Introduction to quantum Monte Carlo methods

Part 1

Claudia Filippi and Ravindra Shinde

MESA+ Institute for Nanotechnology, Universiteit Twente, The Netherlands

Anthony Scemama and Vijay Gopal Chilkuri

Laboratoire de Chimie et Physique Quantique, CNRS, Toulouse, France

Targeting Real Chemical Accuracy at the Exascale project has received funding from the European Union Horizoon 2020 research and innovation programme under Grant Agreement **No. 952165.**

Monte Carlo methods

Approaches which make repeated use of random numbers:

- to simulate truly stochastic events
- to solve deterministic problems using probabilities

Very important class of methods in statistical mechanics

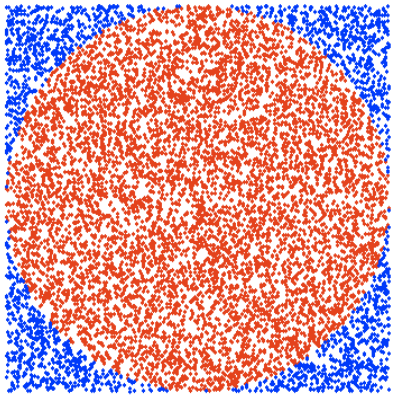
$\rightarrow$ Sampling Boltzmann distribution

Computation of averages (integrals in many dimensions)

For quantum mechanical simulations $\rightarrow$ Quantum Monte Carlo

A simple example of a Monte Carlo simulation

Basic idea of Monte Carlo through the "dartboard method"



$\rightarrow$ Throw darts, compute A_{circle}, compute π

Throw darts which land randomly within the square

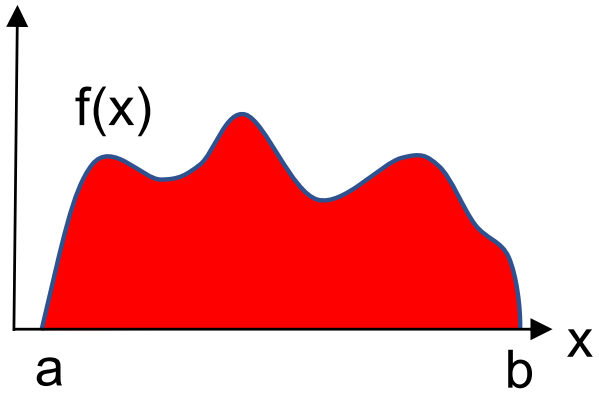
$$\frac{\#\text{ hits inside circle}}{\#\text{ hits inside the square}} = \frac{A_{\text{circle}}}{A_{\text{square}}} = \frac{\pi}{4}$$

$\uparrow$

many, many hits

Monte Carlo integration

We want to compute the integral of f(x) in the interval $[a, b]$



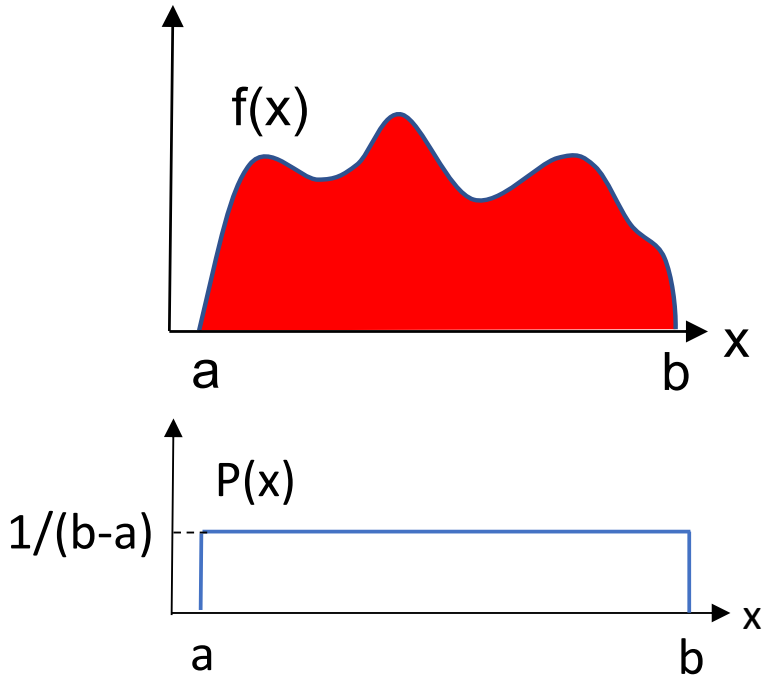
$$
\begin{aligned}
I &= \int_a^b f(x)\,\mathrm{d}x = (b-a)\int_a^b f(x)\frac{1}{b-a}\,\mathrm{d}x \\
&= (b-a)\langle f\rangle_{[a,b]}
\end{aligned}
$$

where $\langle f\rangle_{[a,b]}$ is the average of the function in the range $[a, b]$

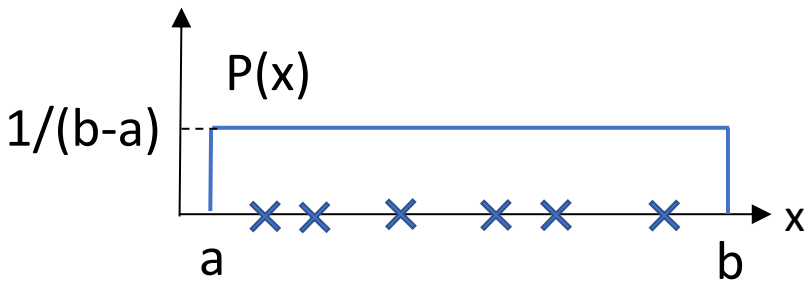
Monte Carlo integration

$$\langle f \rangle_{[a,b]} = \int_a^b f(x) \frac{1}{b-a} \, \mathrm{d}x$$

$$= \int_a^b f(x) P(x) \, \mathrm{d}x$$



Draw M random numbers distributed unformely in $[a, b]$



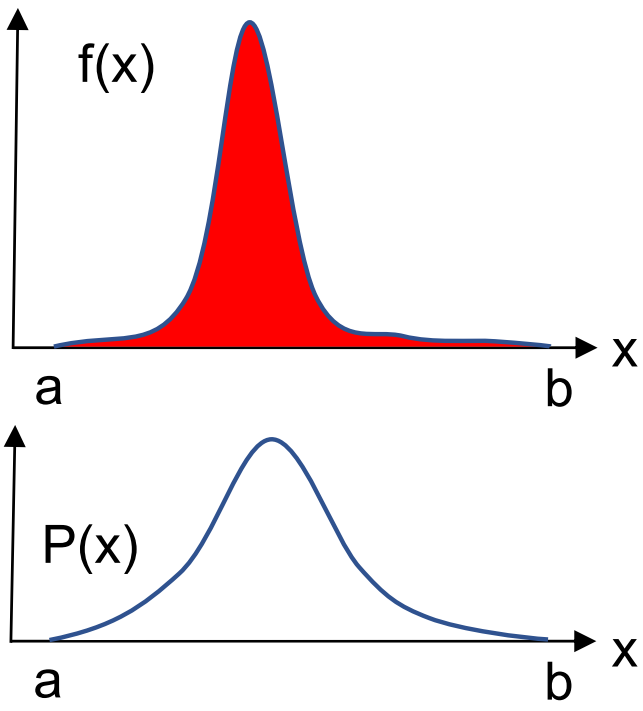
$\longrightarrow$

$$\boxed{\langle f \rangle_{[a,b]} \approx \frac{1}{M} \sum_{i=1}^{M} f(x_i)}$$

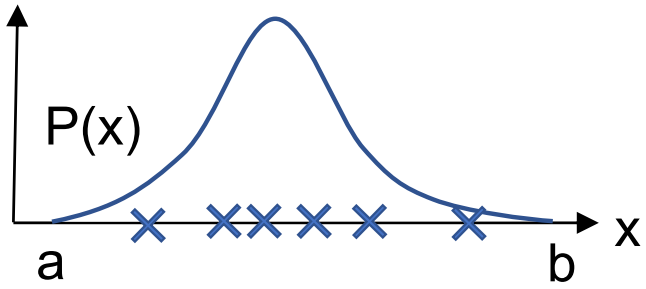
A less uniform function

$$I = \int_a^b f(x)\,\mathrm{d}x$$

$$= \int_a^b \frac{f(x)}{P(x)} P(x)\,\mathrm{d}x$$



Draw M random numbers distributed as $P(x)$



$\longrightarrow$ $$\boxed{I \approx \frac{1}{M} \sum_{i=1}^{M} \frac{f(x_i)}{P(x_i)}}$$

Monte Carlo integration in a nutshell

We want to compute $\boxed{\langle A \rangle = \int_a^b A(x)P(x)}$

with $\boxed{P(x) \geq 0 \text{ and } \int_a^b P(x) = 1}$ $\leftarrow$ a probability density!

Monte Carlo $\rightarrow$ Sample $\{x_1, \ldots, x_M\}$ from $P(x)$

$$\text{Estimate} \quad \langle A \rangle \approx \frac{1}{M} \sum_{i=1}^{M} A(x_i)$$

Statistical physics: $P(x) = \dfrac{e^{-\beta E(x)}}{Z}$, the Boltzman distribution

Quantum chemical simulations

- Density functional theory methods

  Large systems but approximate exchange/correlation

- Quantum chemistry post-Hartree-Fock methods

  Accurate on small-medium systems

  $\rightarrow$ Jungle of approaches: CI, MCSCF, CC, CASPT2 ...

- Quantum Monte Carlo techniques

  Stochastic solution of the Schrödinger equation

  Accurate correlated calculations for medium-large systems

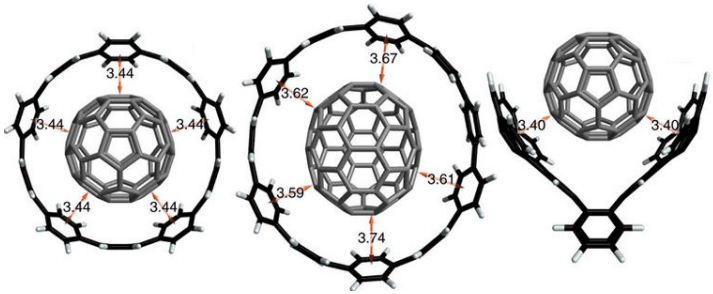
Some general words about quantum Monte Carlo methods

Stochastically solve interacting Schrödinger equation

Why (real-space) quantum Monte Carlo?

- Favorable scaling $\rightarrow$ Energy is $\boxed{O(N^4)}$
- Flexibility in choice of functional form of wave function
- Easy parallelization
- Among most accurate calculations for medium-large systems

  Routinely, molecules of up to 100 (mainly 1st/2nd-row) atoms



upto $C_{136}H_{44}$ (Alfé 2017)

A different way of writing the expectation values

Consider the expectation value of the Hamiltonian on Ψ

$$
\begin{aligned}
E_V &= \frac{\langle \Psi | \mathcal{H} | \Psi \rangle}{\langle \Psi | \Psi \rangle} = \frac{\int \mathrm{d}\mathbf{R}\, \Psi^*(\mathbf{R}) \mathcal{H} \Psi(\mathbf{R})}{\int \mathrm{d}\mathbf{R}\, \Psi^*(\mathbf{R}) \Psi(\mathbf{R})} \geq E_0 \\
&= \int \mathrm{d}\mathbf{R}\, \frac{\mathcal{H}\Psi(\mathbf{R})}{\Psi(\mathbf{R})} \boxed{\frac{|\Psi(\mathbf{R})|^2}{\int \mathrm{d}\mathbf{R} |\Psi(\mathbf{R})|^2}} \\
&= \int \mathrm{d}\mathbf{R}\, E_{\mathrm{L}}(\mathbf{R})\, P(\mathbf{R}) = \langle E_{\mathrm{L}}(\mathbf{R}) \rangle_P
\end{aligned}
$$

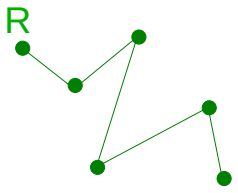
$P(\mathbf{R})$ is a probability density and $E_{\mathrm{L}}(\mathbf{R}) = \dfrac{\mathcal{H}\Psi(\mathbf{R})}{\Psi(\mathbf{R})}$ the local energy

Variational Monte Carlo: a random walk of the electrons

Use Monte Carlo integration to compute expectation values

- Sample $\mathbf{R}$ from $P(\mathbf{R})$ using Metropolis algorithm
- Average local energy $E_{\mathrm{L}}(\mathbf{R}) = \dfrac{\mathcal{H}\Psi(\mathbf{R})}{\Psi(\mathbf{R})}$ to obtain E_V as

$$\boxed{E_V = \langle E_{\mathrm{L}}(\mathbf{R})\rangle_P \approx \frac{1}{M}\sum_{i=1}^{M} E_{\mathrm{L}}(\mathbf{R}_i)}$$

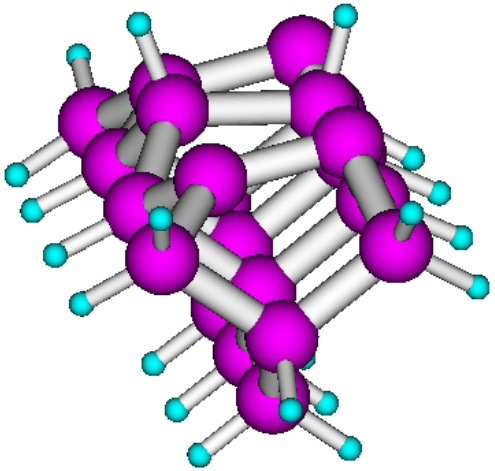


Random walk in 3N dimensions, $\mathbf{R} = (\mathbf{r}_1, \ldots, \mathbf{r}_N)$

Just a $\boxed{\text{trick}}$ to evaluate integrals in many dimensions

Is it really "just" a trick?

$Si_{21}H_{22}$



Number of electrons $4 \times 21 + 22 = 106$

Number of dimensions $3 \times 106 = \boxed{318}$

Integral on a grid with 10 points/dimension $\rightarrow 10^{318}$ points!

MC is a powerful trick $\Rightarrow$ Freedom in form of the wave function Ψ

Monte Carlo integration

We want to compute an integral

$$E_V = \int \mathrm{d}\mathbf{R} E_{\mathrm{L}}(\mathbf{R}) P(\mathbf{R})$$

We sample $P(\mathbf{R}) \rightarrow$ $\boxed{E_V = \langle E_{\mathrm{L}}(\mathbf{R}) \rangle_P \approx \frac{1}{M} \sum_{i=1}^{M} E_{\mathrm{L}}(\mathbf{R}_i)}$

– Does the trick always work?

– How efficient is it?

The Central Limit Theorem

Probability density P and function f with finite mean and variance

$$\boxed{\mu} = \int \mathrm{d}x\, f(x) P(x) \qquad \boxed{\sigma^2} = \int \mathrm{d}x\, (f(x) - \mu)^2 P(x)$$

Sample M independent random variables $x_1, \ldots, x_M$ from $P(x)$

Define
$$\boxed{F_M = \frac{1}{M} \sum_{i=1}^{M} f(x_i)}$$

As M increases, F_M is normally distributed as $\frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/2\sigma_M^2}$

with a mean $\boxed{\mu}$ and variance $\boxed{\sigma_M^2 = \sigma^2/\mathrm{M}}$

$\rightarrow$ $\boxed{\text{Irrespective}}$ of the original probability density function

Monte Carlo versus deterministic integration

Integration error ϵ using $M_{\rm int}$ integration/$M_{\rm MC}$ Monte Carlo points

– Monte Carlo methods

$$\epsilon \propto \frac{1}{\sqrt{M_{\rm MC}}}$$ independent on dimension !

It follows from Central Limit Theorem

$\rightarrow$ width of Gaussian decreases as $\dfrac{\sigma}{\sqrt{M_{\rm MC}}}$ for finite variance

– Deterministic integration methods

1-dim Simpson rule: $\epsilon \propto \dfrac{1}{M_{\rm int}^4}$

d-dim Simpson rule: $\epsilon \propto \dfrac{1}{M_{\rm int}^{4/d}}$

Scaling with number of electrons

Roughly, Monte Carlo integration advantageous if $d > 8$

... for many-body wave functions $d = 3N_{\mathrm{elec}}$!

– Simpson rule integration ($\mathrm{M}_{\mathrm{int}}$ integration points)

$$\epsilon = \frac{c}{M_{\mathrm{int}}^{4/d}} = \frac{c}{M_{\mathrm{int}}^{4/3N_{\mathrm{elec}}}} \quad \Rightarrow \quad M_{\mathrm{int}} = \left(\frac{c}{\epsilon}\right)^{3N_{\mathrm{elec}}/4}$$

Exponential

– Monte Carlo integration (M_{MC} Monte Carlo samples)

$$\epsilon = \frac{\sigma}{\sqrt{M_{\mathrm{MC}}}} = c\sqrt{\frac{N_{\mathrm{elec}}}{M_{\mathrm{MC}}}} \quad \Rightarrow \quad M_{\mathrm{MC}} = \left(\frac{c}{\epsilon}\right)^2 N_{\mathrm{elec}}$$

Linear

Summary of variational Monte Carlo

Expectation value of the Hamiltonian on Ψ

$$E_V = \frac{\langle\Psi|\mathcal{H}|\Psi\rangle}{\langle\Psi|\Psi\rangle} = \int \mathrm{d}\mathbf{R}\, \frac{\mathcal{H}\Psi(\mathbf{R})}{\Psi(\mathbf{R})} \frac{|\Psi(\mathbf{R})|^2}{\int \mathrm{d}\mathbf{R}|\Psi(\mathbf{R})|^2} = \int \mathrm{d}\mathbf{R}\, E_{\mathrm{L}}(\mathbf{R})\, P(\mathbf{R})$$

$$\begin{aligned} E_V &= \int \mathrm{d}\mathbf{R}\, E_{\mathrm{L}}(\mathbf{R})\, P(\mathbf{R}) \\ \sigma^2 &= \int \mathrm{d}\mathbf{R}(E_{\mathrm{L}}(\mathbf{R}) - E_V)^2 P(\mathbf{R}) \end{aligned}$$

Estimate E_V and σ from M independent samples as

$$\begin{aligned} \bar{E}_V &= \frac{1}{M}\sum_{i=1}^{M} E_{\mathrm{L}}(\mathbf{R}_i) \\ \bar{\sigma}^2 &= \frac{1}{M-1}\sum_{i=1}^{M}(E_{\mathrm{L}}(\mathbf{R}_i) - \bar{E}_V)^2 \end{aligned}$$

Are there any conditions on many-body Ψ to be used in VMC?

