleus
- K-space: (PBC's only) Restore long-range correlations in J2
- **Nodes** are not affected, only **amplitude** of Ψ!

QMC Wave Functions for Solids – Jastrow

- Correlation goes like 1/r, but truncated at cell boundary[1,2]
- Restore proper long-range behavior via 2-body Jastrow in k-space:

$$u(\mathbf{k}) = \sum_{\mathbf{k}} c(\mathbf{k}) \rho_{\mathbf{k}} \bar{\rho}_{\mathbf{k}}, \qquad \rho_{\mathbf{k}} \equiv \sum_{\mathbf{r}} e^{i\mathbf{k}\cdot\mathbf{r}}$$



[1]Gaskell *Proc. Phys. Soc.* **77**, 1182 (1961)
[2]Becker et al., PRL **175**, (1968)

QMC Wave Functions for Solids - Orbitals

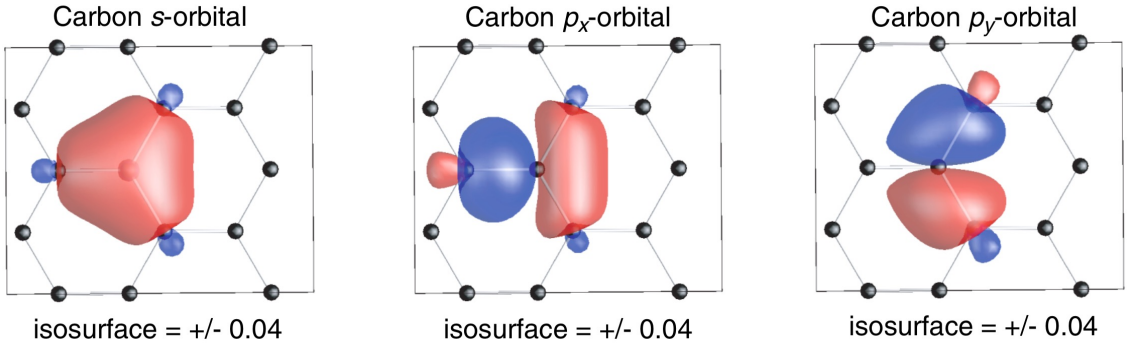
- DFT orbitals are commonly defined in a plane wave basis:

$$\phi_m(\mathbf{x}) = \sum_n c_{mn} e^{i\mathbf{k}_n \cdot \mathbf{x}}$$

- There are ~10^4 planewaves per orbital
- That's ~10^6 function calls *per move*
- That's ~10^8 function calls *per MC step*
- Quite expensive to call so many sin() & cos() !!!

QMC Wave Functions for Solids – Orbitals

- Key idea: Use 3d b-splines to represent each orbital[1]

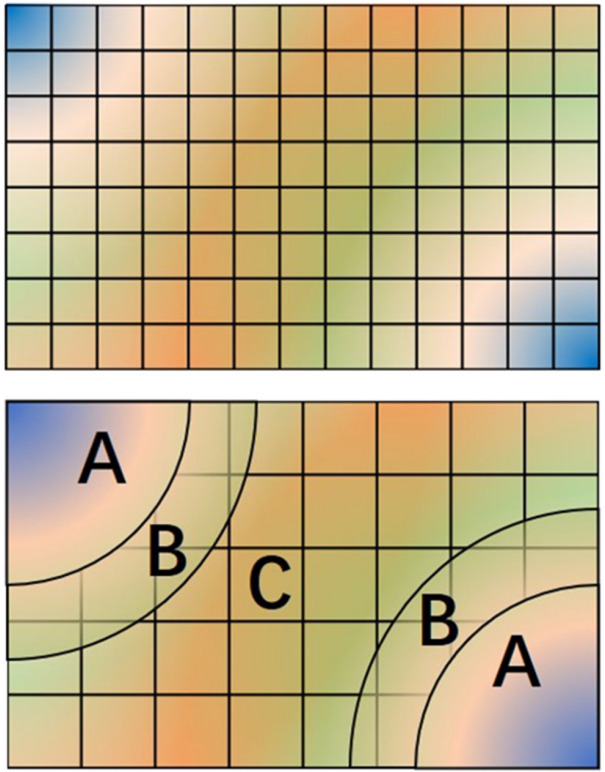


- Evaluate 64 grid points per orbital independent of system and basis set size
- Efficient routines transform DFT orbitals to splines automatically

