VMC vs DMC

Study of the Ground State Energy of He^4

B. Dannowitz, B. Dellabetta, A. Ghalsasi

The University of Illinois at Urbana-Champaign

May 12, 2011

Outline

VMC Overview

- Objective: find E_0 and test out Ψ_T
- Model Ψ_T for liquid He^4
- Tune a_1 and a_2 to find minimum $E_0 \leq \frac{1}{N}\sum_i \frac{\hat{\mathcal{H}}\Psi_T(\mathbf{R}_i)}{\Psi_T(\mathbf{R}_i)} = \frac{1}{N}\sum_i E_L(\mathbf{R}_i)$

VMC Overview

- Objective: find E_0 and test out Ψ_T
- Model Ψ_T for liquid He^4
- Tune a_1 and a_2 to find minimum
$E_0 \leq \frac{1}{N} \sum_i \frac{\hat{\mathcal{H}}\Psi_T(\mathbf{R}_i)}{\Psi_T(\mathbf{R}_i)} = \frac{1}{N} \sum_i E_L(\mathbf{R}_i)$

VMC Overview

- Objective: find E_0 and test out Ψ_T
- Model Ψ_T for liquid He^4
- Tune a_1 and a_2 to find minimum $E_0 \leq \frac{1}{N}\sum_i \frac{\hat{\mathcal{H}}\Psi_T(\mathbf{R}_i)}{\Psi_T(\mathbf{R}_i)} = \frac{1}{N}\sum_i E_L(\mathbf{R}_i)$

VMC Overview

- Model Ψ_T for liquid He^4 as $\Psi_T = \prod_{i<j} \exp\left[-(a_1/r_{ij})^{a_2}\right]$
-
$$E_L(\mathbf{R}_i) = \frac{\hbar^2}{2m}\sum_{i=1}^{N}\frac{\nabla_i^2\Psi_T}{\Psi_T} + \sum_{i<j}^{N} V(r_{ij})$$
$$E_L(\mathbf{R}_i) = \frac{a_2(a_2-1)\hbar^2 a_1^{a_2}}{2mr^{a_2+2}} + \frac{4\epsilon\sigma^{12}}{r^{12}} - \frac{4\epsilon\sigma^6}{r^6}. \quad (1)$$
-
$$E_L(\mathbf{R}_i) = \sum_{i<j} V(r_{ij}) - 2\frac{\hbar^2}{2m}\nabla^2(a_1/r_{ij})^{a_2} - \frac{\hbar^2}{2m}\sum_i G_i^2$$
$$E_L(\mathbf{R}_i) = \sum_{i<j}\left[\frac{-a_2(a_2-1)\hbar^2 a_1^{a_2}}{mr^{a_2+2}} + \frac{4\epsilon\sigma^{12}}{r^{12}} - \frac{4\epsilon\sigma^6}{r^6}\right] \quad (2)$$
$$+\sum_i \frac{\hbar^2}{2m}\left[\frac{a_2 a_1^{a_2}}{r^{a_2+1}}\right]^2.$$

VMC Overview

- Model Ψ_T for liquid He^4 as $\Psi_T = \prod_{i<j} \exp\left[-(a_1/r_{ij})^{a_2}\right]$
- $$E_L(\mathbf{R}_i) = \frac{\hbar^2}{2m}\sum_{i=1}^{N}\frac{\nabla_i^2\Psi_T}{\Psi_T} + \sum_{i<j}^{N} V(r_{ij})$$
  $$E_L(\mathbf{R}_i) = \frac{a_2(a_2-1)\hbar^2 a_1^{a_2}}{2mr^{a_2+2}} + \frac{4\epsilon\sigma^{12}}{r^{12}} - \frac{4\epsilon\sigma^6}{r^6}. \quad (1)$$
- $$E_L(\mathbf{R}_i) = \sum_{i<j} V(r_{ij}) - 2\frac{\hbar^2}{2m}\nabla^2(a_1/r_{ij})^{a_2} - \frac{\hbar^2}{2m}\sum_i G_i^2$$
  $$E_L(\mathbf{R}_i) = \sum_{i<j}\left[\frac{-a_2(a_2-1)\hbar^2 a_1^{a_2}}{mr^{a_2+2}} + \frac{4\epsilon\sigma^{12}}{r^{12}} - \frac{4\epsilon\sigma^6}{r^6}\right] \quad (2)$$
  $$+\sum_i \frac{\hbar^2}{2m}\left[\frac{a_2 a_1^{a_2}}{r^{a_2+1}}\right]^2.$$