Within VMC, we can use any "computable" wave function if

- Continuous, normalizable, proper symmetry
- Finite variance

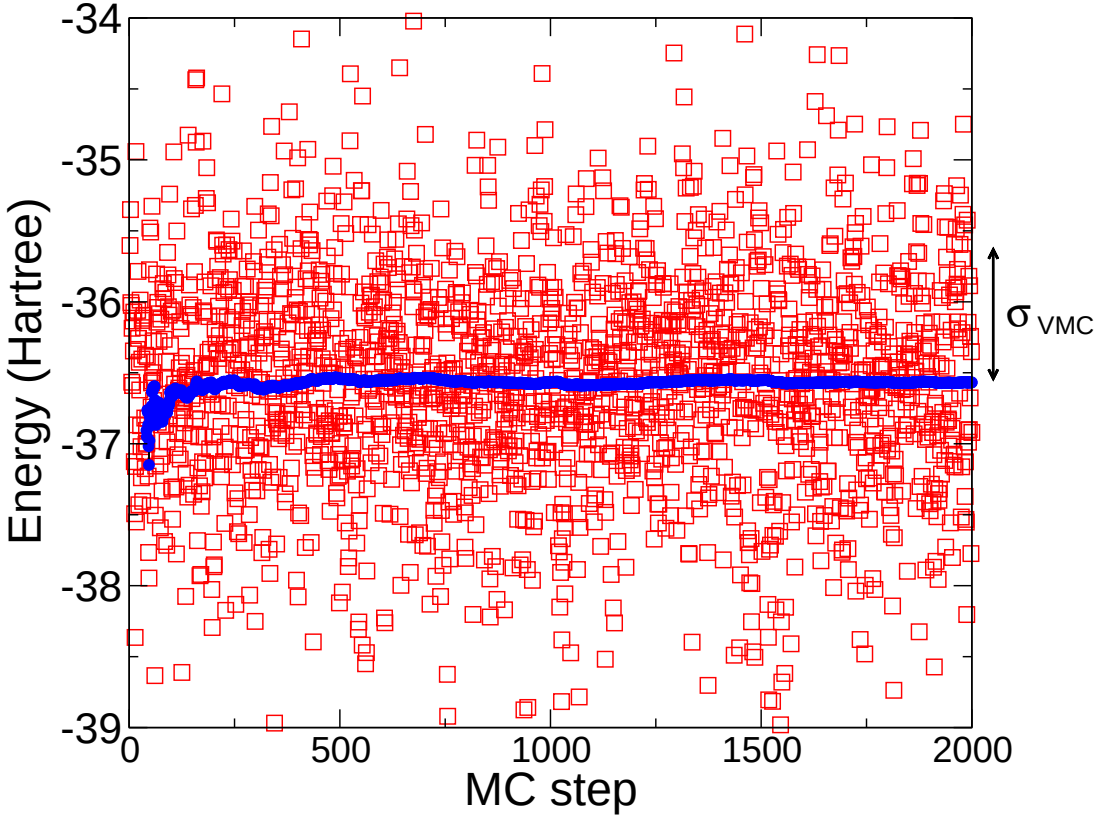
$$\sigma^2 = \frac{\langle \Psi | (\mathcal{H} - E_V)^2 | \Psi \rangle}{\langle \Psi | \Psi \rangle} = \langle (E_{\mathrm{L}}(\mathbf{R}) - E_V)^2 \rangle_P$$

since the Monte Carlo error goes as $\boxed{\mathrm{err}(E_V) \sim \frac{\sigma}{\sqrt{M}}}$

Zero variance principle: if $\Psi \to \Psi_0$, $E_{\mathrm{L}}(\mathbf{R})$ does not fluctuate

Typical VMC run

Example: Local energy and average energy of acetone (C_3H_6O)



$$E_{\mathrm{VMC}} = \langle E_{\mathrm{L}}(\mathbf{R}) \rangle_P = -36.542 \pm 0.001 \text{ Hartree } (40 \times 20000 \text{ steps})$$

$$\sigma_{\mathrm{VMC}} = \langle (E_{\mathrm{L}}(\mathbf{R}) - E_{\mathrm{VMC}})^2 \rangle_P = 0.90 \text{ Hartree}$$

Variational Monte Carlo: To do list

– Method to $\boxed{\text{sample}}$ distribution function $P(\mathbf{R}) = \dfrac{|\Psi(\mathbf{R})|^2}{\int \mathrm{d}\mathbf{R}|\Psi(\mathbf{R})|^2}$

$\rightarrow$ Obtain a set of $\{\mathbf{R}_1, \mathbf{R}_2, \ldots, \mathbf{R}_M\}$ distributed as $P(\mathbf{R})$

How? As in classical Monte Carlo with Metropolis algorithm!

– Build the wave function $\Psi(\mathbf{R})$. Which $\boxed{\text{functional form}}$?

Here, we spend most of our time, open topic of research

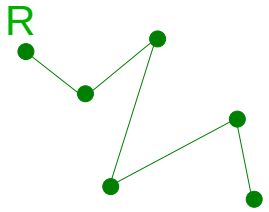
– Compute expectation values $\boxed{\dfrac{\langle\Psi|\mathcal{O}|\Psi\rangle}{\langle\Psi|\Psi\rangle}}$

Reformulate them to reduce fluctuations, open topic of research

How do we sample $P(\mathbf{R})$?

Generate a Markov chain

$$\ldots \xrightarrow{M} \mathbf{R} \xrightarrow{M} \mathbf{R}' \xrightarrow{M} \mathbf{R}'' \xrightarrow{M} \ldots$$



Construct $M(\mathbf{R}_\mathrm{f}|\mathbf{R}_\mathrm{i})$ as probability for transition $\mathbf{R}_\mathrm{i} \to \mathbf{R}_\mathrm{f}$ so that

- $M(\mathbf{R}_\mathrm{f}|\mathbf{R}_\mathrm{i}) \geq 0$ and $\int d\mathbf{R}_\mathrm{f} M(\mathbf{R}_\mathrm{f}|\mathbf{R}_\mathrm{i}) = 1$ (stochastic)
- If we start from an arbitrary distribution P_init, we evolve to P

  $\to$ Impose stationarity condition

Constructing M

To sample P, use M which satisfies stationarity condition:

$$\int \mathrm{d}\mathbf{R}_\mathrm{i}\, M(\mathbf{R}_\mathrm{f}|\mathbf{R}_\mathrm{i})\, P(\mathbf{R}_\mathrm{i}) = P(\mathbf{R}_\mathrm{f}) \quad \forall\, \mathbf{R}_\mathrm{f}$$

- Stationarity condition

  $\Rightarrow$ If we start with P, we continue to sample P

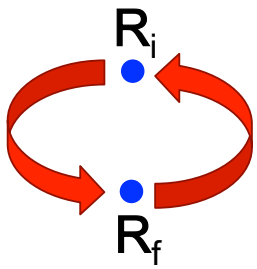
- Stationarity condition + stochastic property of M + ergodicity

  $\Rightarrow$ Any initial distribution will evolve to P

More stringent condition

In practice, we impose detailed balance condition

$$M(\mathbf{R}_\mathrm{f}|\mathbf{R}_\mathrm{i})\, P(\mathbf{R}_\mathrm{i}) = M(\mathbf{R}_\mathrm{i}|\mathbf{R}_\mathrm{f})\, P(\mathbf{R}_\mathrm{f})$$



Stationarity condition can be obtained by summing over $\mathbf{R}_\mathrm{i}$

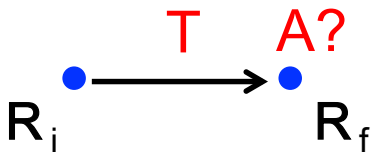
$$\int \mathrm{d}\mathbf{R}_\mathrm{i} M(\mathbf{R}_\mathrm{f}|\mathbf{R}_\mathrm{i})\, P(\mathbf{R}_\mathrm{i}) = \underbrace{\int \mathrm{d}\mathbf{R}_\mathrm{i} M(\mathbf{R}_\mathrm{i}|\mathbf{R}_\mathrm{f})}_{1}\, P(\mathbf{R}_\mathrm{f}) = P(\mathbf{R}_\mathrm{f})$$

Detailed balance is a sufficient but not necessary condition

How do we construct the transition matrix P in practice?

Metropolis method $\rightarrow$ Write M as proposal T $\times$ acceptance A

$$M(\mathbf{R}_\mathrm{f}|\mathbf{R}_\mathrm{i}) = A(\mathbf{R}_\mathrm{f}|\mathbf{R}_\mathrm{i})\ T(\mathbf{R}_\mathrm{f}|\mathbf{R}_\mathrm{i})$$



Let us rewrite the detailed balance condition

$$\begin{aligned}
M(\mathbf{R}_\mathrm{f}|\mathbf{R}_\mathrm{i})\ P(\mathbf{R}_\mathrm{i}) &= M(\mathbf{R}_\mathrm{i}|\mathbf{R}_\mathrm{f})\ P(\mathbf{R}_\mathrm{f}) \\
A(\mathbf{R}_\mathrm{f}|\mathbf{R}_\mathrm{i})\ T(\mathbf{R}_\mathrm{f}|\mathbf{R}_\mathrm{i})\ P(\mathbf{R}_\mathrm{i}) &= A(\mathbf{R}_\mathrm{i}|\mathbf{R}_\mathrm{f})\ T(\mathbf{R}_\mathrm{i}|\mathbf{R}_\mathrm{f})\ P(\mathbf{R}_\mathrm{f}) \\
\Rightarrow \quad \frac{A(\mathbf{R}_\mathrm{f}|\mathbf{R}_\mathrm{i})}{A(\mathbf{R}_\mathrm{i}|\mathbf{R}_\mathrm{f})} &= \frac{T(\mathbf{R}_\mathrm{i}|\mathbf{R}_\mathrm{f})\ P(\mathbf{R}_\mathrm{f})}{T(\mathbf{R}_\mathrm{f}|\mathbf{R}_\mathrm{i})\ P(\mathbf{R}_\mathrm{i})}
\end{aligned}$$

Choice of acceptance matrix A

Original choice by Metropolis *et al.* maximizes the acceptance

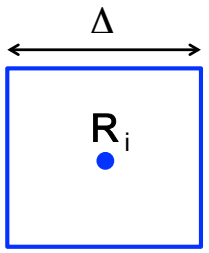
$$A(\mathbf{R}_\mathrm{f}|\mathbf{R}_\mathrm{i}) = \min\left\{1, \frac{T(\mathbf{R}_\mathrm{i}|\mathbf{R}_\mathrm{f})\ P(\mathbf{R}_\mathrm{f})}{T(\mathbf{R}_\mathrm{f}|\mathbf{R}_\mathrm{i})\ P(\mathbf{R}_\mathrm{i})}\right\}$$

Note: $P(\mathbf{R})$ does not have to be normalized

$\rightarrow$ For complicated Ψ we do not know the normalization!

$\rightarrow$ $P(\mathbf{R}) = |\Psi(\mathbf{R})|^2$

Original Metropolis method



Symmetric $T(\mathbf{R}_\mathrm{f}|\mathbf{R}_\mathrm{i}) = 1/\Delta^{3N} \quad \Rightarrow \quad A(\mathbf{R}_\mathrm{f}|\mathbf{R}_\mathrm{i}) = \min\left\{1, \frac{P(\mathbf{R}_\mathrm{f})}{P(\mathbf{R}_\mathrm{i})}\right\}$

Better choices of proposal matrix T

Sequential correlation $\Rightarrow$ $M_{\rm eff} < M$ independent observations

$$M_{\rm eff} = \frac{M}{T_{\rm corr}}$$ with $T_{\rm corr}$ autocorrelation time of desired observable

Aim is to achieve fast evolution and reduce correlation times

Use **freedom** in choice of T: For example, use available trial Ψ

$$T(\mathbf{R}_{\rm f}|\mathbf{R}_{\rm i}) = \mathcal{N} \exp\left[-\frac{(\mathbf{R}_{\rm f} - \mathbf{R}_{\rm i} - \mathbf{V}(\mathbf{R}_{\rm i})\tau)^2}{2\tau}\right] \quad \text{with} \quad \mathbf{V}(\mathbf{R}_{\rm i}) = \frac{\nabla\Psi(\mathbf{R}_{\rm i})}{\Psi(\mathbf{R}_{\rm i})}$$

Acceptance and T_{corr} for the total energy E_V

Example: All-electron Be atom with simple wave function

Simple Metropolis

Δ	T_{corr}	$\bar{A}$
1.00	41	0.17
0.75	21	0.28
0.50	17	0.46
0.20	45	0.75

Drift-diffusion transition

τ	T_{corr}	$\bar{A}$
0.100	13	0.42
0.050	7	0.66
0.020	8	0.87
0.010	14	0.94

Generalized Metropolis algorithm

1. Choose distribution $P(\mathbf{R})$ and proposal matrix $T(\mathbf{R}_\mathrm{f}|\mathbf{R}_\mathrm{i})$
2. Initialize the configuration $\mathbf{R}_i$
3. Advance the configuration from $\mathbf{R}_i$ to $\mathbf{R}'$
   a) Sample $\mathbf{R}'$ from $T(\mathbf{R}'|\mathbf{R}_\mathrm{i})$.
   b) Calculate the ratio $p = \dfrac{T(\mathbf{R}_\mathrm{i}|\mathbf{R}')}{T(\mathbf{R}'|\mathbf{R}_\mathrm{i})}\dfrac{P(\mathbf{R}')}{P(\mathbf{R}_\mathrm{i})}$
   c) Accept or reject with probability p

      Pick a uniformly distributed random number $\chi \in [0, 1]$

      if $\chi < p$, move accepted $\rightarrow$ set $\mathbf{R}_\mathrm{f} = \mathbf{R}'$

      if $\chi > p$, move rejected $\rightarrow$ set $\mathbf{R}_\mathrm{f} = \mathbf{R}$
4. Throw away first κ configurations of equilibration time
5. Collect the averages

$\boxed{\text{Variational Monte Carlo} \rightarrow \text{Freedom in choice of } \Psi}$

Monte Carlo integration allows the use of complex and accurate Ψ

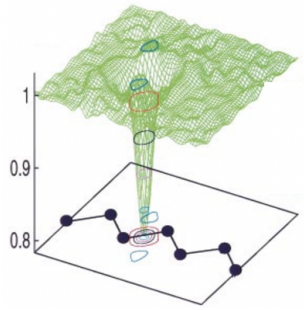
$\Rightarrow$ More $\boxed{\text{compact}}$ representation of Ψ than in quantum chemistry

$\Rightarrow$ $\boxed{\text{Beyond}}$ $c_0 D_{\mathrm{HF}} + c_1 D_1 + c_2 D_2 + \ldots$ $\boxed{\text{millions}}$ of determinants

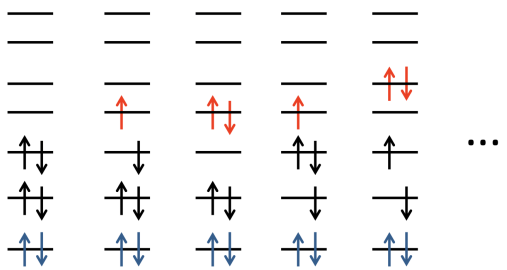
Jastrow-Slater wave function

Commonly employed compact Jastrow-Slater wave functions

$$\Psi(\mathbf{r}_1, \ldots, \mathbf{r}_N) = \mathcal{J}(\mathbf{r}_1, \ldots, \mathbf{r}_N) \times \sum_i c_i \, D_i(\mathbf{r}_1, \ldots, \mathbf{r}_N)$$



$\times$



$\boxed{\mathcal{J}}$ $\longrightarrow$ Jastrow correlation factor

– Explicit dependence on electron-electron distances $\boxed{r_{ij}}$

$\boxed{\sum c_i \, D_i}$ $\longrightarrow$ Determinants of single-particle orbitals

– $\boxed{\text{Few}}$ and not millions of determinants

Divergence in potential and behavior of the local energy

Consider two particles of masses m_i, m_j and charges q_i, q_j

Assume $r_{ij} \to 0$ while all other particles are well separated

Keep only diverging terms in $\dfrac{\mathcal{H}\Psi}{\Psi}$ and go to relative coordinates close to $\mathbf{r} = \mathbf{r}_{ij} = 0$

$$-\frac{1}{2\mu_{ij}}\frac{\nabla^2\Psi}{\Psi} + \mathcal{V}(r) \sim -\frac{1}{2\mu_{ij}}\frac{\Psi''}{\Psi} - \frac{1}{\mu_{ij}}\frac{1}{r}\frac{\Psi'}{\Psi} + \mathcal{V}(r)$$

$$\sim \boxed{-\frac{1}{\mu_{ij}}\frac{1}{r}\frac{\Psi'}{\Psi} + \mathcal{V}(r)}$$

where $\mu_{ij} = m_i\, m_j/(m_i + m_j)$

Divergence in potential and cusp conditions

Diverging terms in the local energy

$$-\frac{1}{\mu_{ij}}\frac{1}{r}\frac{\Psi'}{\Psi} + \mathcal{V}(r) = -\frac{1}{\mu_{ij}}\frac{1}{r}\frac{\Psi'}{\Psi} + \frac{q_i q_j}{r} = \text{ finite}$$

$\Rightarrow$ Ψ must satisfy Kato's cusp conditions:

$$\boxed{\left.\frac{\partial \hat{\Psi}}{\partial r_{ij}}\right|_{r_{ij}=0} = \mu_{ij} q_i\, q_j \Psi(r_{ij} = 0)}$$

where $\hat{\Psi}$ is a spherical average

Note: We assumed $\Psi(r_{ij} = 0) \neq 0$

Cusp conditions: example

The condition for the local energy to be finite at $r = 0$ is

$$\frac{\Psi'}{\Psi} = \mu_{ij} q_i \, q_j$$

- Electron-nucleus: $\mu = 1,\ q_i = 1,\ q_j = -Z \quad \Rightarrow \quad \boxed{\left.\frac{\Psi'}{\Psi}\right|_{r=0} = -Z}$
- Electron-electron: $\mu = \frac{1}{2},\ q_i = 1,\ q_j = 1 \quad \Rightarrow \quad \boxed{\left.\frac{\Psi'}{\Psi}\right|_{r=0} = 1/2}$

Cusp conditions and QMC wave functions

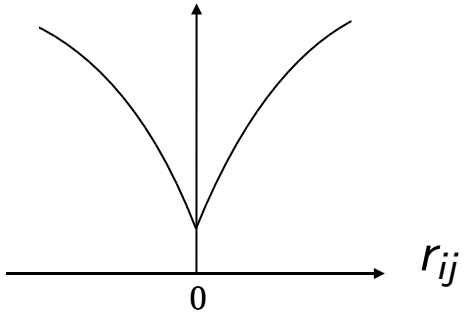
- Electron-electron cusps imposed through the Jastrow factor

  Example: Simple Jastrow factor

$$\mathcal{J}(r_{ij}) = \prod_{i<j} \exp\left\{ b_0 \, \frac{r_{ij}}{1 + b \, r_{ij}} \right\}$$

with $\quad b_0^{\uparrow\downarrow} = \dfrac{1}{2} \quad$ or $\quad b_0^{\uparrow\uparrow} = b_0^{\downarrow\downarrow} = \dfrac{1}{4}$

Imposes cusp conditions
+
keeps electrons apart

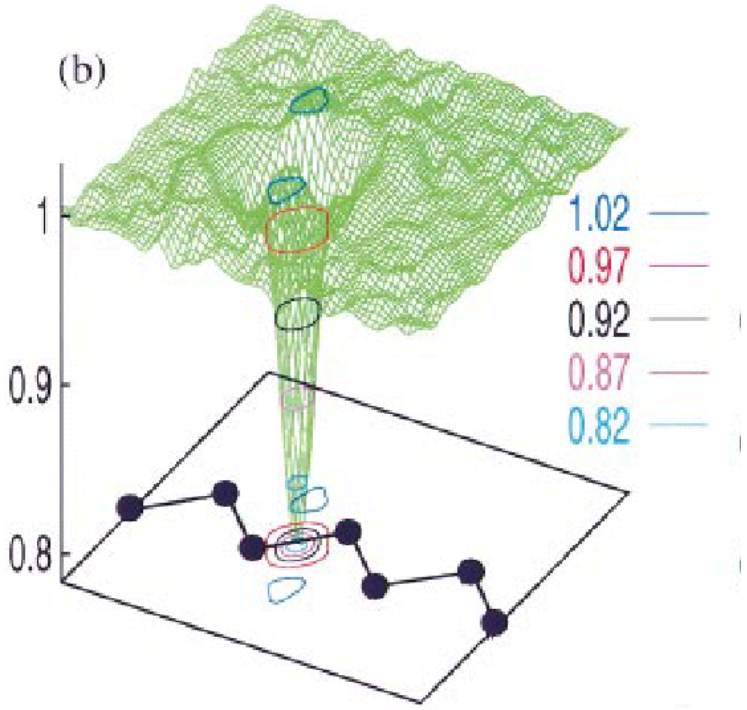


- Electron-nucleus cusps imposed through the determinantal part

The effect of the Jastrow factor

Pair correlation function for $\uparrow\downarrow$ electrons in the (110) plane of Si

$g_{\uparrow\downarrow}(\mathbf{r}, \mathbf{r}')$ with one electron is at the bond center



Hood *et al.* Phys. Rev. Lett. **78**, 3350 (1997)

Why should $\Psi_{\mathrm{QMC}} = \mathcal{J} D$ work?