[1]Alfe and Gillan, PRB **70**, 161101 (2004)
Fig from: Marzari et al. RMP **84**, (2012)

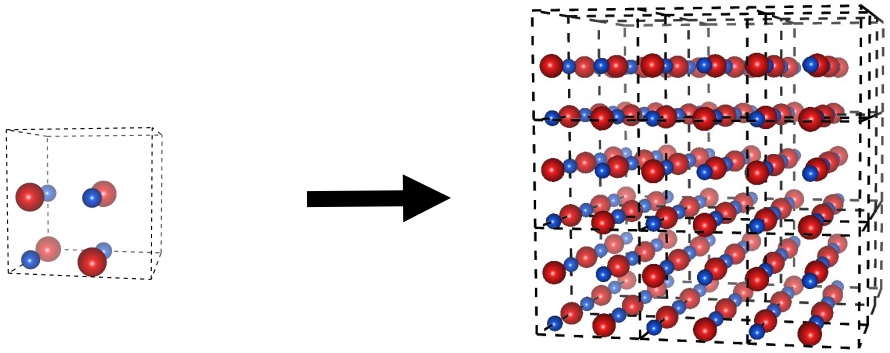
QMC Wave Functions for Solids – Orbitals

- We can further reduce memory costs using a hybrid representation[1]
- Use spherical harmonics near nucleus (A)
- Smooth transition region (B)
- Normal 3d B-splines in intersitial space (C)
- Memory reduction of 4-8x possible!
- Controlled by **`hybridrep`** tag
  - See folder "`hybridrep-tests`"



[1]Luo et al. JCP **149**, 084107 (2018)

QMC Wave Functions for Solids – k points



3x3x3 kpoint mesh here = 1x1x1 kpoint mesh here

- Another trick: Exploit crystal symmetry
- Generate SPOs in prim. cell with k-points, then tile into super cell
  - Straightforward application of Bloch's theorem
- This gets you an additional memory reduction of factor of N_{copies}

Topics Covered

1. Boundary conditions for solids & twist-averaging
2. Many-body wave functions for solids
3. **Finite size effects**
4. Laboratory exercises

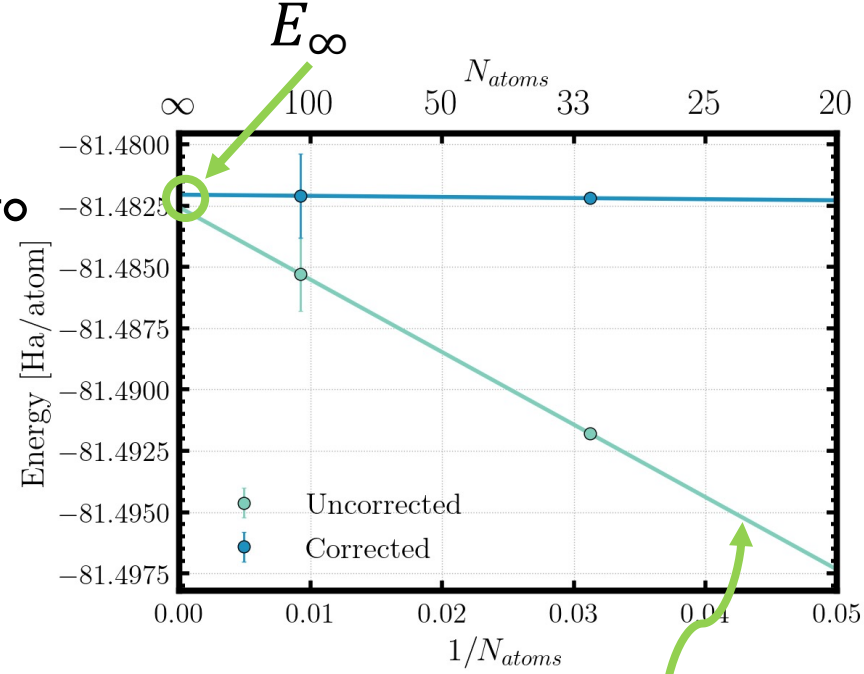
Q&A break

What Are Finite Size Effects?