VMC Overview

- Model Ψ_T for liquid He^4 as $\Psi_T = \prod_{i<j} \exp\left[-(a_1/r_{ij})^{a_2}\right]$
-
$$E_L(\mathbf{R}_i) = \frac{\hbar^2}{2m}\sum_{i=1}^{N}\frac{\nabla_i^2\Psi_T}{\Psi_T} + \sum_{i<j}^{N} V(r_{ij})$$
$$E_L(\mathbf{R}_i) = \frac{a_2(a_2-1)\hbar^2 a_1^{a_2}}{2mr^{a_2+2}} + \frac{4\epsilon\sigma^{12}}{r^{12}} - \frac{4\epsilon\sigma^6}{r^6}. \quad (1)$$
-
$$E_L(\mathbf{R}_i) = \sum_{i<j} V(r_{ij}) - 2\frac{\hbar^2}{2m}\nabla^2(a_1/r_{ij})^{a_2} - \frac{\hbar^2}{2m}\sum_i G_i^2$$
$$E_L(\mathbf{R}_i) = \sum_{i<j}\left[\frac{-a_2(a_2-1)\hbar^2 a_1^{a_2}}{mr^{a_2+2}} + \frac{4\epsilon\sigma^{12}}{r^{12}} - \frac{4\epsilon\sigma^6}{r^6}\right] \quad (2)$$
$$+\sum_i \frac{\hbar^2}{2m}\left[\frac{a_2 a_1^{a_2}}{r^{a_2+1}}\right]^2.$$

VMC Algorithm

1. Initialize box of particles: L, N, a_1, a_2
2. For each particle, i

- Propose move from $\mathbf{r}$ to $\mathbf{r_i'} = \mathbf{r_i} + \xi \frac{\mathbf{L}}{\mathbf{2}}$
- Compute weight of move and accept with:

$$A(\mathbf{r_i} \rightarrow \mathbf{r_i}') = \min \left[1, \frac{|\Psi_T(\mathbf{r_i'})|^2}{|\Psi_T(\mathbf{r_i})|^2} \right] \quad (3)$$

- Compute E_L as per Eq. 1 and 2
- Repeat for many combinations of a_1 and a_2

VMC Algorithm

1. Initialize box of particles: L, N, a_1, a_2
2. For each particle, i

- Propose move from $\mathbf{r}$ to $\mathbf{r_i'} = \mathbf{r_i} + \xi\frac{\mathbf{L}}{\mathbf{2}}$
- Compute weight of move and accept with:

$$A(\mathbf{r_i} \rightarrow \mathbf{r_i}') = \min\left[1, \frac{|\Psi_T(\mathbf{r_i'})|^2}{|\Psi_T(\mathbf{r_i})|^2}\right] \quad (3)$$

- Compute E_L as per Eq. 1 and 2
- Repeat for many combinations of a_1 and a_2

VMC Algorithm

1 Initialize box of particles: L, N, a_1, a_2
2 For each particle, i

- Propose move from $\mathbf{r}$ to $\mathbf{r_i'} = \mathbf{r_i} + \xi\frac{\mathbf{L}}{\mathbf{2}}$
- Compute weight of move and accept with:

$$A(\mathbf{r_i} \rightarrow \mathbf{r_i}') = \min\left[1, \frac{|\Psi_T(\mathbf{r_i'})|^2}{|\Psi_T(\mathbf{r_i})|^2}\right] \quad (3)$$

- Compute E_L as per Eq. 1 and 2
- Repeat for many combinations of a_1 and a_2

VMC Algorithm

1. Initialize box of particles: L, N, a_1, a_2
2. For each particle, i

- Propose move from $\mathbf{r}$ to $\mathbf{r_i'} = \mathbf{r_i} + \xi \frac{\mathbf{L}}{\mathbf{2}}$
- Compute weight of move and accept with:

$$A(\mathbf{r_i} \rightarrow \mathbf{r_i}') = \min\left[1, \frac{|\Psi_T(\mathbf{r_i'})|^2}{|\Psi_T(\mathbf{r_i})|^2}\right] \quad (3)$$