Full wave-function	$\longrightarrow$	Factorized wave-function
Ψ		$\mathcal{J}\Phi$
$\downarrow$		$\downarrow$
Full Hamiltonian	$\longrightarrow$	Effective Hamiltonian
$\mathcal{H}$		$\mathcal{H}_{\mathrm{eff}}$

$$\boxed{\mathcal{H}\Psi = \mathrm{E}\Psi} \longrightarrow \mathcal{H}\mathcal{J}\Phi = \mathrm{E}\mathcal{J}\Phi \rightarrow \frac{\mathcal{H}\mathcal{J}}{\mathcal{J}}\Phi = \mathrm{E}\Phi$$

$$\boxed{\mathcal{H}_{\mathrm{eff}}\Phi = \mathrm{E}\Phi}$$

$\mathcal{H}_{\mathrm{eff}}$ weaker Hamiltonian than $\mathcal{H}$

$\Rightarrow$ $\Phi \approx$ non-interacting wave function D

$\Rightarrow$ Quantum Monte Carlo wave function $\Psi = \mathcal{J} D$

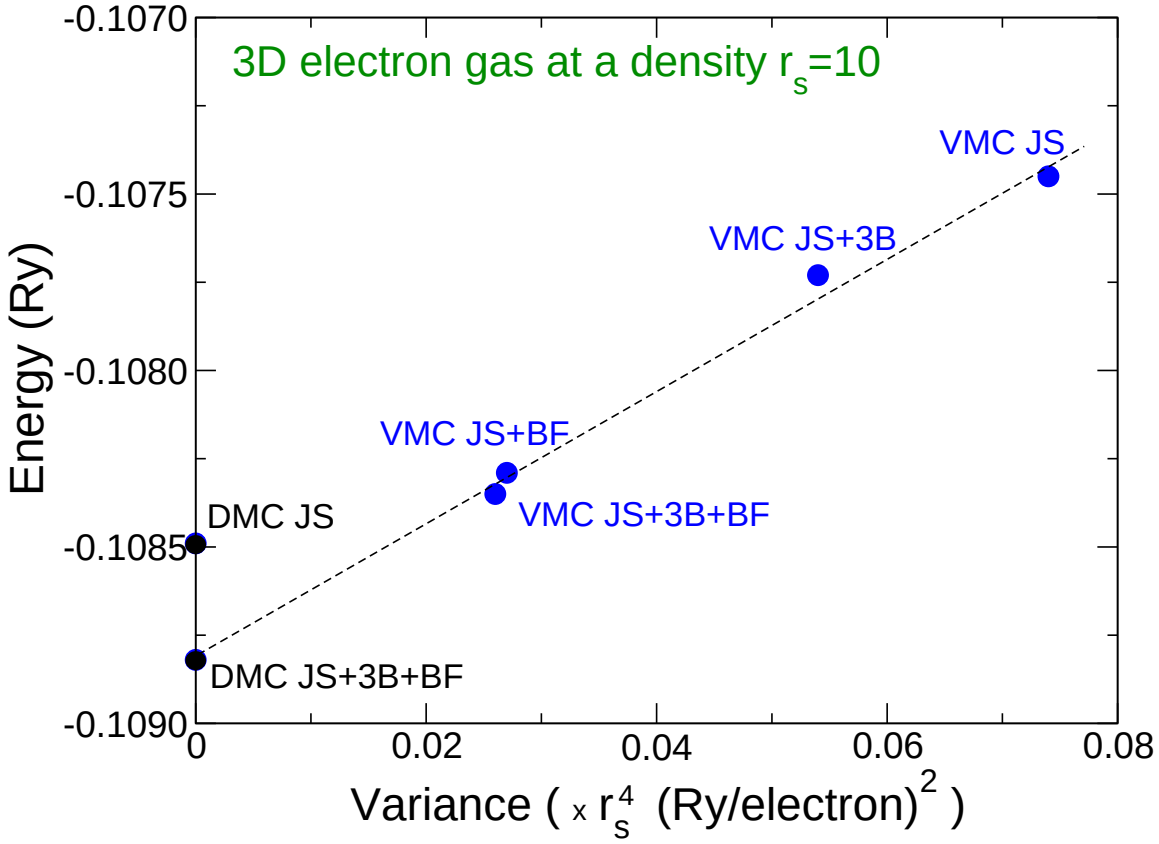
Beyond VMC?

Removing or reducing wave function bias?

$\Rightarrow$ Projection Monte Carlo methods

Why going beyond VMC?

Dependence of VMC from wave function Ψ



Kwon, Ceperley, Martin, Phys. Rev. B **58**, 6800 (1998)

Why going beyond VMC?

What goes in, comes out! Can we remove wave function bias?

Projector (diffusion) Monte Carlo method

- Construct an operator which inverts spectrum of $\mathcal{H}$

  Diffusion Monte Carlo $\rightarrow$ $e^{-\tau(\mathcal{H}-E_{\mathrm{ref}})}$

- Use it to stochastically project the ground state of $\mathcal{H}$

Diffusion Monte Carlo

Consider initial guess $\Psi^{(0)}$ and repeatedly apply projection operator

$$\boxed{\Psi^{(n)} = e^{-\tau(\mathcal{H}-E_{\mathrm{ref}})}\Psi^{(n-1)}}$$

Expand $\Psi^{(0)}$ on the eigenstates Ψ_i with energies E_i of $\mathcal{H}$

$$\Psi^{(n)} = e^{-n\tau(\mathcal{H}-E_{\mathrm{ref}})}\Psi^{(0)} = \sum_i \Psi_i \langle\Psi_i|\Psi^{(0)}\rangle e^{-n\tau(E_i-E_{\mathrm{ref}})}$$

and obtain in the limit of $n \to \infty$

$$\lim_{n\to\infty} \Psi^{(n)} = \Psi_0 \langle\Psi_0|\Psi^{(0)}\rangle e^{-n\tau(E_0-E_{\mathrm{ref}})}$$

If we choose $E_{\mathrm{ref}} \approx E_0$, we obtain $\boxed{\lim_{n\to\infty} \Psi^{(n)} = \Psi_0}$

How do we perform the projection?

Rewrite projection equation in integral form

$$\boxed{\Psi(\mathbf{R}', t+\tau) = \int \mathrm{d}\mathbf{R}\, G(\mathbf{R}', \mathbf{R}, \tau)\Psi(\mathbf{R}, t)}$$

where $\quad G(\mathbf{R}', \mathbf{R}, \tau) = \langle \mathbf{R}' | e^{-\tau(\mathcal{H} - E_{\mathrm{ref}})} | \mathbf{R} \rangle$

- Can we sample the wave function?

  For the moment, assume we are dealing with $\boxed{\text{bosons}}$, so $\Psi > 0$

- Can we interpret $G(\mathbf{R}', \mathbf{R}, \tau)$ as a transition probability?

  If yes, we can perform this integral by Monte Carlo integration

VMC and DMC as power methods

VMC Distribution function is given $\boxed{P(\mathbf{R}) = \frac{|\Psi(\mathbf{R})|^2}{\int \mathrm{d}\mathbf{R}|\Psi(\mathbf{R})|^2}}$

Construct M which satisfies stationarity condition so that

$$\lim_{n\to\infty} \int \mathrm{d}\mathbf{R}_n \cdots \mathrm{d}\mathbf{R}_1 M(\mathbf{R}, \mathbf{R}_n) \cdots M(\mathbf{R}_3, \mathbf{R}_2) M(\mathbf{R}_2, \mathbf{R}_1) P_{\mathrm{init}}(\mathbf{R}_1) = P(\mathbf{R})$$

DMC Opposite procedure!

The matrix M is given $\rightarrow$ $\boxed{M \equiv G = \langle \mathbf{R}' | e^{-\tau(\mathcal{H} - E_{\mathrm{ref}})} | \mathbf{R} \rangle}$

We do not know $\boxed{P}$!

$$\lim_{n\to\infty} \int \mathrm{d}\mathbf{R}_n \cdots \mathrm{d}\mathbf{R}_1 G(\mathbf{R}, \mathbf{R}_n) \cdots G(\mathbf{R}_3, \mathbf{R}_2) G(\mathbf{R}_2, \mathbf{R}_1) P_{\mathrm{init}}(\mathbf{R}_1) = \Psi_0(\mathbf{R})$$

In either case, we want to find the dominant eigenvector of M

What can we say about the Green's function?

$$G(\mathbf{R}', \mathbf{R}, \tau) = \langle \mathbf{R}' | e^{-\tau(\mathcal{H} - E_{\mathrm{ref}})} | \mathbf{R} \rangle$$

$G(\mathbf{R}', \mathbf{R}, \tau)$ satisfies the imaginary-time Schrödinger equation

$$\boxed{(\mathcal{H} - E_{\mathrm{ref}}) G(\mathbf{R}, \mathbf{R}_0, t) = -\frac{\partial G(\mathbf{R}, \mathbf{R}_0, t)}{\partial t}}$$

with $G(\mathbf{R}', \mathbf{R}, 0) = \delta(\mathbf{R}' - \mathbf{R})$

Evolution equation of the probability distribution

We can understand the behavior of G which satisfies

$$\boxed{(\mathcal{H} - E_{\mathrm{ref}})G(\mathbf{R}, \mathbf{R}_0, t) = -\frac{\partial G(\mathbf{R}, \mathbf{R}_0, t)}{\partial t}}$$

to understand evolution of the distribution Ψ

$$\Psi(\mathbf{R}, t) = \int \mathrm{d}\mathbf{R}_0 \, G(\mathbf{R}, \mathbf{R}_0, t)\Psi^{(0)}(\mathbf{R}_0)$$

which satisfies the imaginary-time Schrödinger equation

$$\boxed{(\mathcal{H} - E_{\mathrm{ref}})\Psi(\mathbf{R}, t) = -\frac{\partial \Psi(\mathbf{R}, t)}{\partial t}}$$

$$\boxed{\text{Can we interpret } G(\mathbf{R}', \mathbf{R}, \tau) \text{ as a transition probability?}} \tag{1}$$

$\boxed{\mathcal{H} = \mathcal{T}}$

Imaginary-time Schrödinger equation is a diffusion equation

$$-\frac{1}{2}\nabla^2 G(\mathbf{R}, \mathbf{R}_0, t) = -\frac{\partial G(\mathbf{R}, \mathbf{R}_0, t)}{\partial t}$$

The Green's function is given by a Gaussian

$$G(\mathbf{R}', \mathbf{R}, \tau) = (2\pi\tau)^{-3N/2} \exp\left[-\frac{(\mathbf{R}' - \mathbf{R})^2}{2\tau}\right]$$

$\boxed{\text{Positive and can be sampled}}$

$\boxed{\text{Can we interpret } G(\mathbf{R}', \mathbf{R}, \tau) \text{ as a transition probability?}}$ (2)

$\boxed{\mathcal{H} = \mathcal{V}}$

$$(\mathcal{V}(\mathbf{R}) - E_{\mathrm{ref}}) G(\mathbf{R}, \mathbf{R}_0, t) = -\frac{\partial G(\mathbf{R}, \mathbf{R}_0, t)}{\partial t},$$

The Green's function is given by

$$G(\mathbf{R}', \mathbf{R}, \tau) = \exp\left[-\tau\left(\mathcal{V}(\mathbf{R}) - E_{\mathrm{ref}}\right)\right] \delta(\mathbf{R} - \mathbf{R}'),$$

$\boxed{\text{Positive}}$ but does not preserve the normalization

It is a factor by which we multiply the distribution $\Psi(\mathbf{R}, t)$

$\boxed{\mathcal{H} = \mathcal{T} + \mathcal{V} \text{ and a combination of diffusion and branching}}$

Let us combine previous results

$$\boxed{G(\mathbf{R}', \mathbf{R}, \tau) \approx (2\pi\tau)^{-3N/2} \exp\left[-\frac{(\mathbf{R}' - \mathbf{R})^2}{2\tau}\right] \exp\left[-\tau\left(\mathcal{V}(\mathbf{R}) - E_{\mathrm{T}}\right)\right]}$$

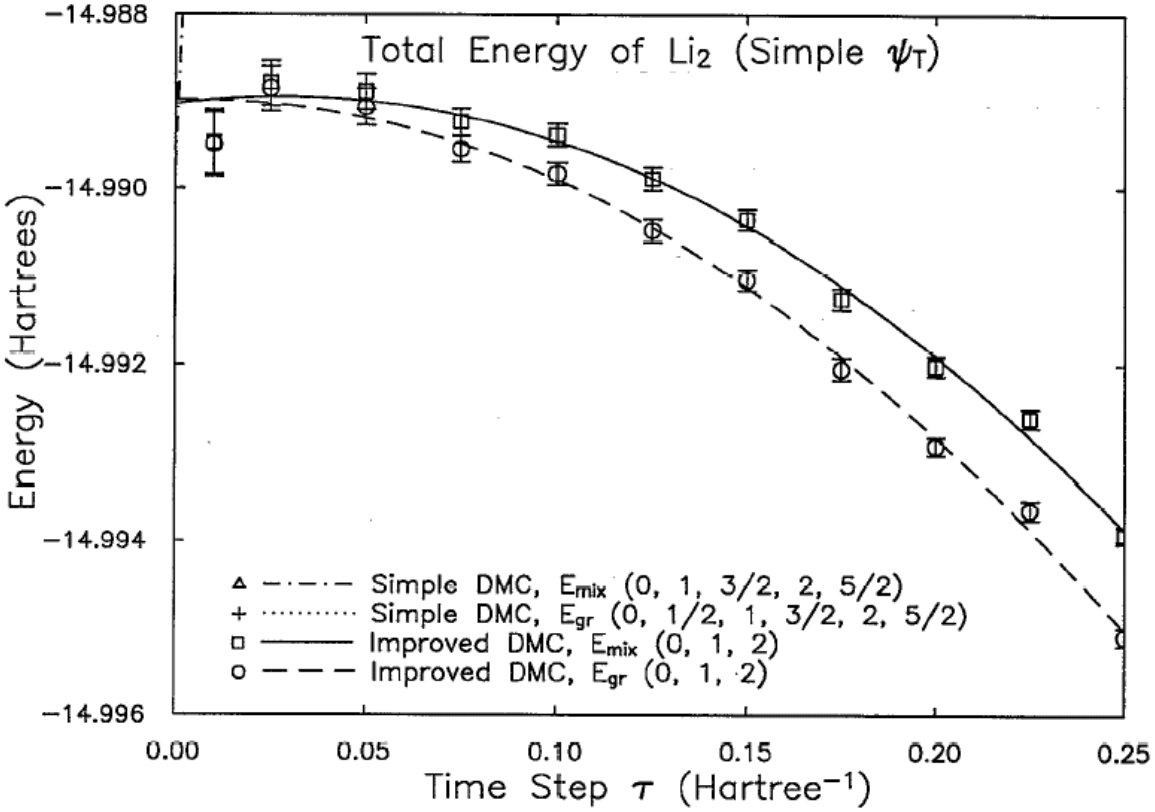
Diffusion + branching factor leading to survival/death/cloning

Why? Trotter's theorem $\rightarrow$ $\boxed{e^{(A+B)\tau} = e^{A\tau} e^{B\tau} + \mathcal{O}(\tau^2)}$

$\rightarrow$ Green's function in the $\boxed{\text{short-time approximation}}$ to $\mathcal{O}(\tau^2)$

Time-step extrapolation

Example: Energy of Li_2 versus time-step τ



Umrigar, Nightingale, Runge, J. Chem. Phys. **94**, 2865 (1993)

Diffusion Monte Carlo as a branching random walk

The basic DMC algorithm is rather simple:

1. Sample $\Psi^{(0)}(\mathbf{R})$ with the Metropolis algorithm

   Generate M_0 walkers $\mathbf{R}_1, \ldots, \mathbf{R}_{M_0}$ (zeroth generation)

2. Diffuse each walker as $\boxed{\mathbf{R}' = \mathbf{R} + \xi}$

   where ξ is sampled from $g(\xi) = (2\pi\tau)^{-3N/2} \exp\left(-\xi^2/2\tau\right)$

3. For each walker, compute the factor

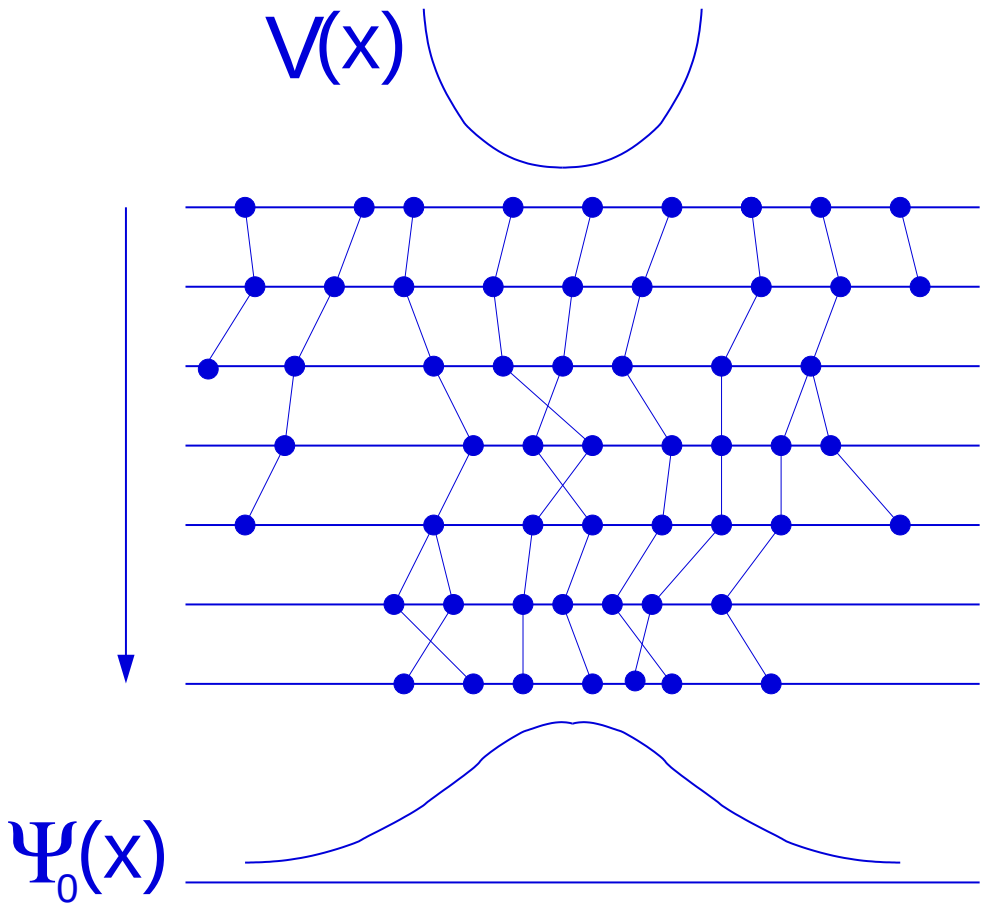
$$\boxed{p = \exp\left[-\tau(\mathcal{V}(\mathbf{R}) - E_{\mathrm{ref}})\right]}$$

   p is the probability to survive/proliferate/die

4. Adjust E_{ref} so that population fluctuates around target M_0

$\rightarrow$ After many iterations, walkers distributed as $\Psi_0(\mathbf{R})$

Diffusion and branching in a harmonic potential



Walkers proliferate/die where potential is lower/higher than E_{ref}

Problems with simple algorithm

The simple algorithm is inefficient and unstable

- Potential can vary a lot and be unbounded

  e.g. electron-nucleus interaction $\rightarrow$ Exploding population

- Branching factor grows with system size

Importance sampling

Start from integral equation

$$\Psi(\mathbf{R}', t+\tau) = \int \mathrm{d}\mathbf{R}\, G(\mathbf{R}', \mathbf{R}, \tau)\Psi(\mathbf{R}, t)$$

Multiply each side by trial Ψ_{T} and define $\boxed{\pi(\mathbf{R}, t) = \Psi_{\mathrm{T}}(\mathbf{R})\Psi(\mathbf{R}, t)}$

$$\pi(\mathbf{R}', t+\tau) = \int \mathrm{d}\mathbf{R}\, \tilde{G}(\mathbf{R}', \mathbf{R}, \tau)\pi(\mathbf{R}, t)$$

where the importance sampled Green's function is

$$\boxed{\tilde{G}(\mathbf{R}', \mathbf{R}, \tau) = \Psi_{\mathrm{T}}(\mathbf{R}')\langle \mathbf{R}'|e^{-\tau(\mathcal{H}-E_{\mathrm{ref}})}|\mathbf{R}\rangle/\Psi_{\mathrm{T}}(\mathbf{R})}$$

We obtain $\boxed{\lim_{n\to\infty} \pi(\mathbf{R}) = \Psi_{\mathrm{T}}(\mathbf{R})\Psi_0(\mathbf{R})}$

Importance sampled Green's function

The importance sampled $\tilde{G}(\mathbf{R}, \mathbf{R}_0, \tau)$ satisfies

$$\boxed{-\frac{1}{2}\nabla^2 \tilde{G} + \nabla \cdot [\tilde{G}\, \mathbf{V}(\mathbf{R})] + [E_{\mathrm{L}}(\mathbf{R}) - E_{\mathrm{ref}}]\, \tilde{G} = -\frac{\partial \tilde{G}}{\partial \tau}}$$

with quantum velocity $\mathbf{V}(\mathbf{R}) = \dfrac{\nabla \Psi_{\mathrm{T}}(\mathbf{R})}{\Psi_{\mathrm{T}}(\mathbf{R})}$ and $E_{\mathrm{L}}(\mathbf{R}) = \dfrac{\mathcal{H}\Psi_{\mathrm{T}}(\mathbf{R})}{\Psi_{\mathrm{T}}(\mathbf{R})}$

We now have $\boxed{\text{drift}}$ in addition to diffusion and branching terms

Trotter's theorem $\Rightarrow$ Consider them separately for small enough τ

The drift-diffusion-branching Green's function

Drift-diffusion-branching short-time Green's function is

$$\tilde{G}(\mathbf{R}', \mathbf{R}, \tau) = (2\pi\tau)^{-3N/2} \exp\left[-\frac{(\mathbf{R}' - \mathbf{R} - \tau\mathbf{V}(\mathbf{R}))^2}{2\tau}\right] \times \\ \times \exp\left\{-\tau\left(E_{\mathrm{L}}(\mathbf{R}) - E_{\mathrm{ref}}\right)\right\}$$

What is new in the drift-diffusion-branching expression?