- Finite size errors arise because in PBC's space & momentum are restricted
  - Think discrete vs continuous set of points in BZ for finite cell
  - FSE exist *even if you had the exact* Ψ!
- So every QMC output (on solids) is biased when comparing to experiment
- Two ways to get QMC estimates in thermodynamic limit:
  1. Extrapolate over system size
  2. Use finite size correction schemes

Finite Size Correction – Basic Idea

- In "big enough" cell, correction is an integration error
- Can derive expressions for corrections to kinetic and potential energy
- Must-read papers on FSC schemes:
  - Chiesa et al. *PRL* **97**, 076404 (2006)
  - Drummond et al. *PRB* **78**, 125106 (2008)
  - Holzmann et al. *PRB* **94**, 035126 (2016)
- Excellent recent applications:
  - Yang et al. PRB **101**, 165125 (2020)
  - Annaberdiyev et al. PRB **103**, 205206 (2021)



$$E\left(\frac{1}{N}\right) = E_\infty + m\frac{1}{N}$$

Potential Energy Finite Size Corrections

- Ewald gets Hartree energy exact, but biases XC energy
- Model Periodic Coulomb (MPC) fixes that[1]:

$$V_{MPC} = \frac{1}{2}\sum_{i\neq j} f(r_i - r_j) + \sum_i \int_\Omega \rho(r)[V_e(r_i - r) - f(r_i - r)]dr - \frac{1}{2}\int_\Omega \rho(r)\rho(r')[V_e(r - r') - f(r - r')]dr\,dr'$$

Normal Coulomb inside

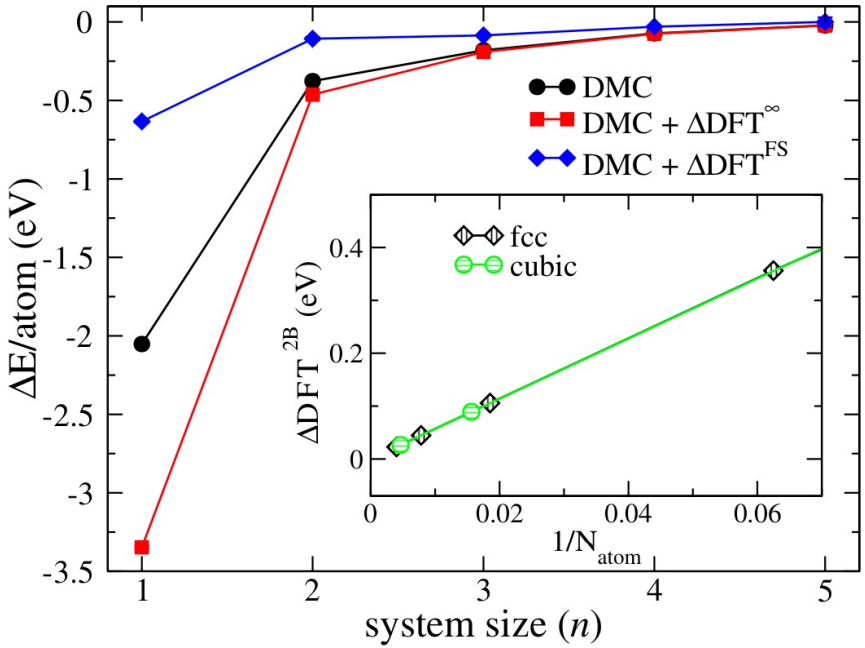
Modified potential outside

- Not a real FSC by itself, but sometimes used in that way
- See **MPC** potential for details

[1]Kent et al. PRB **59**, 1917 (1999)

Potential Energy Finite Size Corrections

- KZK correction[1]:
- Take $E_{XC}(N)$ from the HEG
- In ESPRESSO: **`dft="kzk"`**
  - LDA only by default
- Entirely independent of QMC
- Gives approximate 2-body KE & PE corrections