- Compute E_L as per Eq. 1 and 2
- Repeat for many combinations of a_1 and a_2

VMC Algorithm

1. Initialize box of particles: L, N, a_1, a_2
2. For each particle, i

- Propose move from $\mathbf{r}$ to $\mathbf{r_i'} = \mathbf{r_i} + \xi\frac{\mathbf{L}}{\mathbf{2}}$
- Compute weight of move and accept with:

$$A(\mathbf{r_i} \rightarrow \mathbf{r_i}') = \min\left[1, \frac{|\Psi_T(\mathbf{r_i'})|^2}{|\Psi_T(\mathbf{r_i})|^2}\right] \quad (3)$$

- Compute E_L as per Eq. 1 and 2
- Repeat for many combinations of a_1 and a_2

Implementation

- Use C++'s object-oriented language
- Benefits: Code more "ideal", main code only a few lines long
- Created two classes:

1. Particles: Have a position and a mass
2. Configuration: A box of particles w/ PBC's

- Configuration contains subroutines for moving all particles, returning wave functions, calculating E_L, etc.

Implementation

- Use C++'s object-oriented language
- Benefits: Code more "ideal", main code only a few lines long
- Created two classes:

1. Particles: Have a position and a mass
2. Configuration: A box of particles w/ PBC's

- Configuration contains subroutines for moving all particles, returning wave functions, calculating E_L, etc.

Implementation

- Use C++'s object-oriented language
- Benefits: Code more "ideal", main code only a few lines long
- Created two classes:

1. Particles: Have a position and a mass
2. Configuration: A box of particles w/ PBC's

- Configuration contains subroutines for moving all particles, returning wave functions, calculating E_L, etc.

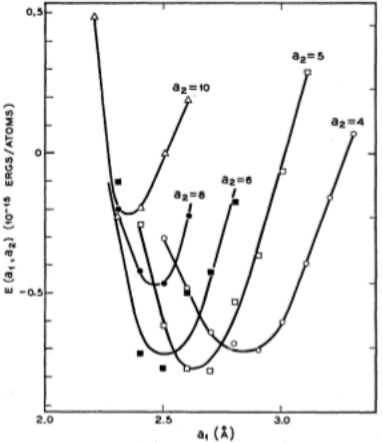
Implementation

- Use C++'s object-oriented language
- Benefits: Code more "ideal", main code only a few lines long
- Created two classes:

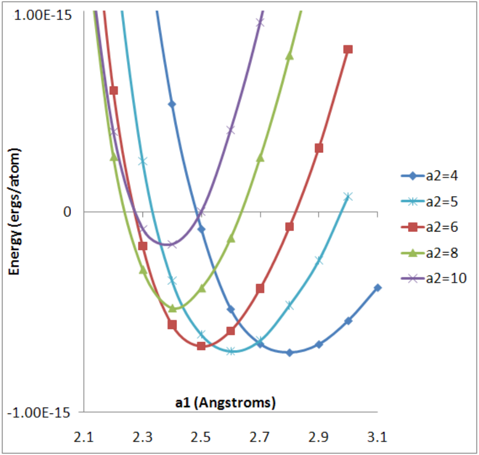
1. Particles: Have a position and a mass
2. Configuration: A box of particles w/ PBC's

- Configuration contains subroutines for moving all particles, returning wave functions, calculating E_L, etc.

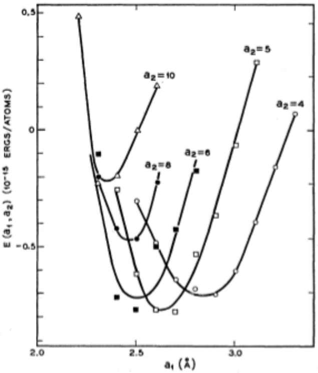
Results

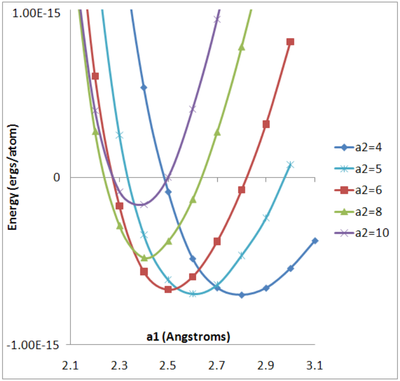


Results


1.00E-15
0
-1.00E-15
Energy (ergs/atom)
a1 (Angstroms)
2.1
2.3
2.5
2.7
2.9
3.1
a2=4
a2=5
a2=6
a2=8
a2=10