- $\mathbf{V}(\mathbf{R})$ pushes walkers where Ψ is large
- $E_{\mathrm{L}}(\mathbf{R})$ is better behaved than the potential $\mathcal{V}(\mathbf{R})$

  Cusp conditions $\Rightarrow$ No divergences when particles approach

  As $\Psi_{\mathrm{T}} \to \Psi_0$, $E_{\mathrm{L}} \to E_0$ and branching factor is smaller

Basic DMC algorithm with importance sampling

1. Sample initial walkers from $|\Psi_{\mathrm{T}}(\mathbf{R})|^2$
2. Drift and diffuse the walkers as $\mathbf{R}' = \mathbf{R} + \tau \mathbf{V}(\mathbf{R}) + \xi$
   where ξ is sampled from $g(\xi) = (2\pi\tau)^{-3N/2} \exp\left(-\xi^2/2\tau\right)$
3. Branching step as in the simple algorithm but with the factor

$$p = \exp\left\{-\tau[(E_{\mathrm{L}}(\mathbf{R}) + E_{\mathrm{L}}(\mathbf{R}'))/2 - E_{\mathrm{ref}}]\right\}$$

4. Adjust the trial energy to keep the population stable

$\rightarrow$ After many iterations, walkers distributed as $\Psi_{\mathrm{T}}(\mathbf{R})\Psi_0(\mathbf{R})$

Electrons are fermions!

We assumed that $\Psi_0 > 0$ and that we are dealing with bosons

Fermions $\rightarrow$ Ψ is antisymmetric and changes sign!

Fermion Sign Problem

All fermion QMC methods suffer from sign problems

These sign problems look different but have the same "flavour"

Arise when you treat something non-positive as probability density

The DMC Sign Problem

How can we impose antisymmetry in simple DMC method?

Idea Evolve separate positive and negative populations of walkers

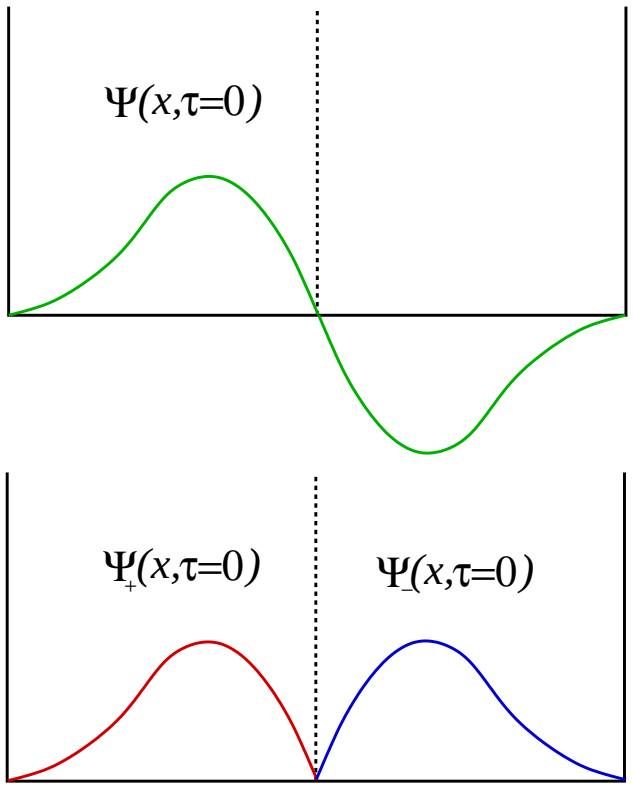
Simple 1D example: Antisymmetric wave function $\Psi(x, \tau = 0)$

Rewrite $\Psi(x, \tau = 0)$ as

$$\Psi = \Psi_+ - \Psi_-$$

where

$$\begin{aligned} \Psi_+ &= \frac{1}{2}(|\Psi| + \Psi) \\ \Psi_- &= \frac{1}{2}(|\Psi| - \Psi) \end{aligned}$$



Particle in a box and the fermionic problem (1)

The imaginary-time Schrödinger equation

$$\boxed{\mathcal{H}\Psi = -\frac{\partial \Psi}{\partial t}}$$

is linear, so solving it with the initial condition

$$\Psi(x, t = 0) \quad = \quad \Psi_+(x, t = 0) - \Psi_-(x, t = 0)$$

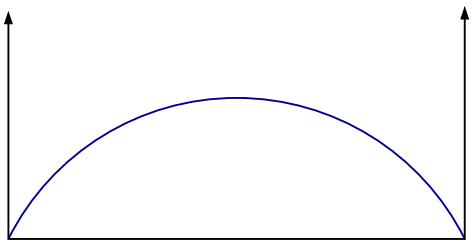
is equivalent to solving

$$\boxed{\mathcal{H}\Psi_+ = -\frac{\partial \Psi_+}{\partial t}} \qquad \text{and} \qquad \boxed{\mathcal{H}\Psi_- = -\frac{\partial \Psi_-}{\partial t}}$$

separately and subtracting one solution from the other

Particle in a box and the fermionic problem (2)

- Since $E_0^s < E_0^a$, both Ψ_+ and Ψ_- evolve to Ψ_0^s

$\Psi_\pm \longrightarrow$ 

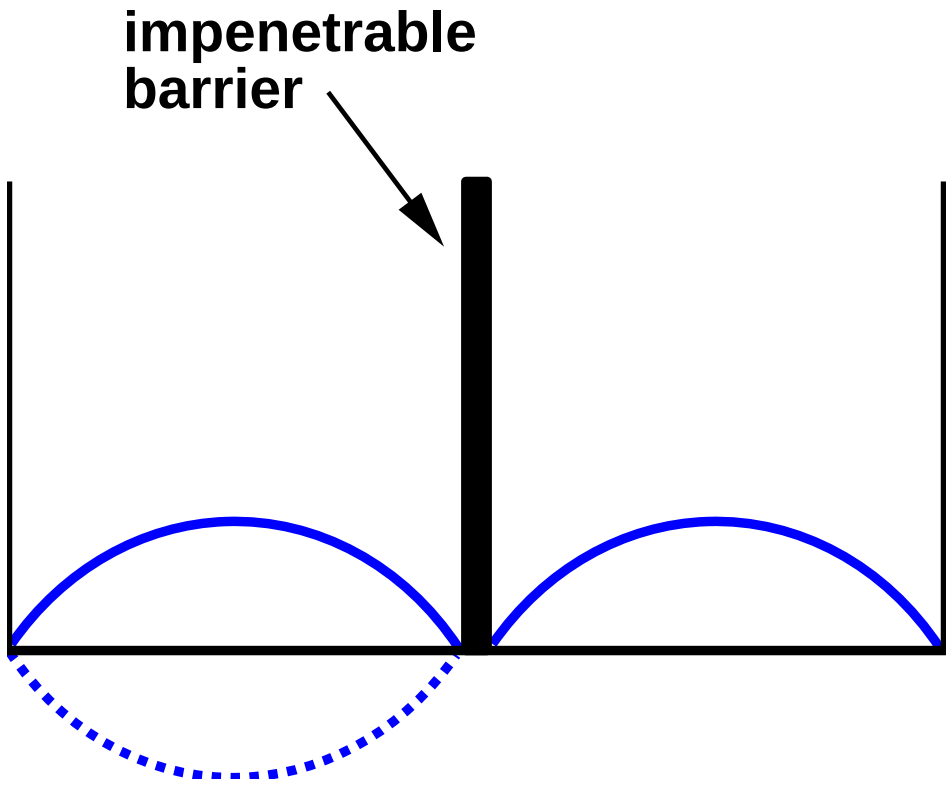
- Antisymmetric component exponentially harder to extract

$$\frac{|\Psi_+ - \Psi_-|}{|\Psi_+ + \Psi_-|} \propto \frac{e^{-E_0^a t}}{e^{-E_0^s t}} \quad \text{as} \quad t \to \infty$$

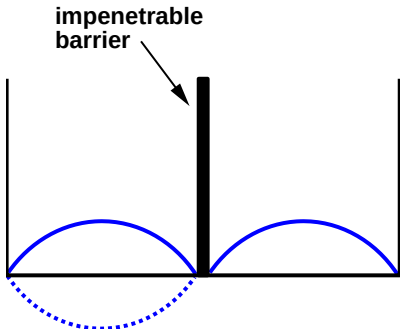
The Fixed-Node Approximation

Problem Small antisymmetric part swamped by random errors

Solution Fix the nodes! (If you don't know them, guess them)



Fixed-node algorithm in simple DMC



How do we impose this additional boundary condition?

- ▷ Annihilate walkers that bump into barrier (and into walls)
  - → This step enforces $\boxed{\Psi = 0}$ boundary conditions
  - → In each nodal pocket, evolution to ground state in pocket

Numerically $\boxed{\text{stable}}$ algorithm (no exponentially growing noise)

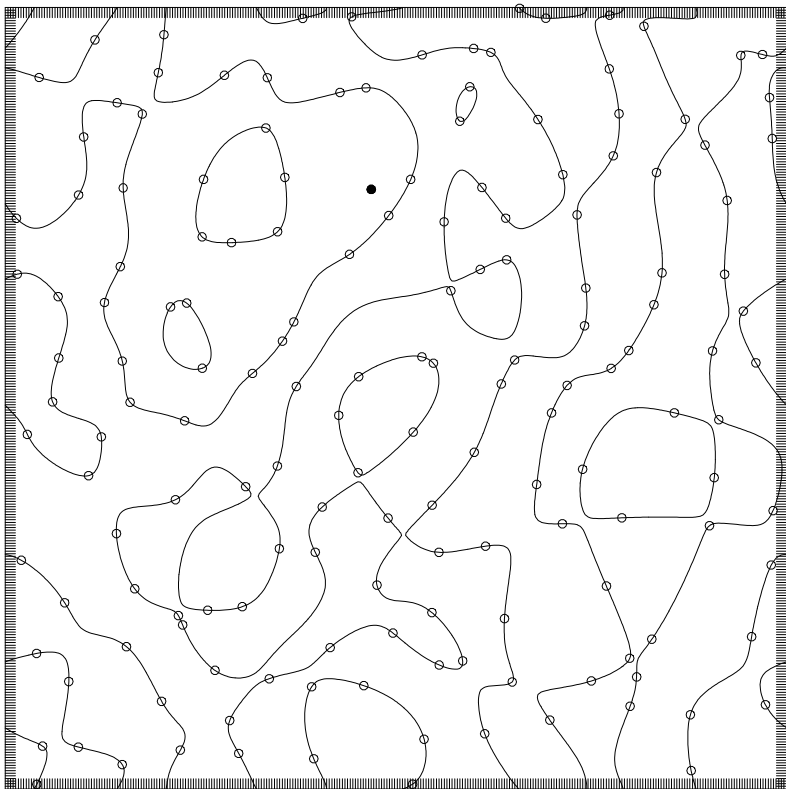
→ Solution is exact if nodes are exact

→ Best solution consistent with the assumed nodes

For many electrons, what are the nodes? A complex beast

Many-electron wave function $\Psi(\mathbf{R}) = \Psi(\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_N)$

$\boxed{\text{Node}}$ $\rightarrow$ surface where $\Psi = 0$ and across which Ψ changes sign



A 2D slice through the 321-dimensional nodal surface of a gas of 161 spin-up electrons.

Some known properties of the nodes

Physical space has d (=1,2,3) dimensions

- Node is $(dN-1)$-dimensional surface in dN dimensions

  $\boxed{\text{One}}$ constraint $(\Psi = 0) \Rightarrow \boxed{(dN-1)}$-dimensional node

- Equations as $\mathbf{r}_i = \mathbf{r}_j$ define $(dN-d)$-dimensional <u>coincidence</u> surfaces and do not define the node completely if $d > 1$

- If $d = 1$, coincidence points $x_i = x_j$ define the ground-state node completely $\rightarrow$ One-dim problems are easy to simulate

Nodal pockets can be divided up into classes

Start from $\mathbf{R}_0$ and continously reach all points with $\Psi(\mathbf{R}) \neq 0$

$\Rightarrow$ Nodal pocket accessible from $\mathbf{R}_0$

Map this subvolume over rest of the space with permutations

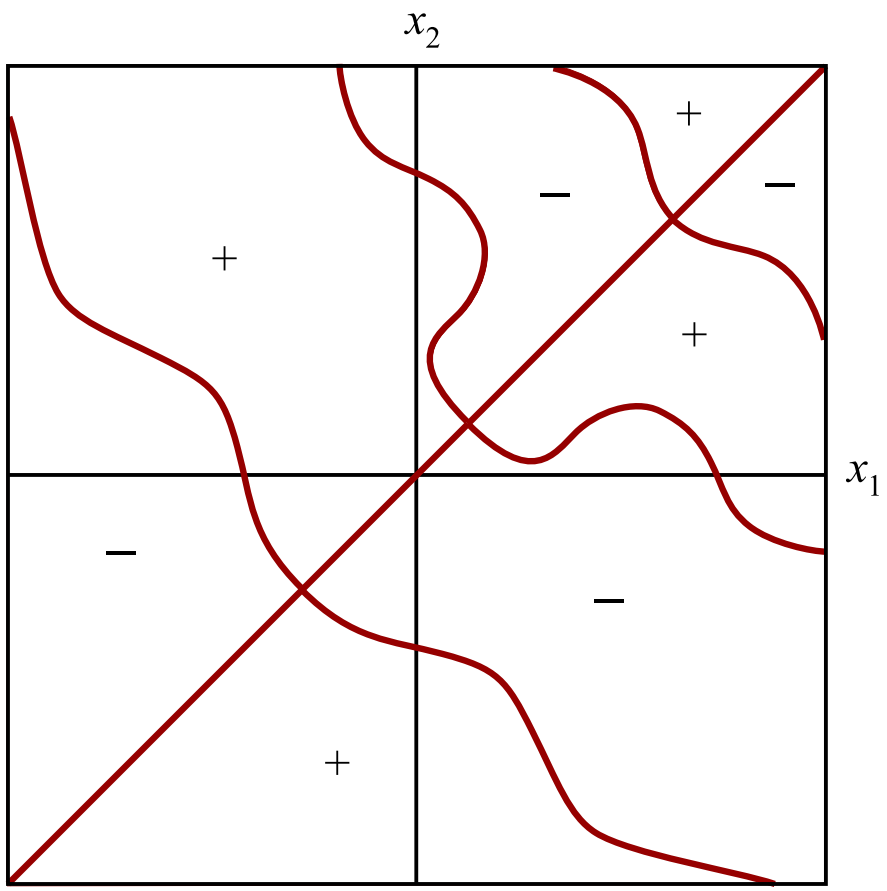


Figure courtesy of Matthew Foulkes

Nodal pockets can be divided up into classes

Start from $\mathbf{R}_0$ and continously reach all points with $\Psi(\mathbf{R}) \neq 0$

$\Rightarrow$ Nodal pocket accessible from $\mathbf{R}_0$

Map this subvolume over rest of the space with permutations

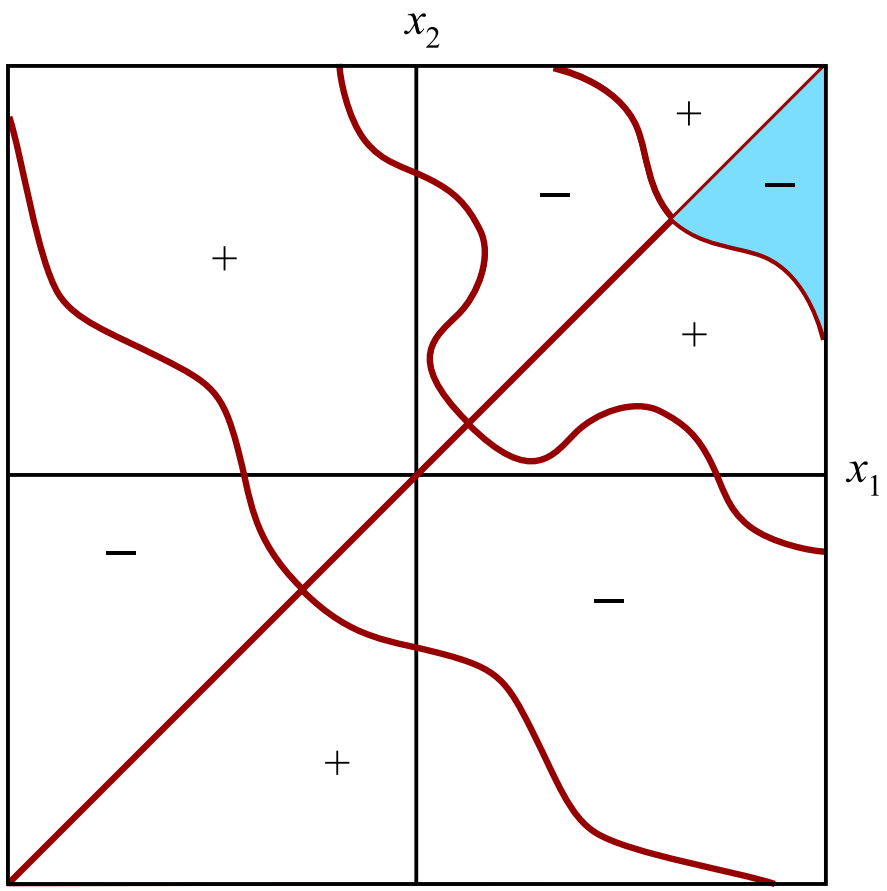


Figure courtesy of Matthew Foulkes

Nodal pockets can be divided up into classes

Start from $\mathbf{R}_0$ and continously reach all points with $\Psi(\mathbf{R}) \neq 0$

$\Rightarrow$ Nodal pocket accessible from $\mathbf{R}_0$

Map this subvolume over rest of the space with permutations

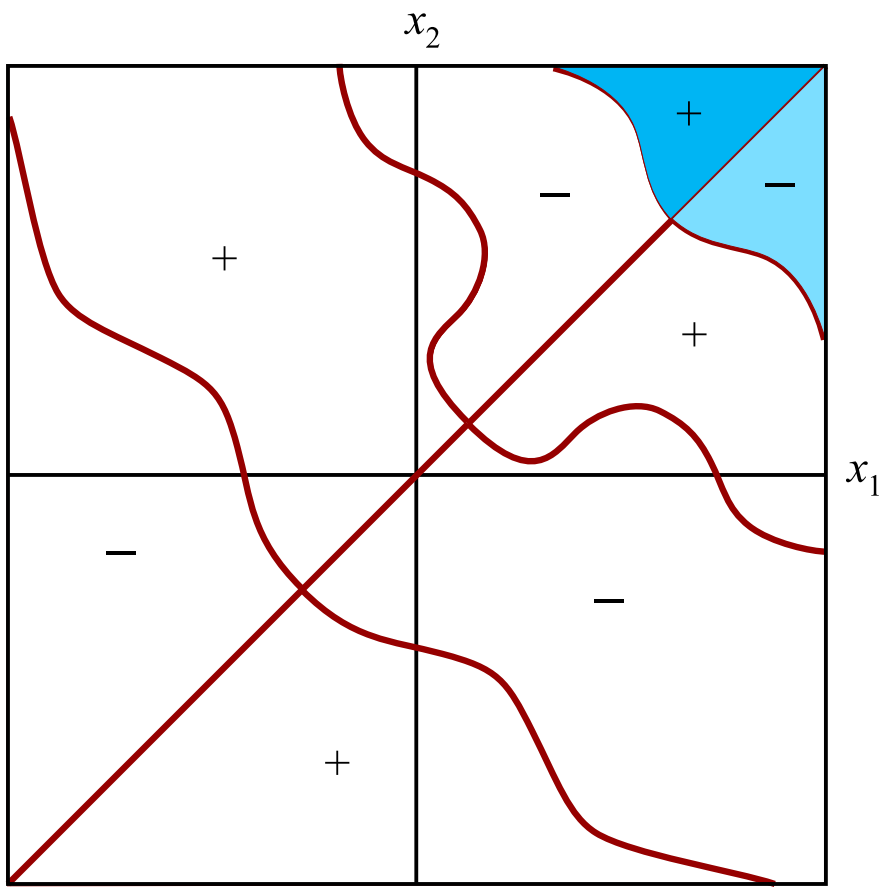


Figure courtesy of Matthew Foulkes

Nodal pockets can be divided up into classes

Start from $\mathbf{R}_0$ and continously reach all points with $\Psi(\mathbf{R}) \neq 0$