[1]Kwee et al. PRL **100**, (2008)

Potential Energy Finite Size Corrections

- **Gold standard**: Use structure factor[1,2,3]
- PE per particle is related to s(k) via:

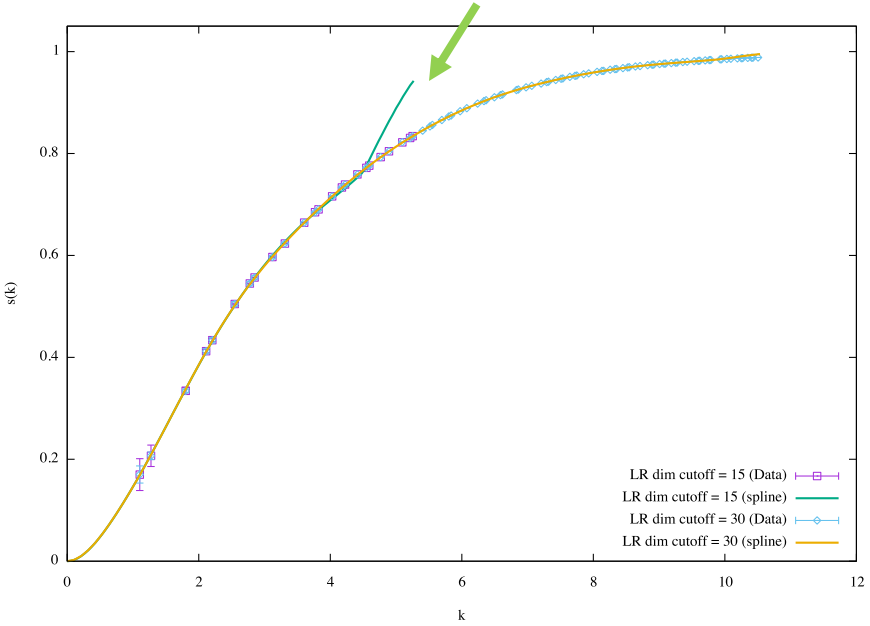
$$V_N = \frac{2\pi}{\Omega} \sum_{\mathbf{k}\neq 0} \frac{1}{k^2} \left(s(\mathbf{k}) - 1\right)$$

- Correction is an integration error:

$$\Delta V_N = \frac{1}{4\pi^2} \int d\mathbf{k}\, \frac{s(\mathbf{k}) - 1}{k^2} - \frac{2\pi}{\Omega} \sum_{\mathbf{k}\neq 0} \frac{s_N(\mathbf{k}) - 1}{k^2}$$

- Use **`SkAll`** + qmcfinitesize in postprocessing

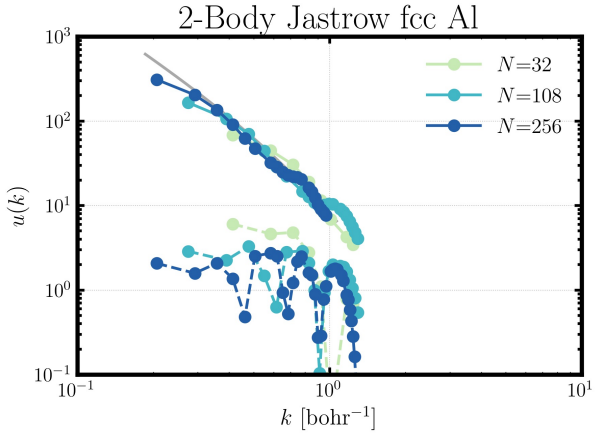
Change **`LR_dim_cutoff`**

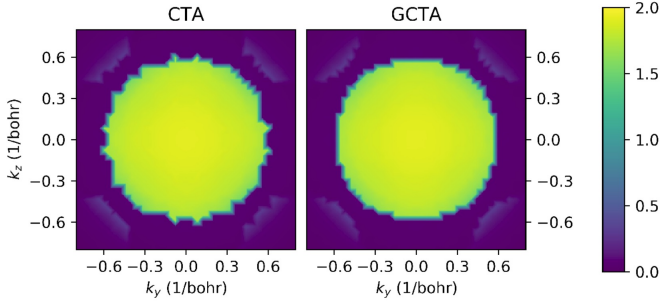


[1]Chiesa et al. PRL **97**, 076404 (2006)
[2]Drummond et al. *PRB* **78**, 125106 (2008)
[3]Holzmann et al. PRB **94**, 035126 (2016)