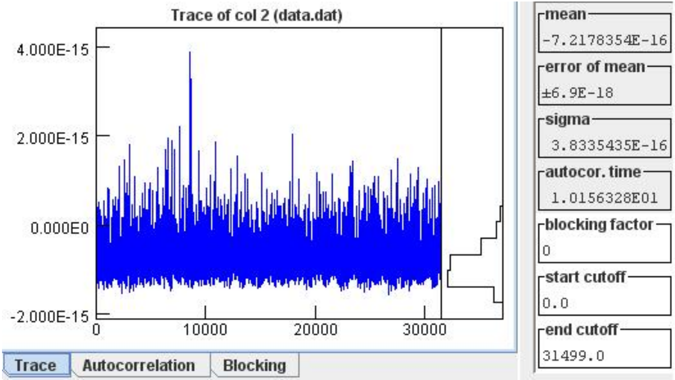
Results


0.5
0
-0.5
2.0
2.5
3.0
$a_2=10$
$a_2=5$
$a_2=4$
$a_2=6$
$a_2=8$
$E(a_1,a_2)$ $(10^{-15}$ ERGS/ATOMS)
a_1 (Å)


1.00E-15
0
-1.00E-15
Energy (ergs/atom)
a1 (Angstroms)
2.1
2.3
2.5
2.7
2.9
3.1
a2=4
a2=5
a2=6
a2=8
a2=10

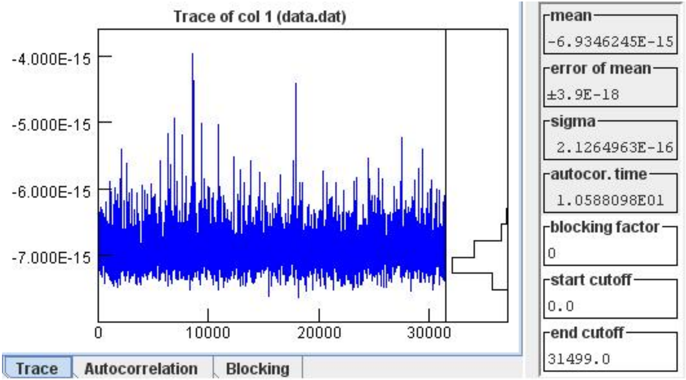
Results

- Using Eq.1 of calculating E_L
- Using Eq.2 of calculating E_L



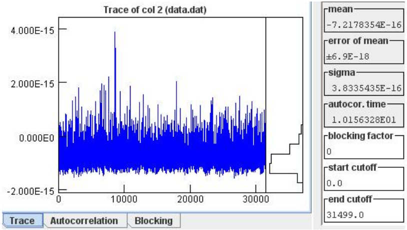
Results

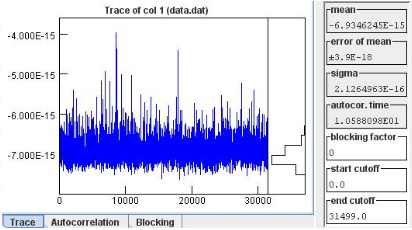
- Using Eq.1 of calculating E_L
- Using Eq.2 of calculating E_L



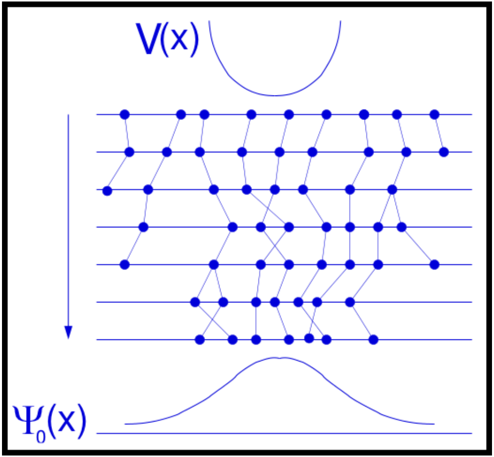
Results

- Using Eq.1 of calculating E_L
- Using Eq.2 of calculating E_L





DMC Overview



DMC Overview

- In the imaginary time transform ($it \rightarrow \tau$)

$$| \Psi\rangle(\tau + \delta\tau) = \sum c_i e^{\epsilon_i \delta\tau} \mid \psi_i\rangle, \quad (4)$$

- In imaginary time, energy states decay, not oscillate

$$\lim_{\tau\rightarrow\infty} \Psi(\mathbf{R}, \tau) = c_0 e^{\epsilon_0 \tau} \mid \psi_0\rangle \quad (5)$$