$\Rightarrow$ Nodal pocket accessible from $\mathbf{R}_0$

Map this subvolume over rest of the space with permutations

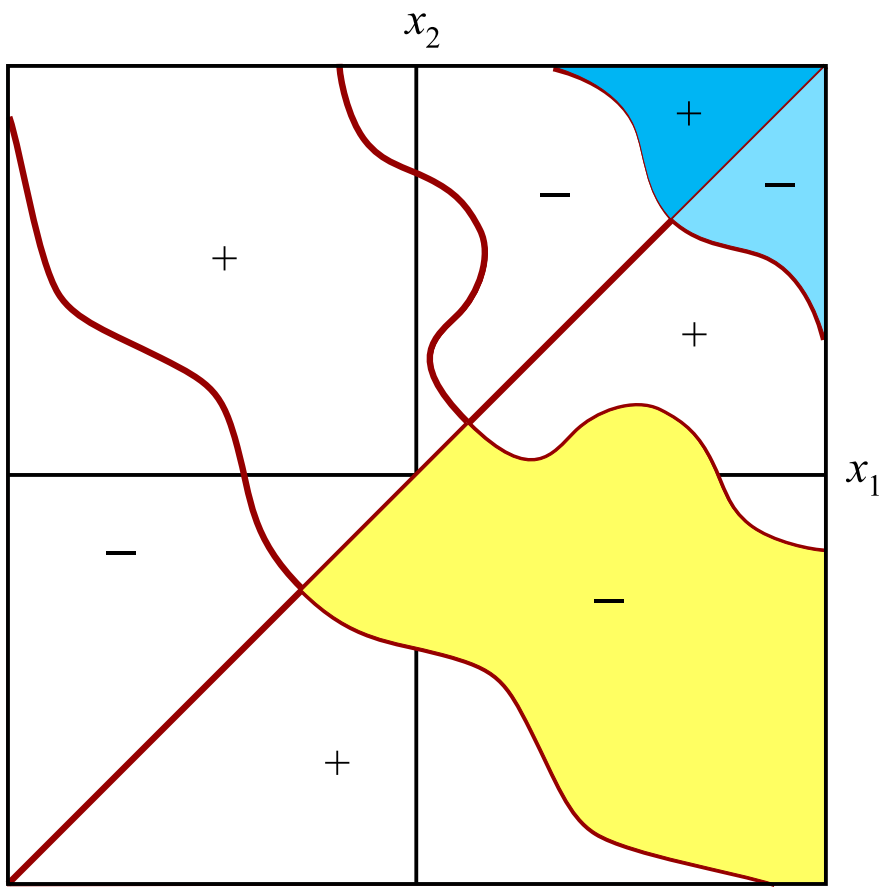


Figure courtesy of Matthew Foulkes

Nodal pockets can be divided up into classes

Start from $\mathbf{R}_0$ and continously reach all points with $\Psi(\mathbf{R}) \neq 0$

$\Rightarrow$ Nodal pocket accessible from $\mathbf{R}_0$

Map this subvolume over rest of the space with permutations

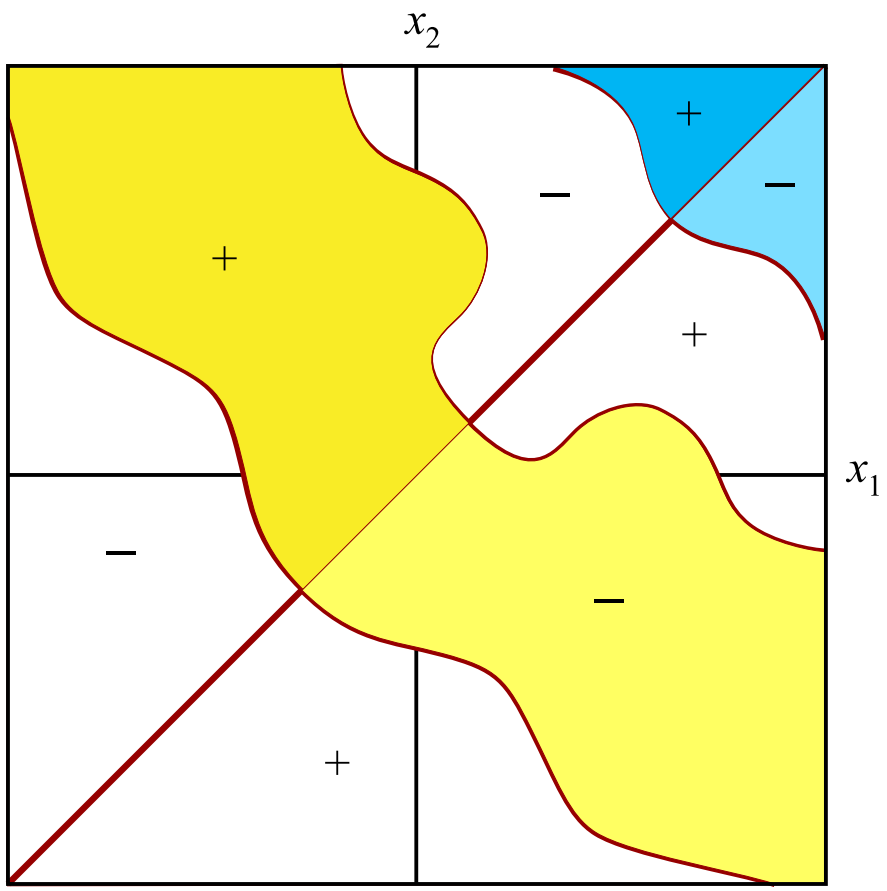


Figure courtesy of Matthew Foulkes

Nodal pockets can be divided up into classes

Start from $\mathbf{R}_0$ and continously reach all points with $\Psi(\mathbf{R}) \neq 0$

$\Rightarrow$ Nodal pocket accessible from $\mathbf{R}_0$

Map this subvolume over rest of the space with permutations

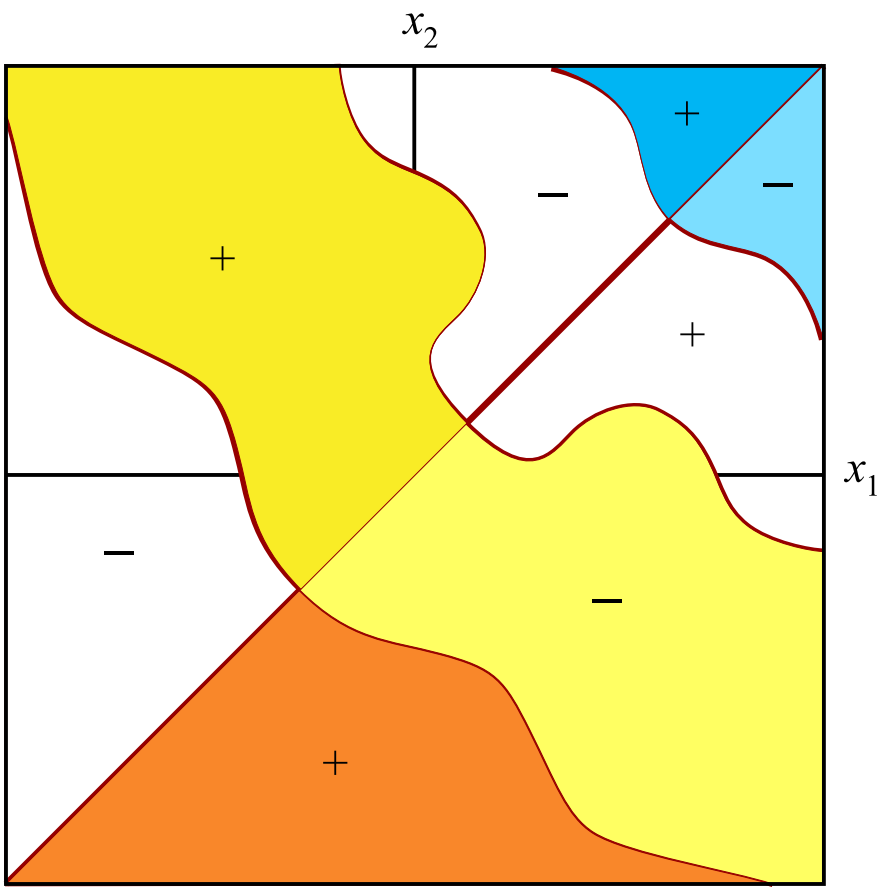


Figure courtesy of Matthew Foulkes

Nodal pockets can be divided up into classes

Start from $\mathbf{R}_0$ and continously reach all points with $\Psi(\mathbf{R}) \neq 0$

$\Rightarrow$ Nodal pocket accessible from $\mathbf{R}_0$

Map this subvolume over rest of the space with permutations

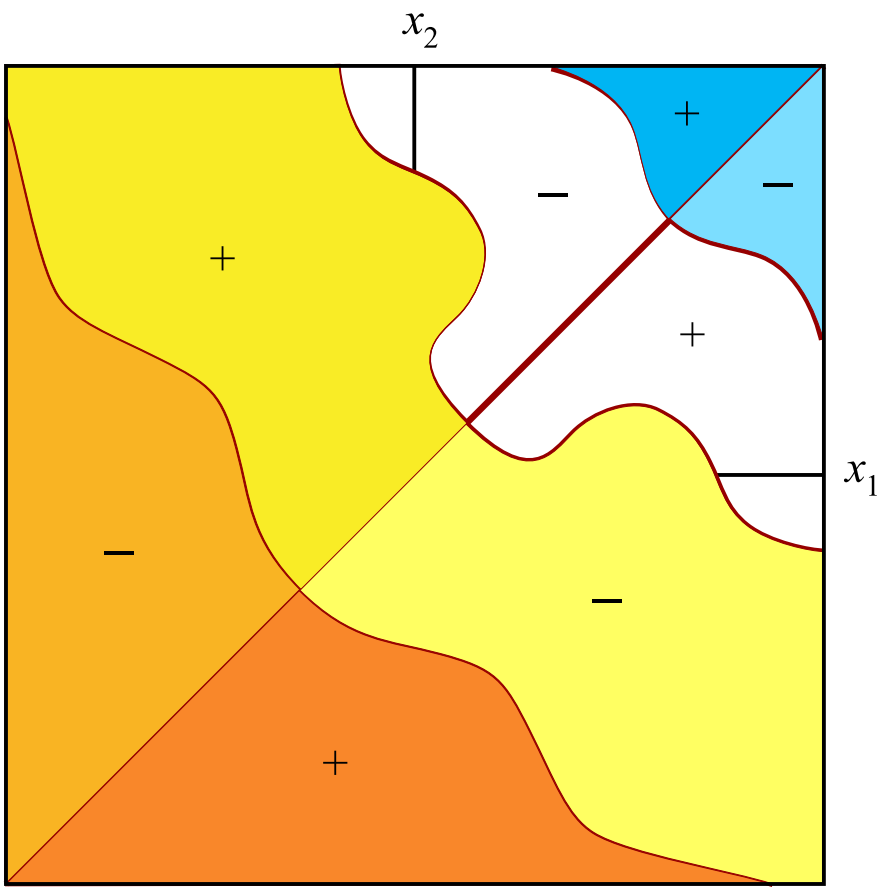


Figure courtesy of Matthew Foulkes

Nodal pockets can be divided up into classes

Start from $\mathbf{R}_0$ and continously reach all points with $\Psi(\mathbf{R}) \neq 0$

$\Rightarrow$ Nodal pocket accessible from $\mathbf{R}_0$

Map this subvolume over rest of the space with permutations

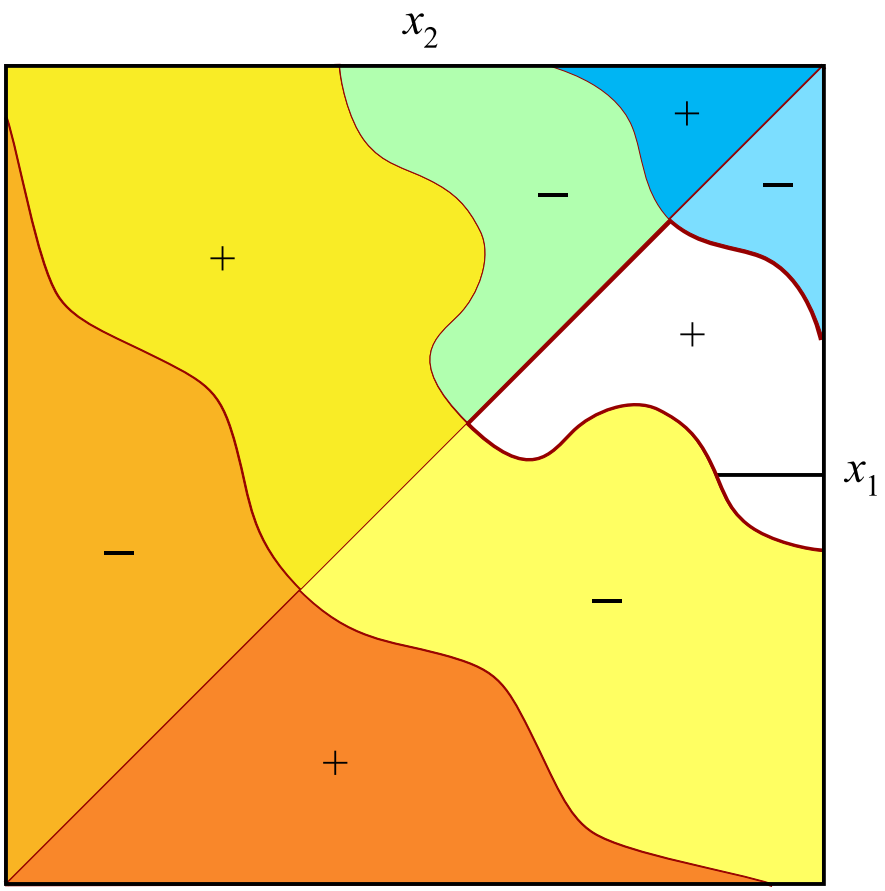


Figure courtesy of Matthew Foulkes

Nodal pockets can be divided up into classes

Start from $\mathbf{R}_0$ and continously reach all points with $\Psi(\mathbf{R}) \neq 0$