Kinetic Energy Finite Size Corrections

- Basically two ways to get a KE correction
- Examine LR behavior of J2:
  - See **`KECorr`** estimator in QMCPACK
  - Not guaranteed to be accurate – you must test!
- Momentum distribution:
  - See "nofk" estimator in QMCPACK
  - Done in post-processing. Must "roll your own"
- Excellent example of both can be found in Yang et al. PRB **101**, 165125 (2020)





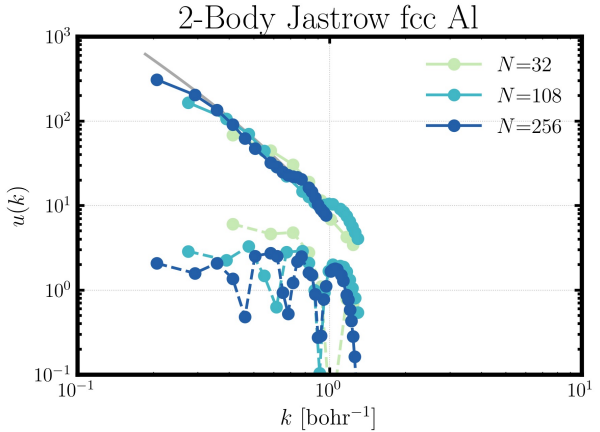
Kinetic Energy Finite Size Corrections

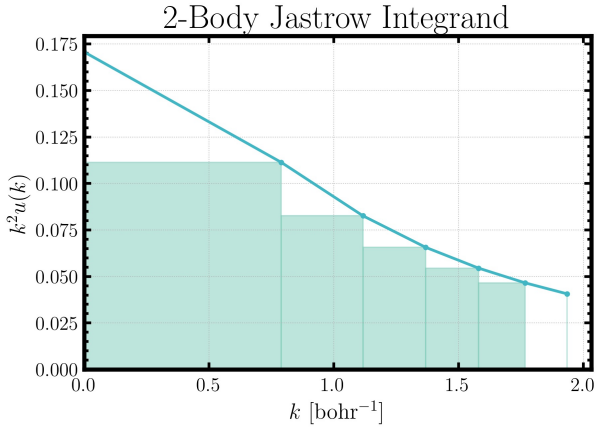
- Twist-averaging reduces kinetic finite size error, but often times further correction is needed
- Can obtain KE correction from long-ranged behavior of J2
- À la Chiesa et al.:

$$\Delta T = \frac{N}{4(2\pi)^3}\int d\mathbf{k} k^2 u(\mathbf{k}) - \frac{N}{4\Omega}\sum_{\mathbf{k}\neq 0} k^2 u_N(\mathbf{k}) \approx \frac{\pi a N}{\Omega}$$

with "a" determined from LR piece of u(k):