- Using $\Psi(\mathbf{R})$, get diffusion equation for behavior with diffusion and branching
- Using $f(\mathbf{R}, \tau) = \Psi_G(\mathbf{R})\Psi(\mathbf{R}, \tau)$, we also get "Drift" Term

$$\frac{\partial f(\mathbf{R}, \tau)}{\partial \tau} = \left[\sum_i -\frac{1}{2}\nabla_i^2 f(\mathbf{R}, \tau)\right] - \nabla \cdot \left[\frac{\nabla \psi_G(\mathbf{R})}{\psi_G(\mathbf{R})} f(\mathbf{R}, \tau)\right] + (E_L(\mathbf{R}) - E_T) f(\mathbf{R}, \tau), \quad (6)$$

DMC Overview

- Solution for this is a Green's function, $G(\mathbf{R}', \mathbf{R}; \tau)$
- From Trotter's theorem, for $\tau \to 0$, we can break up diffusion equation
- Solve approximately for $G(\mathbf{R}', \mathbf{R}; \tau)$

$$G(\mathbf{r}', \mathbf{r}; \tau) \sim Nexp(-\frac{(\mathbf{R}' - \mathbf{R} - \mathbf{V}(\mathbf{R})\tau)^2}{2\tau} exp(-(E_L(\mathbf{R}) + E_L(\mathbf{R}'))\frac{\tau}{2} \quad (7)$$

- And use this as a weight for moves:

$$W(\mathbf{R}', \mathbf{R}) = \frac{|\Psi_G(\mathbf{R}')|^2 G(\mathbf{R}', \mathbf{R}; \tau)}{|\Psi_G(\mathbf{R})|^2 G(\mathbf{R}, \mathbf{R}'; \tau)}, \quad (8)$$

DMC Algorithm

1. Initialize Ensemble containing many Configurations of Particles
2. For each Configuration, j
   - For each particle i: $\mathbf{r}'_i = \mathbf{r}_i + \tau \mathbf{V}(\mathbf{r}_i) + \eta$
   - Compute weight of move and accept with:

$$W(\mathbf{R}', \mathbf{R}) = \frac{|\Psi_G(\mathbf{R}')|^2 G(\mathbf{R}', \mathbf{R}; \tau)}{|\Psi_G(\mathbf{R})|^2 G(\mathbf{R}, \mathbf{R}'; \tau)}, \tag{9}$$

3. Calculate branching probability:
   $P_B = \exp\left(-\tau(\frac{E_L(R')+E_L(R)}{2} - E_T)\right)$
   Branch n copies with $n = floor(P_B + u)$ $u \in [0, 1)$

DMC Algorithm

1. Initialize Ensemble containing many Configurations of Particles
2. For each Configuration, j

- For each particle i: $\mathbf{r}'_i = \mathbf{r}_i + \tau \mathbf{V}(\mathbf{r}_i) + \eta$
- Compute weight of move and accept with:

$$W(\mathbf{R}', \mathbf{R}) = \frac{|\Psi_G(\mathbf{R}')|^2 G(\mathbf{R}', \mathbf{R}; \tau)}{|\Psi_G(\mathbf{R})|^2 G(\mathbf{R}, \mathbf{R}'; \tau)}, \quad (9)$$

3. Calculate branching probability:
$P_B = \exp\left(-\tau\left(\frac{E_L(R') + E_L(R)}{2} - E_T\right)\right)$
Branch n copies with $n = floor(P_B + u)$ $u \in [0, 1)$

DMC Algorithm

1. Initialize Ensemble containing many Configurations of Particles
2. For each Configuration, j

- For each particle i: $\mathbf{r'}_i = \mathbf{r}_i + \tau \mathbf{V}(\mathbf{r}_i) + \eta$
- Compute weight of move and accept with:

$$W(\mathbf{R'}, \mathbf{R}) = \frac{|\Psi_G(\mathbf{R'})|^2 G(\mathbf{R'}, \mathbf{R}; \tau)}{|\Psi_G(\mathbf{R})|^2 G(\mathbf{R}, \mathbf{R'}; \tau)}, \tag{9}$$

3. Calculate branching probability:
$P_B = \exp\left(-\tau(\frac{E_L(R')+E_L(R)}{2} - E_T)\right)$
Branch n copies with $n = floor(P_B + u)$ $u \in [0, 1)$

DMC Algorithm

4. Compute Average E_L over Configurations
5. After $O(100 - 1000)$ Configuration moves:
   - $E_T = \langle E_L \rangle$
   - Randomly branch / destroy Configurations to normalize
6. Repeat