$\Rightarrow$ Nodal pocket accessible from $\mathbf{R}_0$

Map this subvolume over rest of the space with permutations

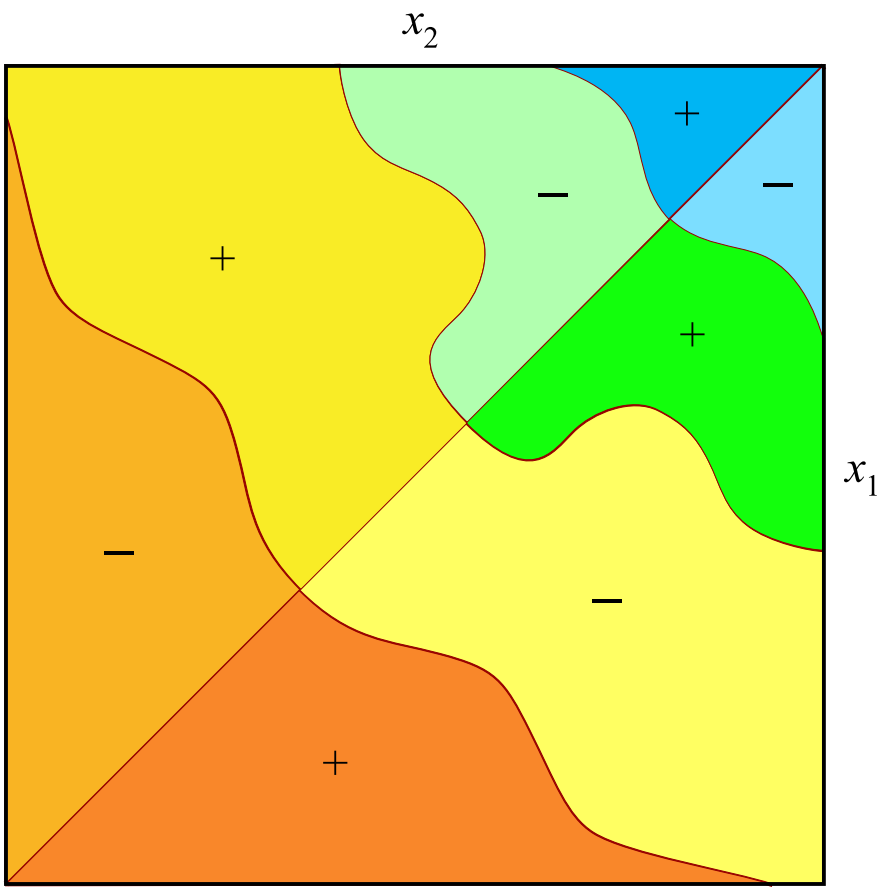


Figure courtesy of Matthew Foulkes

The Tiling Theorem

Consider Hamiltonian with a local potential

For ground-state wavefunction, all pockets are in the same class

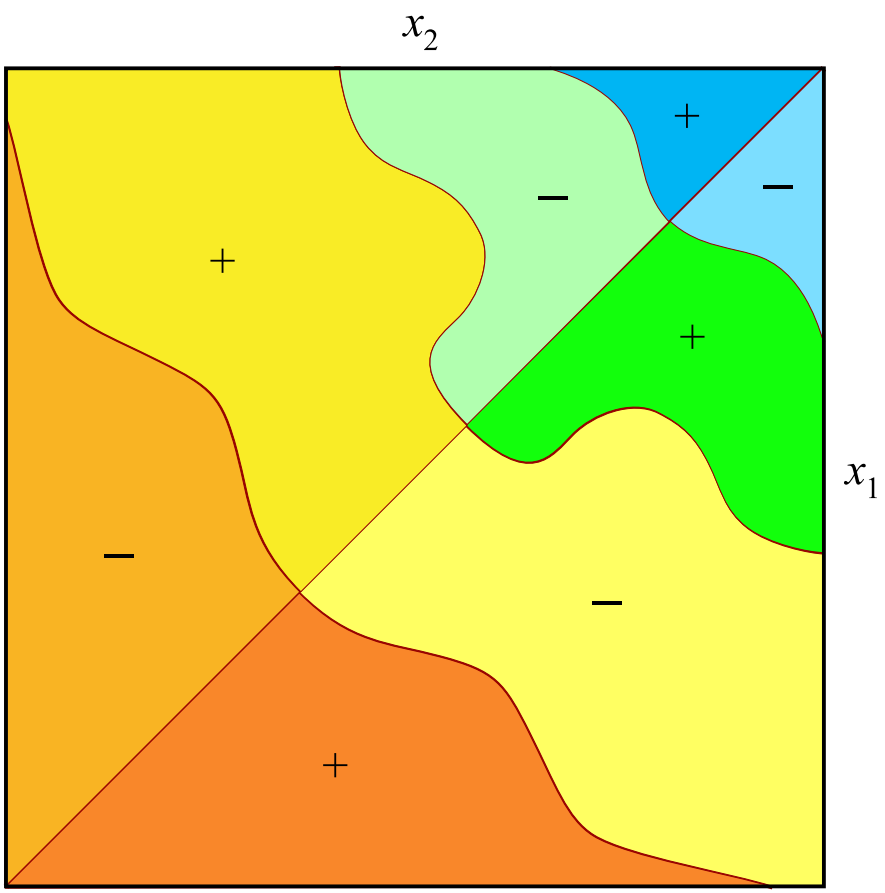


Figure courtesy of Matthew Foulkes

The Tiling Theorem

Consider Hamiltonian with a local potential

For ground-state wavefunction, all pockets are in the same class

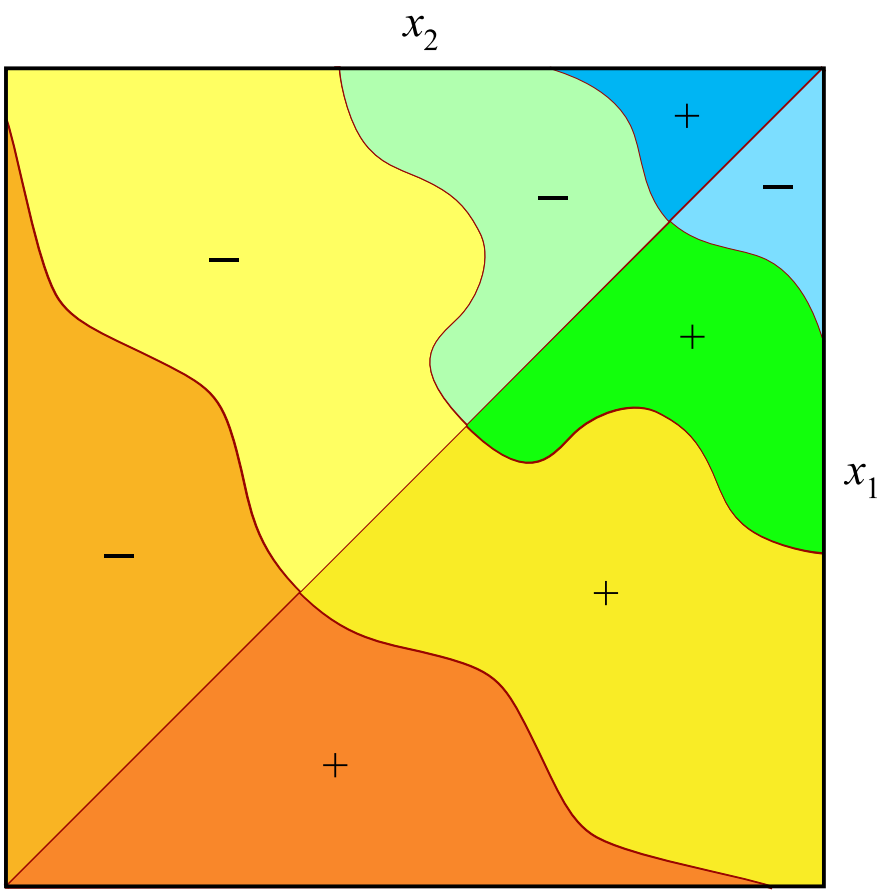


Figure courtesy of Matthew Foulkes

The Tiling Theorem

Consider Hamiltonian with a local potential

For ground-state wavefunction, all pockets are in the same class

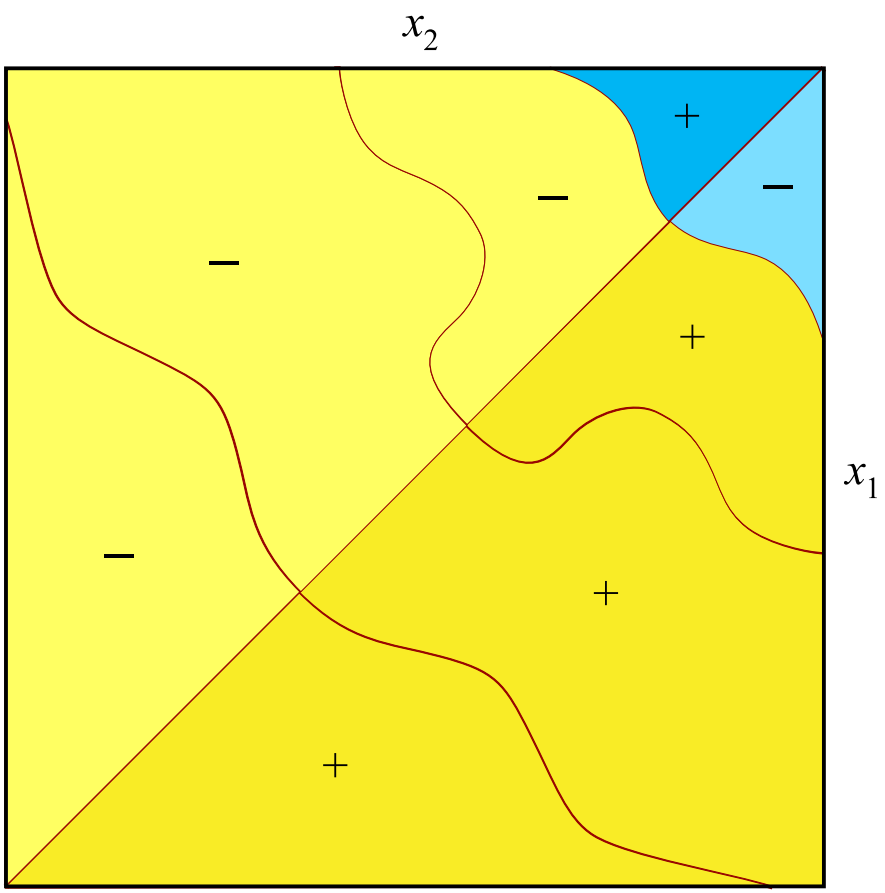


Figure courtesy of Matthew Foulkes

The Tiling Theorem

Consider Hamiltonian with a local potential

For ground-state wavefunction, all pockets are in the same class

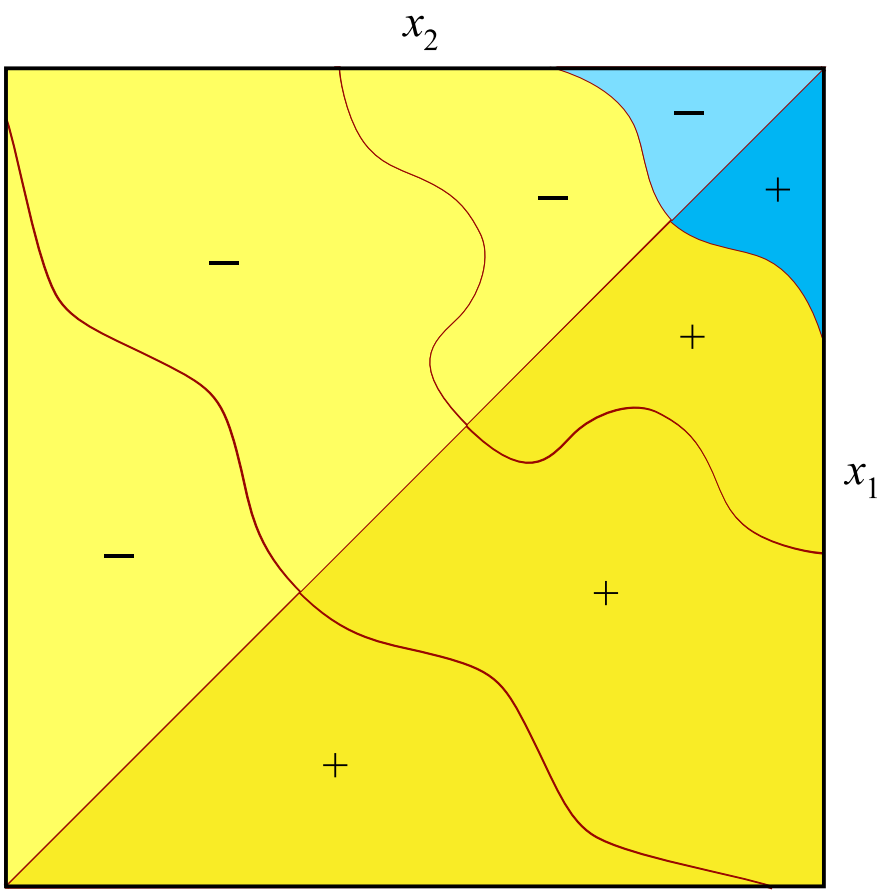


Figure courtesy of Matthew Foulkes

The Tiling Theorem

Consider Hamiltonian with a local potential

For ground-state wavefunction, all pockets are in the same class

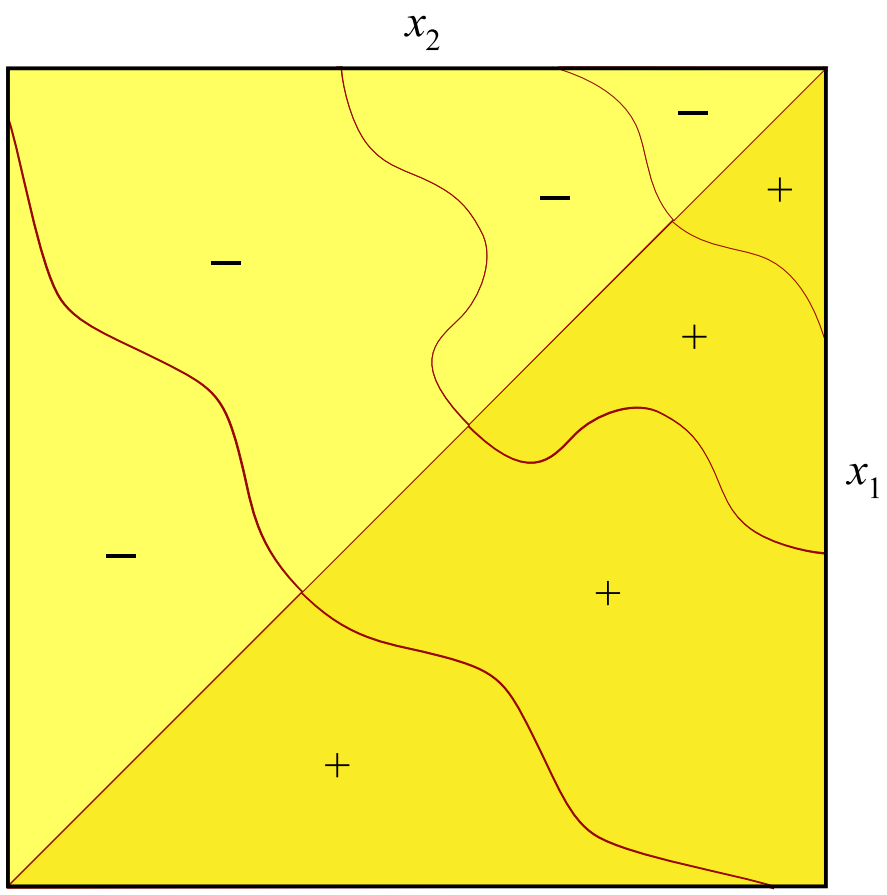
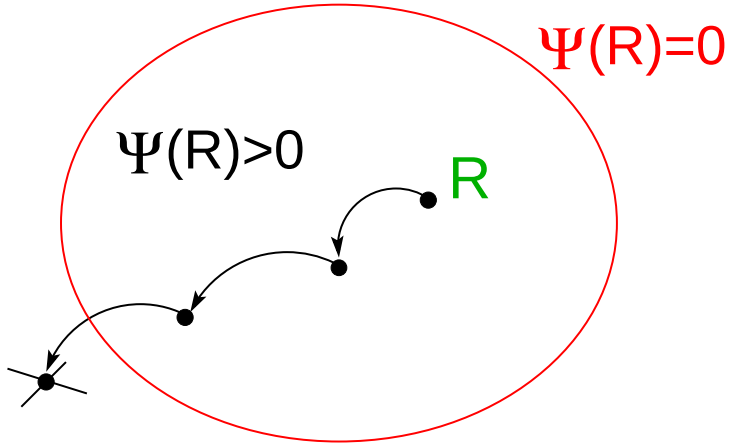


Figure courtesy of Matthew Foulkes

Use the nodes of trial $\Psi_T \rightarrow$ Fixed-node approximation

Use the nodes of the best available trial Ψ_T wave function



Find best solution with same nodes as trial wave function Ψ_T

Fixed-node solution exact if the nodes of trial Ψ_T are exact

Easy to implement in DMC with importance sampling: $\pi \geq 0$

Fixed-node solution and importance-sampling DMC

Given trial $\Psi_{\mathrm{T}}(\mathbf{R})$, evolve $\boxed{\pi(\mathbf{R},t) = \Psi_{\mathrm{T}}(\mathbf{R})\Psi(\mathbf{R},t)}$ as

$$-\frac{1}{2}\nabla^2\pi + \nabla\cdot[\pi\,\mathbf{V}(\mathbf{R})] + [E_{\mathrm{L}}(\mathbf{R}) - E_{\mathrm{ref}}]\,\pi = -\frac{\partial\pi}{\partial\tau}$$

with $\mathbf{V}(\mathbf{R}) = \dfrac{\nabla\Psi_{\mathrm{T}}(\mathbf{R})}{\Psi_{\mathrm{T}}(\mathbf{R})}$ and $E_{\mathrm{L}}(\mathbf{R}) = \dfrac{\mathcal{H}\Psi_{\mathrm{T}}(\mathbf{R})}{\Psi_{\mathrm{T}}(\mathbf{R})}$

Fixed-node approximation $\rightarrow$ $\boxed{\pi(\mathbf{R},t) \geq 0}$

Fixed-node solution and behavior at the nodes

Within the nodes $\boxed{\mathcal{H}\Psi_{\mathrm{FN}}(\mathbf{R}) = E_{\mathrm{FN}}\Psi_{\mathrm{FN}}(\mathbf{R})}$

If the nodes not exact $\rightarrow$ $\Psi_{\mathrm{FN}} \neq \Psi_0$

If the nodes not exact $\rightarrow$ Discontinuity of derivatives at the nodes

$$\boxed{\mathcal{H}\Psi_{\mathrm{FN}}(\mathbf{R}) = E_{\mathrm{FN}}\Psi_{\mathrm{FN}}(\mathbf{R}) + \delta} \quad \text{for} \quad \mathbf{R} \in \delta\Omega$$

Note that the δ function does not affect the computed energy

$$\int \Psi_{FN}\mathcal{H}\Psi_{\mathrm{FN}} = \int \Psi_{FN}(E_{\mathrm{FN}}\Psi_{FN} + \delta) = \int \Psi_{FN}E_{\mathrm{FN}}\Psi_{FN} = E_{\mathrm{FN}}$$

Fixed-node solution is an upper bound to exact energy

In a nodal pocket Ω of the trial wave function Ψ

$$\mathcal{H}\Psi_{\mathrm{FN}}(\mathbf{R}) = E_{\mathrm{FN}}\Psi_{\mathrm{FN}}(\mathbf{R}) \qquad \mathbf{R} \in \Omega$$

with $\Psi_{\mathrm{FN}}(\mathbf{R}) = 0$ for $\mathbf{R} \notin \Omega$ $\rightarrow$ Extend solution over all space

$$\tilde{\Psi}_{\mathrm{FN}}(\mathbf{R}) = \frac{1}{N!}\sum_{\mathrm{P}}(-1)^{\mathrm{P}}\Psi_{\mathrm{FN}}(\mathrm{P}\mathbf{R})$$

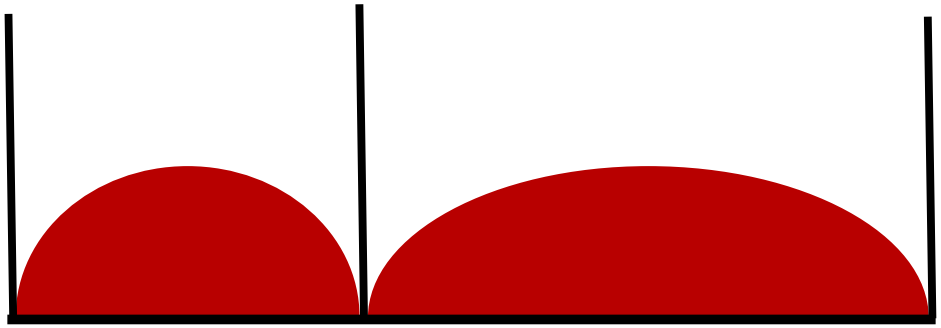
which satisfies

$$\boxed{\frac{\int \mathrm{d}\mathbf{R}\, \tilde{\Psi}^*_{\mathrm{FN}}(\mathbf{R})\mathcal{H}\tilde{\Psi}_{\mathrm{FN}}(\mathbf{R})}{\int \mathrm{d}\mathbf{R}\, \tilde{\Psi}^*_{\mathrm{FN}}(\mathbf{R})\tilde{\Psi}_{\mathrm{FN}}(\mathbf{R})} = E_{\mathrm{FN}} \geq E_0}$$

Fixed-node DMC and excited states

No general fixed-node variational principle for excited states

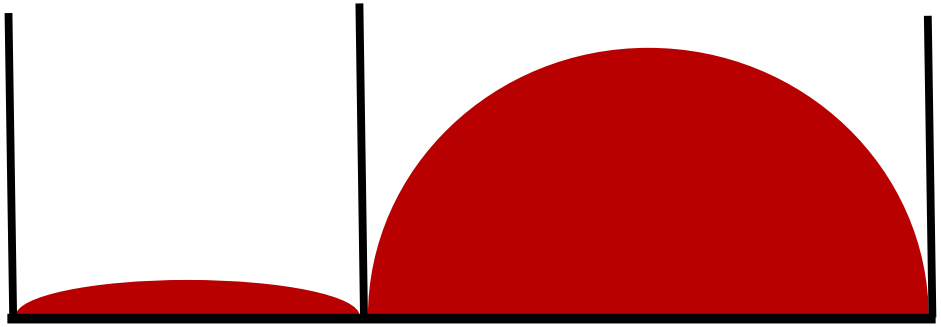
$\tau = 0$:



Fixed-node DMC and excited states

No general fixed-node variational principle for excited states

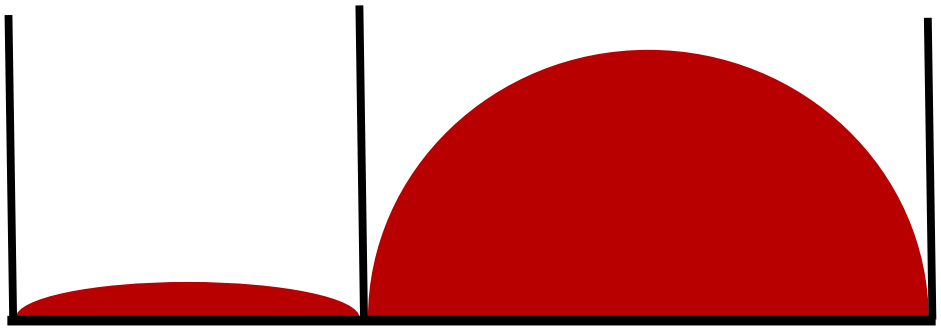
$\tau > \mathbf{0}$:



Fixed-node DMC and excited states

No general fixed-node variational principle for excited states

$\tau > \mathbf{0}$:



For $t \to \infty$, only pockets of the lowest energy class are occupied

It can happen that $E_{\mathrm{FN}} < E_{\mathrm{exact}}$

Fixed-node diffusion Monte Carlo and excited states

Is fixed-node diffusion Monte Carlo variational?

For lowest state in each 1-dim irreducible representation

What about "real" excited states?

In general, exact excited state for exact nodal structure

For excited states, even bigger role of the trial wave function

$\rightarrow$ Enforces fermionic antisymmetry + selects the state

In practice, for reasonable wave function, no collapse

$\rightarrow$ fixed-node DMC approaches excited state from above

Have we solved all our problems?

Results depend on the nodes of the trail wave function Ψ

Diffusion Monte Carlo as a black-box approach?

ϵ_{MAD} for atomization energy of the G1 set

	DMC			CCSD(T)/aug-cc-pVQZ
	HF orb	Optimized orb	CAS	
ϵ_{MAD}	3.1	2.1	1.2	2.8 kcal/mol

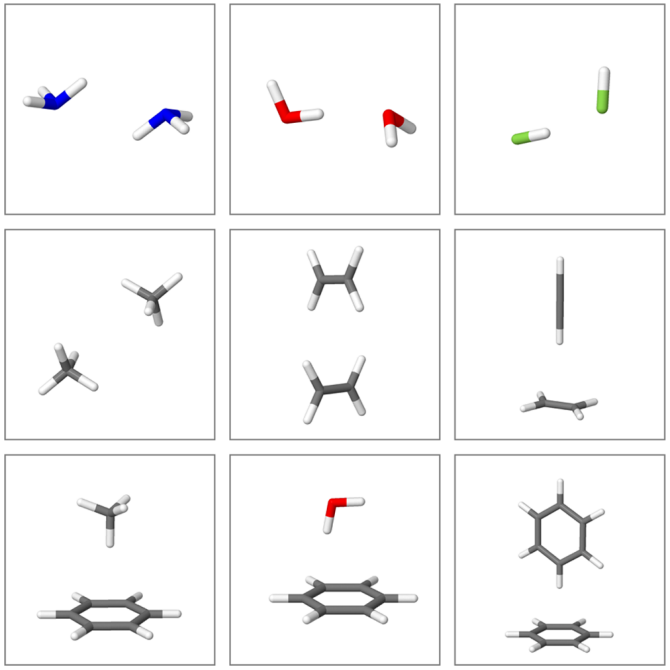
Petruzielo, Toulouse, Umrigar, J. Chem. Phys. **136**, 124116 (2012)

With "some" effort on Ψ, we can do rather well

Diffusion Monte Carlo as a black-box approach?