$$u(\mathbf{k}) = 4\pi a\left[k^{-2} - (k^2 + a^{-1})^{-1}\right]$$



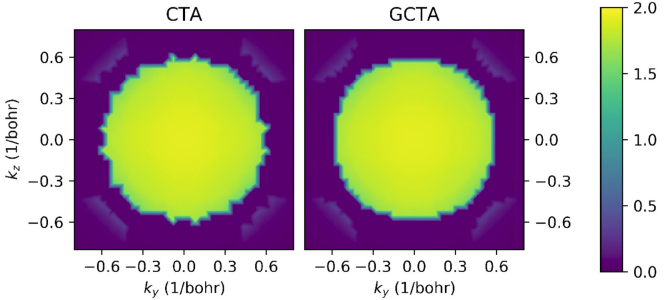


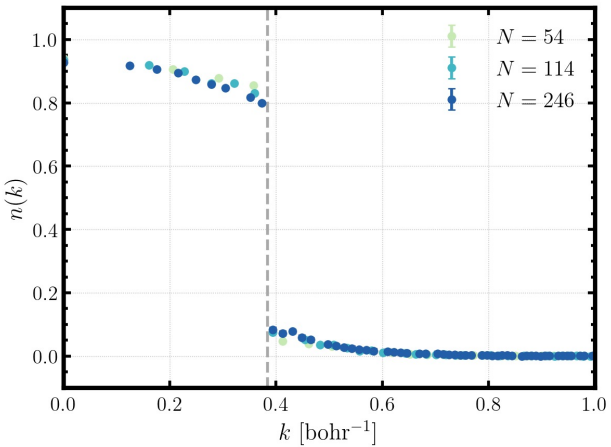
Kinetic Energy Finite Size Corrections

- **Gold standard**: Use momentum distribution[1]
- Independent of Jastrow
- GCTABC may be important
- Ala Holzmann et al.:

$$\Delta T = T_\infty - T_N = \frac{\Omega}{2N}\frac{1}{(2\pi)^3}\int d\mathbf{k} k^2 n_N(\mathbf{k}) - \frac{1}{2N}\sum_{\mathbf{k}} k^2 n_N(\mathbf{k})$$

- NB: $n(k)$ converges to "big enough" slowly!
  - Shape corrections may be important[2,3]





[1]Holzmann et al. PRB **79**, (2016)
[2]Yang et al. PRB 101, (2020)
[3]Holzmann et al. PRL **107**, (2011)

Finite Size Corrections – Final Words

- Many schemes, same physical quantities: $u(k)$, $s(k)$, $n(k)$
- QMCPACK provides all estimators, up to you to process correctly
- Different assumptions and approximations: Testing is required!
- Remember: When in doubt, get a larger supercell!

Checklist for QMC on Solids

1. Converge twist grid
2. Exploit symmetry
3. Plan ahead for estimators/observables
4. Establish work required for desired accuracy in a small cell
5. Plan to quantify/mitigate finite size effects
6. When in doubt, get a bigger supercell!
7. Reach out to QMCPACK community: qmcpack.slack.com

Topics Covered

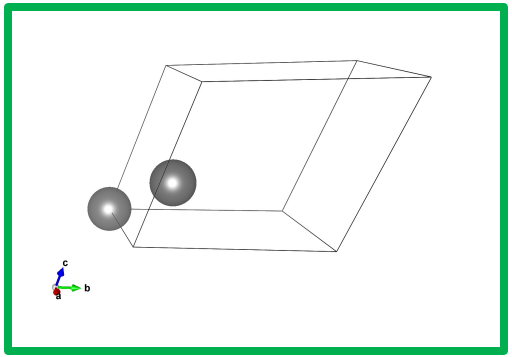
1. Boundary conditions for solids & twist-averaging
2. Many-body wave functions for solids
3. Finite size effects
4. **Laboratory exercises**

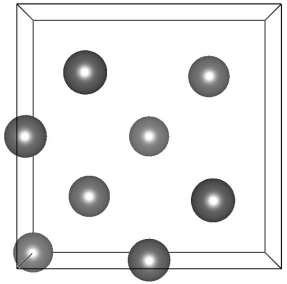
Q&A break

Hands-On Tutorial

- Step 1: DFT & wave function generation in 2-atom primitive cell
- Step 2: QMC Jastrow optimization in 8-atom supercell
- Step 3: QMC Twist grid convergence scan in 8-atom supercell
- Step 4: Production DMC on 8-atom supercell
- Step 5: Analyze results