DMC Algorithm

4. Compute Average E_L over Configurations
5. After $O(100 - 1000)$ Configuration moves:
   - $E_T = \langle E_L \rangle$
   - Randomly branch / destroy Configurations to normalize
6. Repeat

DMC Algorithm

4. Compute Average E_L over Configurations
5. After $O(100 - 1000)$ Configuration moves:
   - $E_T = \langle E_L \rangle$
   - Randomly branch / destroy Configurations to normalize
6. Repeat

DMC Algorithm

4. Compute Average E_L over Configurations
5. After $O(100 - 1000)$ Configuration moves:
   - $E_T = \langle E_L \rangle$
   - Randomly branch / destroy Configurations to normalize
6. Repeat

DMC Implementation

- Created three classes:

1. Particles: Have a position and a mass
2. Configuration: A box of particles w/ PBC's
3. Ensemble: A set of configurations

- Configuration contains subroutines for:

1. Moving all particles
2. Returning wave functions, Green's functions
3. Calculating E_L

- Ensemble contains subroutines for :

1. Branching, destroying configurations
2. Measuring average E_L over all configurations
3. Adjusting E_T

DMC Implementation

- Created three classes:

1. Particles: Have a position and a mass
2. Configuration: A box of particles w/ PBC's
3. Ensemble: A set of configurations

- Configuration contains subroutines for:

1. Moving all particles
2. Returning wave functions, Green's functions
3. Calculating E_L

- Ensemble contains subroutines for :

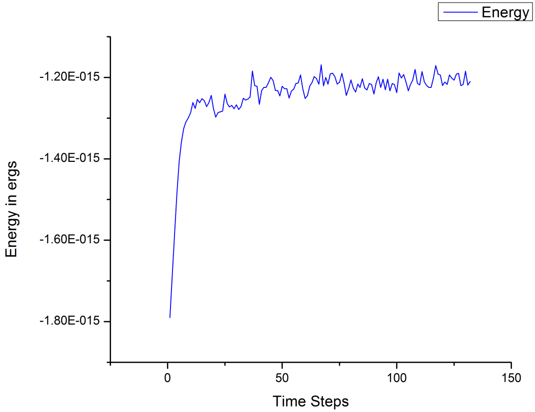
1. Branching, destroying configurations
2. Measuring average E_L over all configurations
3. Adjusting E_T

DMC Implementation

- Created three classes:
  1. Particles: Have a position and a mass
  2. Configuration: A box of particles w/ PBC's
  3. Ensemble: A set of configurations
- Configuration contains subroutines for:
  1. Moving all particles
  2. Returning wave functions, Green's functions
  3. Calculating E_L
- Ensemble contains subroutines for :
  1. Branching, destroying configurations
  2. Measuring average E_L over all configurations
  3. Adjusting E_T

Preliminary Results

Preliminary Results

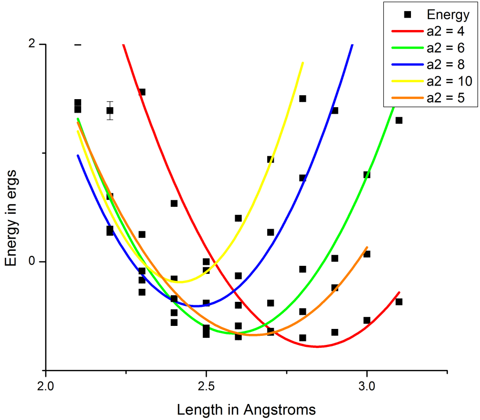


Better Wave Functions

Boronat's Jastrow Trial Wave Function gave us odd results, but could be revisited

$$\Psi_T = \prod_{i<j} \exp\left[-\frac{1}{2}(\frac{b}{r_{ij}})^5 - \frac{L}{2}\exp\left(-(\frac{r_{ij}-\lambda}{\Lambda})^2\right)\right]$$

Thank You


Energy
a2 = 4
a2 = 6
a2 = 8
a2 = 10
a2 = 5
2
0
2.0
2.5
3.0
Energy in ergs
Length in Angstroms

W. L. McMillan
Ground state of liquid He^4..
Phys. Rev., 138(2A):A442-A451, Apr 1965

J. Boronat and J. Casulleras
Monte carlo analysis of an interatomic potential for He.
Phys. Rev. B, 49(13):8920-8930, Apr 1994

R. A. Aziz, F. R. W. McCour, and C. C. .K. Wong
A new determination of the ground state interatomic potential for He_2.
Mol. Phys., 61:1487, 1987