Non-covalent interaction energies for 9 compounds from S22 set

DMC with B3LYP/aug-cc-PVTZ orbitals versus CCSD(T)/CBS



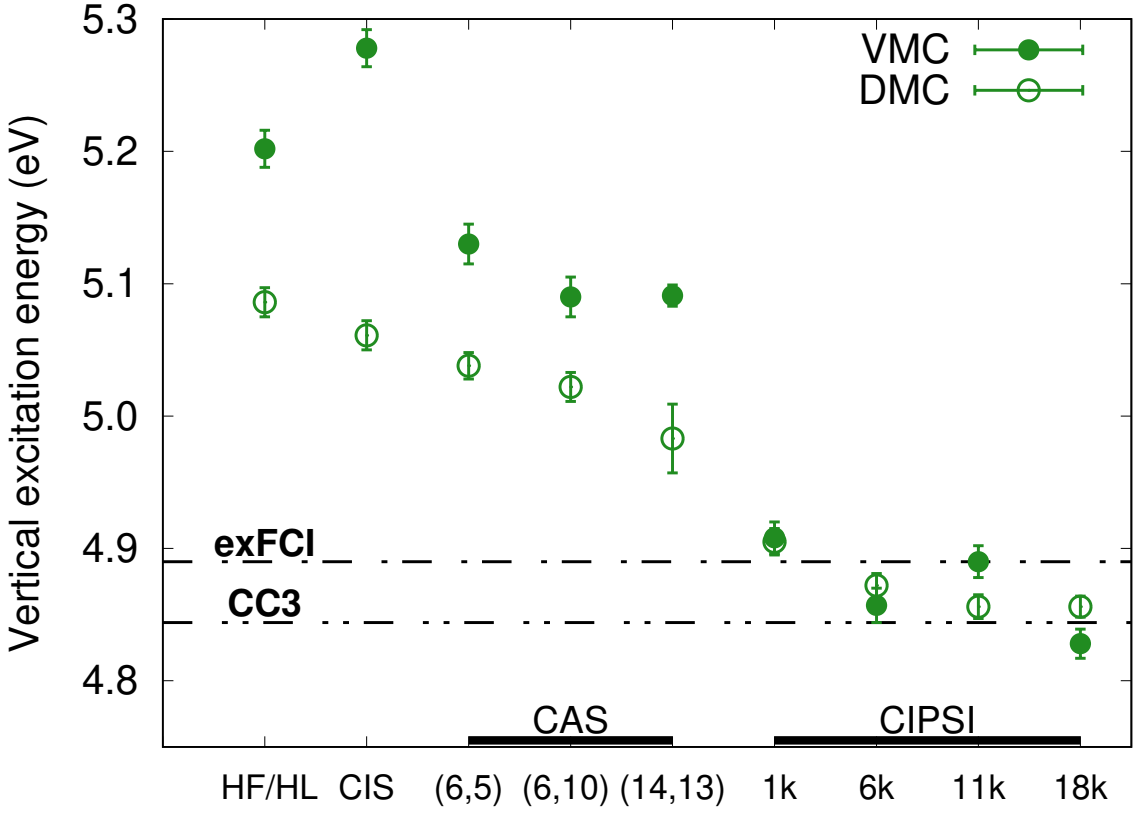
$\Delta_{\mathrm{MAD}} = 0.058$ kcal/mol

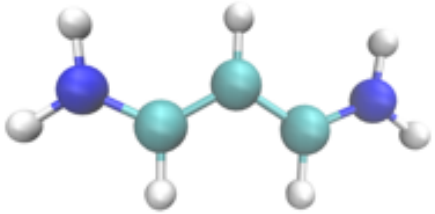
Dubecky *et al.*, JCTC **9**, 4287 (2013)

With "practically no" effort on Ψ, we can do rather well

Diffusion Monte Carlo end excitation energy

Excitation energy and wave function dependence





Cuzzocrea, Scemama, Briels, Moroni, Filippi, JCTC **16**, 4203 (2020)

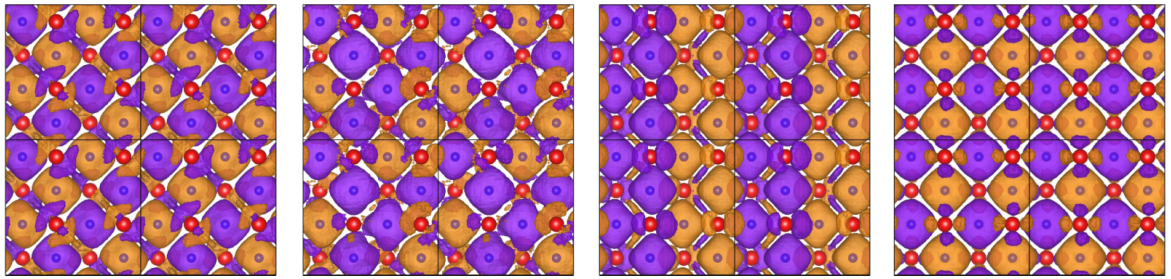
DMC is not a panacea but effort on Ψ pays off!

DMC and solid state calculations

Example: Structural/magnetic properties of superconducting FeSe

→ Accurate lattice constants, bulk moduli, and band dispersion

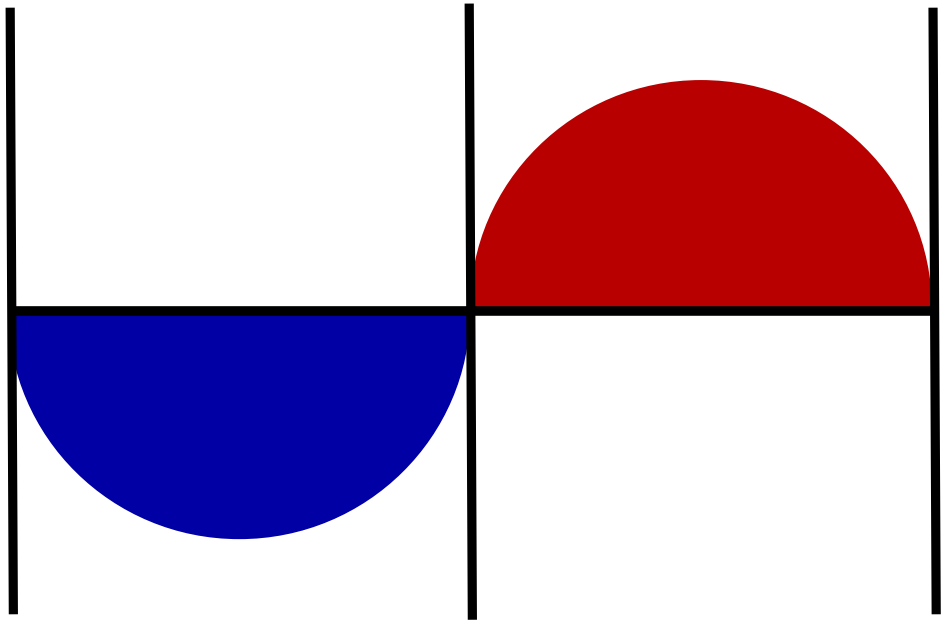
→ Resolving relative energetics of different magnetic ordering



Busemeyer, Dagrada, Sorella, Casula, and Wagner PRB (2016)

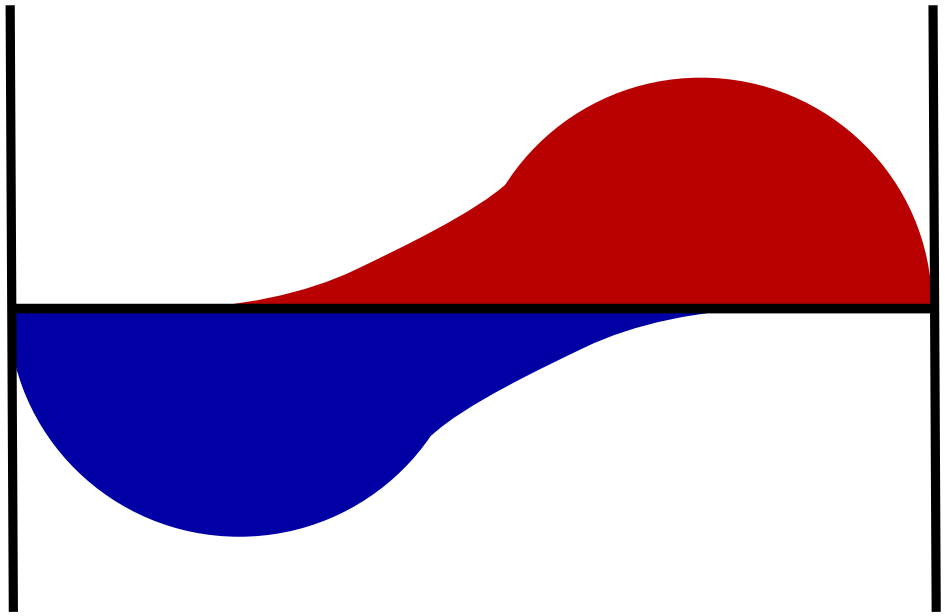
Alternatives to fixed-node DMC: Releasing the nodes

First do a fixed-node DMC simulation



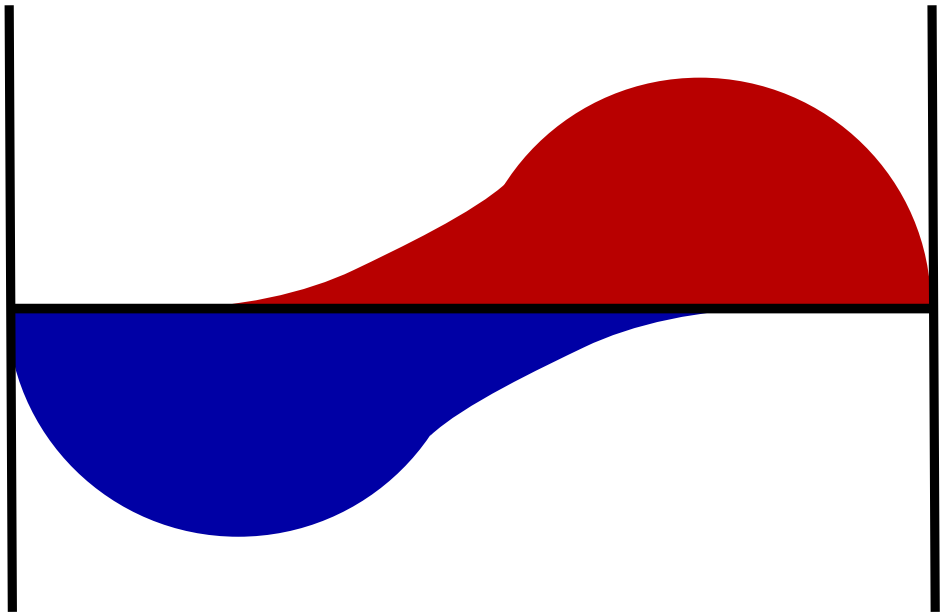
Alternatives to fixed-node DMC: Releasing the nodes

Then release the nodes



Alternatives to fixed-node DMC: Releasing the nodes

Then release the nodes



- Red and blue solutions collapse to boson ground state, but their difference approaches the fermion ground state
- Back to the sign problem: exponentially growing noise

Alternatives to fixed-node DMC: Determinantal QMC (2)

Given single-particle basis, perform projection in determinant space

Different way to deal with fermionic problem

- Determinantal QMC by Zhang and Krakauer

  Appears less plagued by fixed phase than DMC by FN

- Full-CI QMC by Alavi

  Start from $\Psi_{\mathrm{CI}} = \sum_i c_i D_i$

$$\mathcal{H}\Psi = -\frac{\partial \Psi}{\partial t} \rightarrow H_{ij} c_j = -\frac{\partial c_i}{\partial t}$$

DMC in summary

The fixed-node DMC method is (in general)

- Easy to do
- Stable
- Accurate enough for many applications in quantum chemistry
  ... especially in large systems
- Accurate enough also for subtle correlation physics

Use of fixed-node DMC for computation of excited states

- In the general landscape, we are doing quite well !
- Sensitivity to wave function but relatively robust
  $\rightarrow$ basis, size of the determinantal expansion

Beauty of quantum Monte Carlo → Highly parallelizable

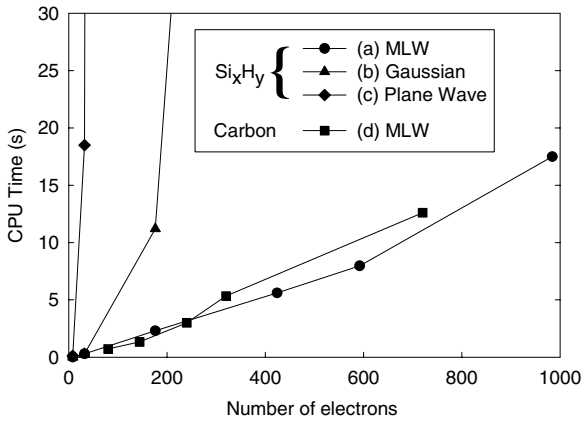
$\Psi(\mathbf{r}_1, \ldots, \mathbf{r}_N)$ → Ensemble of walkers diffusing in 3N dimensions

VMC → Independent walkers ⇒ Trival parallelization

DMC → Nearly independent walkers ⇒ Few communications

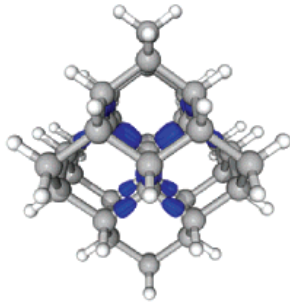
Easily take great advantage of parallel supercomputers!

As early as 2001 ...



Williamson, Hood, Grossman (2001)

Up to $Si_{123}H_{100}$ and C_{180} !



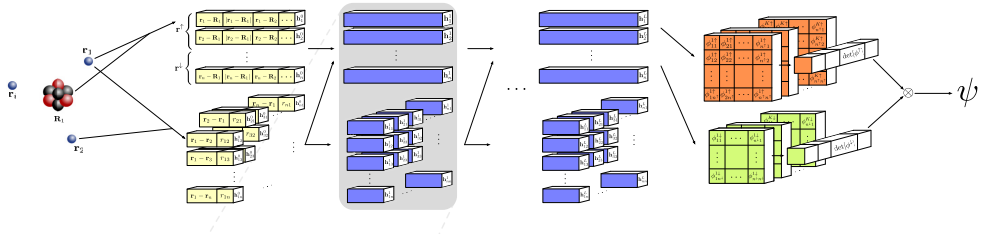
Human and computational cost of a typical QMC calculation

Task	Human time	Computer time
Choice of basis set, pseudo etc.	10%	5%
DFT/HF/CI runs for Ψ setup	65%	10%
Optimization of Ψ	20%	50%
DMC calculation	5%	35%

To conclude: ongoing research in QMC

- Search for different forms of trial wave function

  Neural network architecture $\rightarrow$ Ψ of multi-electron orbitals



  Pfau, Spencer, Matthews, Foulkes, Phys. Rev. Res. (2020)

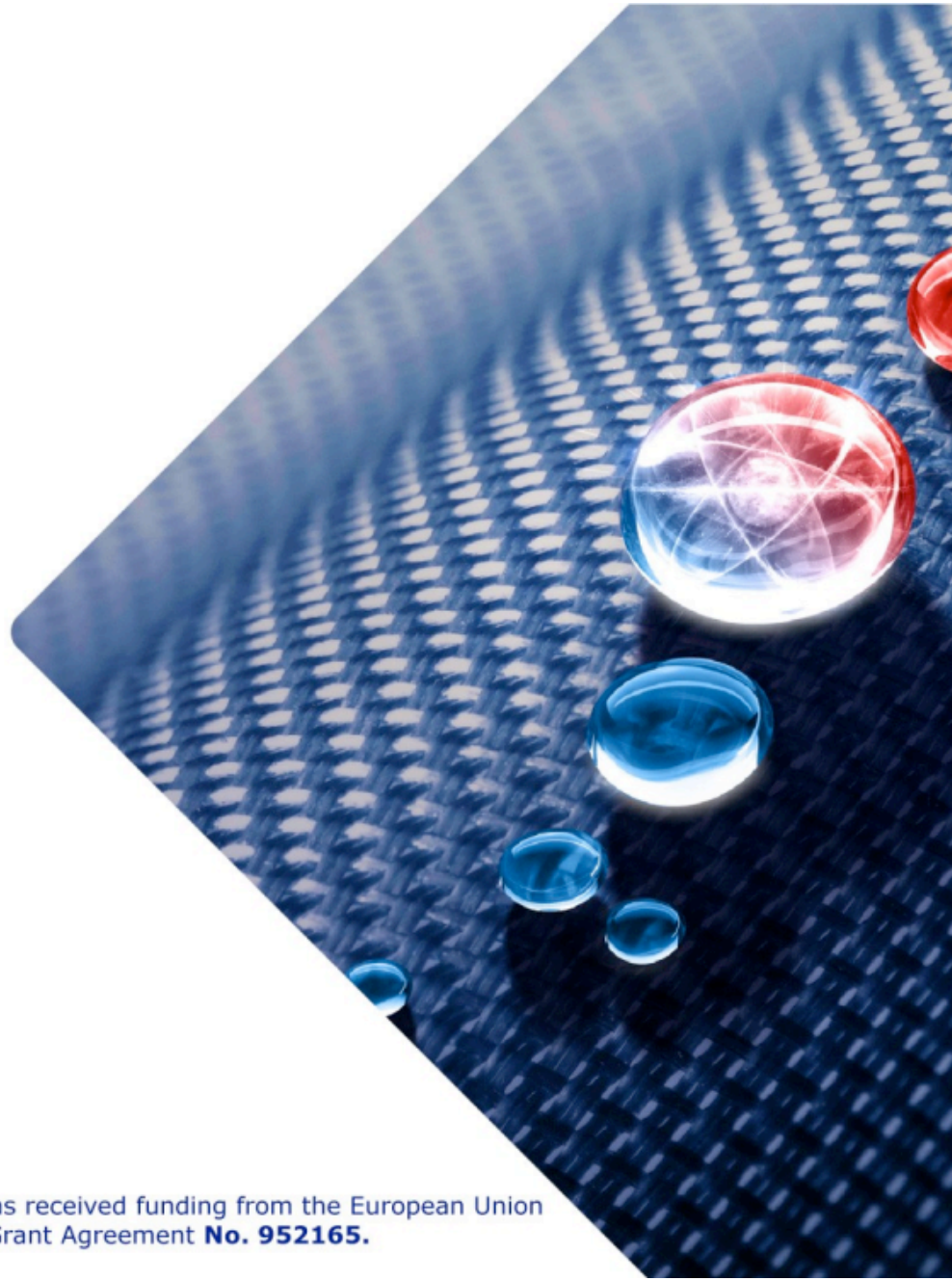
- Push optimization techniques to larger systems
- More work on transition metals
- Alternatives to fixed-node diffusion Monte Carlo

Other applications of quantum Monte Carlo methods

- Electronic structure calculations
- Strongly correlated systems (Hubbard, t-J, ...)
- Quantum spin systems (Ising, Heisenberg, XY, ...)
- Liquid-solid helium, liquid-solid interface, droplets
- Atomic clusters
- Nuclear structure
- Lattice gauge theory

Both zero (ground state) and finite temperature

Targeting Real chemical accuracy at the EXascale



Targeting Real Chemical Accuracy at the Exascale project has received funding from the European Union Horizoon 2020 research and innovation programme under Grant Agreement **No. 952165.**

Targeting Real chemical accuracy at the EXascale

Multideterminant wave functions

Anthony Scemama[1], Vijay Gopal Chilkuri[1], Ravindra Shinde[2], Claudia Filippi[2]

21/06/2022

[1] Lab. Chimie et Physique Quantiques, FERMI, UPS/CNRS, Toulouse (France)

[2] MESA+ Institute for Nanotechnology, Universiteit Twente, (The Netherlands)

Targeting Real Chemical Accuracy at the Exascale project has received funding from the European Union Horizoon 2020 research and innovation programme under Grant Agreement **No. 952165.**

Outline

Introduction

Reminders

- Atomic orbitals (AOs): χ_k. *Non-orthogonal* set of one-electron functions.

$$\chi_k(\mathrm{r}) = \sum_l P_{kl}(\mathrm{r}) e^{-\gamma_{kl}|\mathrm{r}|^p}$$