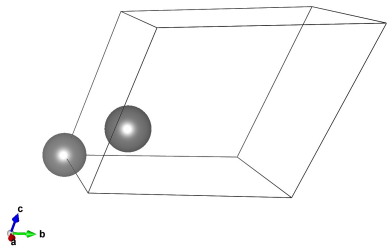
Step 0: Specify the System

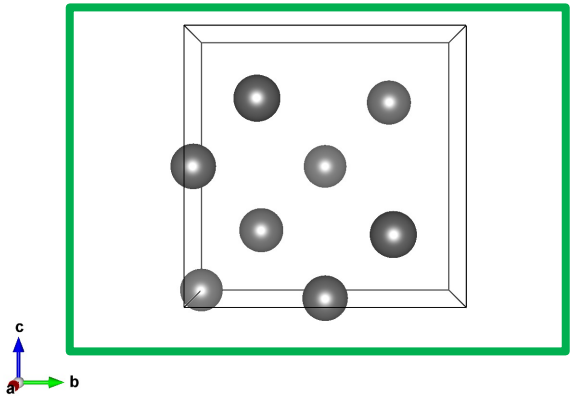





```
a = 3.57
primcell  = generate_physical_system(
    units = 'A',                    # Angstrom units
    axes  = [[a/2, a/2,   0],       # Cell axes
             [  0, a/2, a/2],
             [a/2,   0, a/2]],
    elem  = ['C','C'],              # Element names
    posu  = [[0.00, 0.00, 0.00],    # Element positions (crystal units)
             [0.25, 0.25, 0.25]],
    C     = 4,                      # Pseudpotential valence charge
)
```

Step 0: Specify the System





Set supercell size & twistgrid

```
tiled = primcell.structure.tile_opt(4)
tiled.add_symmetrized_kmesh(kgrid=(2,2,2),kshift=(0,0,0))
supercell = generate_physical_system(structure = tiled, C=4)
```

Step 1: DFT

Nexus input

```
scf = generate_pwscf(
    ...
    calculation  = 'scf',
    input_dft    = 'pbe',
    ...
    system       = primcell,
    pseudos      = ['C.ccECP.upf'],
    kgrid        = (10,10,10),
    kshift       = (0,0,0),
)
```

PWSCF output

```
 highest occupied, lowest unoccupied level (ev):    13.3562   17.5508

!    total energy              =     -22.74923353 Ry
     estimated scf accuracy    <          2.8E-11 Ry

     The total energy is the sum of the following terms:
     one-electron contribution =       7.99385929 Ry
     hartree contribution      =       1.90620439 Ry
     xc contribution           =      -7.09796276 Ry
     ewald contribution        =     -25.55133445 Ry

     convergence has been achieved in   8 iterations
```

~10 sec on 8c AMD desktop

Step 2: Make Orbitals

When complete, you should have a QMC wave function called:
`nscf/pwscf_output/pwscf.pwscf.h5`

- Enables rapid supercell set up
- Supports iteration
- Useful for preliminary "scouting" calculations to outline your problem
- *E.g.* Twist grid convergence

First

```
nscf = generate_pwscf(
    ...
    calculation  = 'nscf',
    input_dft    = 'pbe',
    ...
    system       = supercell,
    ...
)
```

Second

```
conv = generate_pw2qmcpack(
    identifier   = 'wfconv',
    path         = 'diamond/nscf',
    job          = job(cores=cores,
                       app=wfc_bin),
    write_psir   = False,
    dependencies = (nscf,'orbitals'),
)
```

~3 sec on 8c AMD desktop

Step 3: Wave function Optimization

Rules of thumb for Jastrow optimization:

- Don't do too much at once!
- More parameters = more samples
- `oneshift` recommended
- 1-body is medium-long range
- 2-body is looooooooong range
- 3-body is short-ranged and good for deep semi-core states
- K-space piece of J2 less important in larger supercells
- **Always use 1- and 2-body terms!**
- Test for 3-body and k-space terms

```
optJ12 = generate_qmcpack(
    ...
    system           = supercell,
    pseudos          = ['C.ccECP.xml'],
    twistnum         = 0,
    J1               = True,
    J2               = True,
    J1_rcut          = 3.37,
    J2_rcut          = 3.37,
    qmc              = 'opt',
    minmethod        = 'oneshift',
    init_cycles      = 4,
    cycles           = 10,
    warmupsteps      = 8,
    blocks           = 100,
    timestep         = 0.1,
    init_minwalkers  = 0.01,
    minwalkers       = 0.5,
    samples          = 20000,
    use_nonlocalpp_deriv = False,
    ...
)
```

~5 min on 8c AMD desktop

Step 4: Production DMC

Rules of thumb for DMC:

- Use VMC to “seed” DMC walkers
- Prefer ”wide” over “deep”
- Certain estimators = huge files
- My timestep strategy: Run for constant imaginary time
- Smaller timestep = more steps = more wall time
- Always use ≥ 3 timesteps
- QMC is a big hammer!
- When in doubt, add more twists
- If still in doubt, run bigger cell
- After all, FLOPs are free, right? 😀

```
qmc = generate_qmcpack(
    ...
    system           = supercell,
    twistnum         = tw,
    pseudos          = ['C.ccECP.xml'],
    qmc              = 'dmc',

    # Initial VMC stuff
    vmc_warmupsteps = 8,
    vmc_blocks       = 500,
    vmc_steps        = 8,
    vmc_timestep     = 0.1,

    # DMC stuff
    timestep         = 0.02,
    timestep_factor = 0.5,
    ntimesteps       = 3,
    steps            = 1,
    blocks           = 200,
    ...
  )
```

~16 min/twist on 8c AMD desktop

Step 4: Production DMC

Separate twists

		LocalEnergy	Variance	ratio
tw0/dmc	series 0	-45.028218 +/- 0.007974	1.487432 +/- 0.094222	0.0330
tw0/dmc	series 1	-45.165089 +/- 0.005259	1.450411 +/- 0.011777	0.0321
tw0/dmc	series 2	-45.157311 +/- 0.006050	1.441757 +/- 0.012758	0.0319
tw0/dmc	series 3	-45.165836 +/- 0.007257	1.427758 +/- 0.010743	0.0316
tw1/dmc	series 0	-45.805801 +/- 0.007940	1.362946 +/- 0.025835	0.0298
tw1/dmc	series 1	-45.925291 +/- 0.004509	1.392037 +/- 0.016996	0.0303
tw1/dmc	series 2	-45.941738 +/- 0.007447	1.399314 +/- 0.010596	0.0305
tw1/dmc	series 3	-45.932886 +/- 0.004683	1.408587 +/- 0.011354	0.0307
tw2/dmc	series 0	-45.804025 +/- 0.007759	1.384071 +/- 0.029984	0.0302
tw2/dmc	series 1	-45.940675 +/- 0.005337	1.374696 +/- 0.011753	0.0299
tw2/dmc	series 2	-45.955675 +/- 0.003979	1.367148 +/- 0.013787	0.0297
tw2/dmc	series 3	-45.939572 +/- 0.008252	1.431456 +/- 0.018716	0.0312
tw3/dmc	series 0	-45.229272 +/- 0.008749	1.411314 +/- 0.022209	0.0312
tw3/dmc	series 1	-45.374830 +/- 0.003001	1.501216 +/- 0.024146	0.0331
tw3/dmc	series 2	-45.376284 +/- 0.003209	1.459325 +/- 0.011732	0.0322
tw3/dmc	series 3	-45.385221 +/- 0.002824	1.477228 +/- 0.011048	0.0325

Twist-averaged

		LocalEnergy	Variance	ratio
avg	series 0	-45.635631 +/- 0.004364	1.506051 +/- 0.023551	0.0330
avg	series 1	-45.767763 +/- 0.004036	1.489250 +/- 0.006949	0.0325
avg	series 2	-45.779392 +/- 0.002245	1.495680 +/- 0.005244	0.0327
avg	series 3	-45.772102 +/- 0.004556	1.506595 +/- 0.011622	0.0329

Homework // Bonus Stuff

1. Test out hybridrep transformation. See directory:
   `runs/diamond_example_results/hybridrep-tests` for instructions
2. Test out k-space Jastrow optimization. See directory:
   `runs/diamond_example_results/optJk` for instructions
3. Test out 3-body Jastrow optimization. (See Nexus script for details)
   Compare Energy/variance with/without. Is J3 worth it for this problem?
4. Compare input format for parallel vs serial twist calculations. See directory:
   `runs/diamond_example_results/dmc-paralleltwists`
5. Timestep extrapolate twist-averaged results using sample from lab 4