P : Spherical harmonics or Polynomial. $p = 1$: Slater, $p = 2$: Gaussian

- Molecular orbitals (MOs): LCAO. *Orthonormal* set of one-electron functions.

$$\phi_i(\mathrm{r}) = \sum_k C_{ik}\chi_k(\mathrm{r})$$

- Many different types of MOs: Hartree-Fock, Kohn-Sham, localized, natural, ...
- N-electron Wave function: Anti-symmetric product of MOs $\Longrightarrow$ Slater determinant

$$\Psi(\mathrm{r}_1, \mathrm{r}_2, \ldots, \mathrm{r}_N) = \begin{vmatrix} \phi_1(\mathrm{r}_1) & \phi_2(\mathrm{r}_1) & \ldots & \phi_N(\mathrm{r}_1) \\ \phi_1(\mathrm{r}_2) & \phi_2(\mathrm{r}_2) & \ldots & \phi_N(\mathrm{r}_2) \\ \vdots & & \ddots & \vdots \\ \phi_1(\mathrm{r}_N) & \phi_2(\mathrm{r}_N) & \ldots & \phi_N(\mathrm{r}_N) \end{vmatrix}$$

Orbitals

- MOs are linear combinations of AOs (LCAO)
- One can build as many MOs as AOs
- The space spanned by MOs is the same as the space spanned by AOs

- Hartree-Fock MOs are divided into occupied and virtual MOs
  - Occupied Hartree-Fock MOs: Orthonormal set of LCAOs which minimize the energy of a Slater determinant
  - Virtual Hartree-Fock MOs: The orthonormal complement of the Occupied MOs

Slater determinant

The Slater determinant can be rewritten by separating ↑- and ↓-spin electrons:

$$\Psi(\mathrm{r}_1, \mathrm{r}_2, \ldots, \mathrm{r}_{N_\uparrow}, \mathrm{r}_{N_\uparrow+1}, \ldots, \mathrm{r}_N) =$$

$$\begin{vmatrix} \phi_1(\mathrm{r}_1) & \phi_2(\mathrm{r}_1) & \ldots & \phi_{N_\uparrow}(\mathrm{r}_1) \\ \phi_1(\mathrm{r}_2) & \phi_2(\mathrm{r}_2) & \ldots & \phi_{N_\uparrow}(\mathrm{r}_2) \\ \vdots & & \ddots & \vdots \\ \phi_1(\mathrm{r}_{N_\uparrow}) & \phi_2(\mathrm{r}_{N_\uparrow}) & \ldots & \phi_{N_\uparrow}(\mathrm{r}_{N_\uparrow}) \end{vmatrix} \times \begin{vmatrix} \phi_1(\mathrm{r}_{N_\uparrow+1}) & \phi_2(\mathrm{r}_{N_\uparrow+1}) & \ldots & \phi_{N_\downarrow}(\mathrm{r}_{N_\uparrow+1}) \\ \phi_1(\mathrm{r}_{N_\uparrow+2}) & \phi_2(\mathrm{r}_{N_\uparrow+2}) & \ldots & \phi_{N_\downarrow}(\mathrm{r}_{N_\uparrow+2}) \\ \vdots & & \ddots & \vdots \\ \phi_1(\mathrm{r}_N) & \phi_2(\mathrm{r}_N) & \ldots & \phi_{N_\downarrow}(\mathrm{r}_N) \end{vmatrix}$$

$$= D_\uparrow(\mathrm{r}_1, \mathrm{r}_2, \ldots, \mathrm{r}_{N_\uparrow}) \times D_\downarrow(\mathrm{r}_{N_\uparrow+1}, \ldots, \mathrm{r}_N)$$

Electron correlation

$$\begin{aligned}\Psi^2 &= (D_\uparrow \times D_\downarrow)^2 \\ &= D_\uparrow^2 \times D_\downarrow^2\end{aligned}$$

- The N-electron density is the product of a density of $N_\uparrow$ $\uparrow$-spin electrons and a density of $N_\downarrow$ $\downarrow$-spin electrons.
- Mean-field approach: $\uparrow$-spin and $\downarrow$-spin electrons are statistically independent
- Although same-spin electrons are *not* statistically independent, the single-determinant model is said to be uncorrelated.

Electron correlation

We have seen that electron correlation can be introduced with a Jastrow factor:

$$\Psi(\mathbf{r}_1, \ldots, \mathbf{r}_N) = D_\uparrow(\mathbf{r}_1, \ldots, \mathbf{r}_{N_\uparrow}) \times D_\downarrow(\mathbf{r}_{N_\uparrow+1}, \ldots, \mathbf{r}_N) \times \exp\left(J(\mathbf{r}_1, \ldots, \mathbf{r}_N)\right)$$

with

$$J(\mathbf{r}_1, \ldots, \mathbf{r}_N) = \sum_{i<j}^{N} \frac{b_1 \left|\mathbf{r}_i - \mathbf{r}_j\right|}{1 + b_2 \left|\mathbf{r}_i - \mathbf{r}_j\right|} + \ldots$$

J couples $\uparrow$-spin and $\downarrow$-spin electrons, so

$$\Psi(\mathbf{r}_1, \ldots, \mathbf{r}_N)^2 \neq p_\uparrow(\mathbf{r}_1, \ldots, \mathbf{r}_{N_\uparrow}) \times p_\downarrow(\mathbf{r}_{N_\uparrow+1}, \ldots, \mathbf{r}_N)$$

and $\uparrow$-spin and $\downarrow$-spin electrons are correlated.

Correlation energy

$$E_{\text{cor}}[\Psi] = E[\Psi] - E_{\text{HF}}$$

N-electron basis functions

- Ψ is an N-electron function
- It can be expressed as a linear combination of N-electron functions

$$\Psi(\mathbf{r}_1, \ldots, \mathbf{r}_N) = \sum_{i=1}^{N_d} c_i \, D_i(\mathbf{r}_1, \ldots, \mathbf{r}_N)$$

- If the basis is infinitely large, the *exact* wave function can be obtained by finding the c_i which minimize the energy.

$$E(\Psi_1) \geq E(\Psi_m) \geq E(\Psi_{N_d}) \geq E(\Psi_\infty) = E_{\text{exact}}$$

with $1 \leq m \leq N_d$.

Slater determinants

A natural N-electron basis is the basis of all possible Slater determinants that can be built with a given set of M MOs:

$$\Psi = c_1 \begin{pmatrix} -- \\ -- \\ \uparrow\downarrow \end{pmatrix} + c_2 \begin{pmatrix} -- \\ \uparrow- \\ -\downarrow \end{pmatrix} + c_3 \begin{pmatrix} \uparrow- \\ -- \\ -\downarrow \end{pmatrix} + c_4 \begin{pmatrix} -- \\ -\downarrow \\ \uparrow- \end{pmatrix} +$$
$$c_5 \begin{pmatrix} -- \\ \uparrow\downarrow \\ -- \end{pmatrix} + c_6 \begin{pmatrix} \uparrow- \\ -\downarrow \\ -- \end{pmatrix} + c_7 \begin{pmatrix} -\downarrow \\ -- \\ \uparrow- \end{pmatrix} + c_8 \begin{pmatrix} -\downarrow \\ \uparrow- \\ -- \end{pmatrix} + c_9 \begin{pmatrix} \uparrow\downarrow \\ -- \\ -- \end{pmatrix}$$

Each basis function is antisymmetric $\Longrightarrow$ Ψ is antisymmetric

The size of the basis grows fast:

$$N_d = \left(\frac{M!}{N_\uparrow!(M-N_\uparrow)!}\right) \times \left(\frac{M!}{N_\downarrow!(M-N_\downarrow)!}\right)$$

Example

18 electrons in 111 orbitals:
$N_d = 2.5 \times 10^{25}$ determinants.

$$\begin{aligned}\Psi(\mathbf{r}_1,\ldots,\mathbf{r}_N) &= c_1\, D_{1\uparrow}(\mathbf{r}_1,\ldots,\mathbf{r}_{N_\uparrow})\, D_{1\downarrow}(\mathbf{r}_{N_\uparrow+1},\ldots,\mathbf{r}_N) + \\ &\quad c_2\, D_{2\uparrow}(\mathbf{r}_1,\ldots,\mathbf{r}_{N_\uparrow})\, D_{2\downarrow}(\mathbf{r}_{N_\uparrow+1},\ldots,\mathbf{r}_N)\end{aligned}$$

$\Psi(\mathbf{r}_1,\ldots,\mathbf{r}_N)^2 \neq p_\uparrow(\mathbf{r}_1,\ldots,\mathbf{r}_{N_\uparrow}) \times p_\downarrow(\mathbf{r}_{N_\uparrow+1},\ldots,\mathbf{r}_N) \Longrightarrow$ electron correlation.

Configuration state functions

- The exact wave function is an eigenfunction of the spin operator $\hat{S}^2$
- Slater determinants are eigenfunctions of $\hat{S}_z$, but not of $\hat{S}^2$
- To obtain Ψ eigenfunction of $\hat{S}^2$, one needs to have in the determinant set all possible spin flips in open shells

$$\begin{pmatrix} \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \\ \uparrow\uparrow \end{pmatrix} = a \begin{pmatrix} -\downarrow \\ -\downarrow \\ \uparrow- \\ \uparrow- \\ \uparrow\downarrow \end{pmatrix} + b \begin{pmatrix} \uparrow- \\ -\downarrow \\ -\downarrow \\ \uparrow- \\ \uparrow\downarrow \end{pmatrix} + c \begin{pmatrix} \uparrow- \\ -\downarrow \\ \uparrow- \\ -\downarrow \\ \uparrow\downarrow \end{pmatrix} + d \begin{pmatrix} \uparrow- \\ \uparrow- \\ -\downarrow \\ -\downarrow \\ \uparrow\downarrow \end{pmatrix} + e \begin{pmatrix} -\downarrow \\ \uparrow- \\ \uparrow- \\ -\downarrow \\ \uparrow\downarrow \end{pmatrix} + f \begin{pmatrix} -\downarrow \\ \uparrow- \\ -\downarrow \\ \uparrow- \\ \uparrow\downarrow \end{pmatrix}$$

Configuration state functions

- Configuration state functions (CSF): Linear combinations of Slater determinants, which are eigenfunctions of S^2:

$$
\begin{pmatrix} \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \\ \uparrow\uparrow \end{pmatrix}
= A \times \frac{1}{2}\left[
\begin{pmatrix} \uparrow\!\!\uparrow - \\ - \downarrow\!\!\downarrow \\ - \downarrow\!\!\downarrow \\ \uparrow\!\!\uparrow - \\ \uparrow\!\!\downarrow \end{pmatrix}
+
\begin{pmatrix} - \downarrow\!\!\downarrow \\ \uparrow\!\!\uparrow - \\ \uparrow\!\!\uparrow - \\ - \downarrow\!\!\downarrow \\ \uparrow\!\!\downarrow \end{pmatrix}
-
\begin{pmatrix} \uparrow\!\!\uparrow - \\ - \downarrow\!\!\downarrow \\ \uparrow\!\!\uparrow - \\ - \downarrow\!\!\downarrow \\ \uparrow\!\!\downarrow \end{pmatrix}
-
\begin{pmatrix} - \downarrow\!\!\downarrow \\ \uparrow\!\!\uparrow - \\ - \downarrow\!\!\downarrow \\ \uparrow\!\!\uparrow - \\ \uparrow\!\!\downarrow \end{pmatrix}
\right]
$$

$$
+ B \times \frac{\sqrt{3}}{6}\left[
-2\begin{pmatrix} - \downarrow\!\!\downarrow \\ - \downarrow\!\!\downarrow \\ \uparrow\!\!\uparrow - \\ \uparrow\!\!\uparrow - \\ \uparrow\!\!\downarrow \end{pmatrix}
+
\begin{pmatrix} \uparrow\!\!\uparrow - \\ - \downarrow\!\!\downarrow \\ - \downarrow\!\!\downarrow \\ \uparrow\!\!\uparrow - \\ \uparrow\!\!\downarrow \end{pmatrix}
+
\begin{pmatrix} \uparrow\!\!\uparrow - \\ - \downarrow\!\!\downarrow \\ \uparrow\!\!\uparrow - \\ - \downarrow\!\!\downarrow \\ \uparrow\!\!\downarrow \end{pmatrix}
-2\begin{pmatrix} \uparrow\!\!\uparrow - \\ \uparrow\!\!\uparrow - \\ - \downarrow\!\!\downarrow \\ - \downarrow\!\!\downarrow \\ \uparrow\!\!\downarrow \end{pmatrix}
+
\begin{pmatrix} - \downarrow\!\!\downarrow \\ \uparrow\!\!\uparrow - \\ \uparrow\!\!\uparrow - \\ - \downarrow\!\!\downarrow \\ \uparrow\!\!\downarrow \end{pmatrix}
+
\begin{pmatrix} - \downarrow\!\!\downarrow \\ \uparrow\!\!\uparrow - \\ - \downarrow\!\!\downarrow \\ \uparrow\!\!\uparrow - \\ \uparrow\!\!\downarrow \end{pmatrix}
\right]
$$

- The CSF basis is smaller than the determinant basis: one selects only basis functions with the desired $\langle \hat{S}^2 \rangle$

Configuration interaction (CI)

- Ψ is a linear combination of Slater determinants (or CSFs)
- The energy is minimized by diagonalizing the Hamiltonian in the basis of Slater determinants (or CSFs)

$$E = \frac{\langle \Psi | H | \Psi \rangle}{\langle \Psi | \Psi \rangle}$$

CI methods

Differ by the choice of the basis:

- Full configuration interaction (FCI): All possible Slater determinants. $\mathcal{O}(N!)$
- CI with Single and Double substitutions (CISD): No more than one or two MOs differ from the Hartree-Fock determinant. $\mathcal{O}(N_o^2 N_v^2)$
- Complete Active Space (CAS): Only a subset of m MOs can be substituted from the Hartree-Fock determinant. $\mathcal{O}(m!)$

Dynamic *vs* static correlation

- Dynamic : short-range effects due to the Coulomb hole. Hartree-Fock qualitatively correct, many small contributions.
- Static : near-degeneracies. Hartree-Fock qualitatively incorrect, few large contributions.

Examples

- CH_4, 6-31G: 38×10^6 determinants. Dynamic

E_{HF}	-40.1805 a.u
E_{FCI}	-40.3011 a.u

- Dissociated H_2, STO-6G: 2 determinants (1 CSF). Static

$$\Psi(1,2) = \frac{1}{\sqrt{2}}\Big(\phi_1(1)\phi_1(2) - \phi_2(1)\phi_2(2)\Big)$$

E_{HF}	-0.5572 a.u	ϵ_1 = -0.08619 a.u
E_{FCI}	-0.9421 a.u	ϵ_2 = -0.08619 a.u

Dynamic *vs* static correlation

- Dynamic: Well described by a Jastrow factor
- Static: Well described by a linear combination of Slater determinants
- Optimal representation:

$$\Psi = \left(\sum_i c_i D_i \right) \exp\left(J\right)$$

- Interplay between static and dynamic correlation: c_i should be optimized in the presence of $\exp(J)$.

Size consistency

Size consistency: Strict separability. When two systems A and B are far enough to not interact:

$$E[\Psi_{A\ldots B}] = E[\Psi_A] + E[\Psi_B]$$

- If the MOs are localized on fragments A and B, determinants can be written as

$$|K^{A\ldots B}\rangle = |I^A J^B\rangle = |I^A\rangle \otimes |J^B\rangle$$

- $\mathrm{FCI}^{\mathrm{AB}}$ is built as the tensor product of $\mathrm{FCI}^{\mathrm{A}}$ and $\mathrm{FCI}^{\mathrm{B}}$

$$\Psi_{A\ldots B} = \sum_K c_K |K^{A\ldots B}\rangle = \left(\sum_I c_I^A |I^A\rangle\right) \otimes \left(\sum_J c_J^B |J^B\rangle\right)$$

CI is usually *not size-consistent*. Example: CISD

The CISD space for $A \ldots B$ is *not* the tensor product of the spaces of A and B

- $|I^A\rangle = \hat{T}_{ij}^{kl}|\mathrm{HF}^A\rangle \quad |J^B\rangle = \hat{T}_{mn}^{pq}|\mathrm{HF}^B\rangle$
- $|I^A J^B\rangle = \hat{T}_{ij}^{kl}\hat{T}_{mn}^{pq}|\mathrm{HF}^A\mathrm{HF}^B\rangle$
- $|K^{A\ldots B}\rangle = \hat{T}_{ijmn}^{klpq}|\mathrm{HF}^{A\ldots B}\rangle$: quadruple excitation, missing in CISD space

The size-consistency error is positive:

$$E[\Psi_{\mathrm{CISD}}^{A\ldots B}] \geq E[\Psi_{\mathrm{CISD}}^{A}] + E[\Psi_{\mathrm{CISD}}^{B}]$$

Size-consistent particular cases

Hartree-Fock FCI CAS-SCF

Configuration Interaction

Configuration Interaction

- Define an orthonormal basis of N-electron functions: Slater determinants or CSFs $\{|I\rangle\}$
- Express the wave function on this basis: $\langle I|\Psi\rangle = c_I$

$$|\Psi\rangle = \sum_I c_I |I\rangle$$

- The energy is given by

$$E[\Psi] = \frac{\langle\Psi|\hat{H}|\Psi\rangle}{\langle\Psi|\Psi\rangle}$$

- The optimal c_I are obtained when $|\Psi\rangle$ is an eigenfunction of H, and E is the corresponding eigenvalue

Diagonalization of the Hamiltonian

- $\langle I|J\rangle = \delta_{IJ}$, because MOs are orthonormal.
- $H_{IJ} = \langle I|\hat{H}|J\rangle$
- $\langle \Psi|\Psi\rangle = \sum_{IJ} c_I\, c_J\, \delta_{IJ} = \sum_I c_I^2 = 1$

$$E[\Psi] = \sum_{IJ} c_I\, c_J\, H_{IJ}$$

- When N_d is small $< 10^4$, direct diagonalization of H
- When N_d is large, Davidson's algorithm to extract the desired roots.
  Iterative computation of $|W\rangle = \sum_I w_I|I\rangle = \sum_I |I\rangle\langle I|H|\Psi\rangle$ (power method).

Slater-Condon rules

Thanks to $\langle I|J\rangle = \delta_{IJ}$:

- Diagonal terms:

$$H_{II} = \sum_i \langle i|\hat{h}|i\rangle + \sum_{ij} \langle ij||ij\rangle$$

- $|J\rangle = \hat{T}_p^r|I\rangle$: $|I\rangle$ and $|J\rangle$ differ by one MO:

$$H_{IJ} = \langle p|\hat{h}|r\rangle + \sum_i \langle pi||ri\rangle$$

- $|J\rangle = \hat{T}_{pq}^{rs}|I\rangle$: $|I\rangle$ and $|J\rangle$ differ by two MOs:

$$H_{IJ} = \langle pq||rs\rangle$$

- $|I\rangle$ and $|J\rangle$ differ by more than two MOs:

$$H_{IJ} = 0$$

Computational aspects

There are:

- $\mathcal{O}(N^4)$ two-electron integrals
- N_d Slater determinants

Algorithms

- Integral-driven
  - Loop over integrals
  - Add the contributions to $|W\rangle$
- Determinant-driven
  - Loop over determinants
  - Usually, $N_d \ggg \mathcal{O}(N^4)$, so less efficient than determinant-driven

Excited states

- Same symmetry:
  Obtained as different eigenvectors of H. Expanded on the same set of determinants:
  $$\Psi^{(k)} = \sum_I c_I^{(k)} |I\rangle$$
- Lowest states of different symmetries:
  H is block-diagonal:
  - Pick only determinants of the desired symmetry
  - Obtain the ground state

  Expanded on different sets of determinants:
  $$\Psi^{(k)} = \sum_I c_I^{(k)} |I^{(k)}\rangle$$

Summary of CI methods

- All CI methods are approximations of the FCI
- They differ by the choice of the Slater determinant basis

- CIS, CISD, CISDT, CISDTQ, ... : Number of differences *wrt* Hartree-Fock (dynamic)
- CAS, RAS, GAS, ... : CI in an active space (static)

- MR-CI : active space + CISD for each reference (static + dynamic)
- MP2, CAS-PT2, dynamic correlation is computed with perturbation theory: cheaper than CI

Selected Configuration Interaction

Full CI

FCI: Exact solution of $\hat{H}\Psi = E\Psi$ in a complete basis of Slater determinants

- The determinant basis is derived from the one-electron basis set
- Only approximation : one-electron basis-set incompleteness
- Intractable : $\mathcal{O}(N!)$ scaling
- All the post-Hartree-Fock methods are *approximations* of the FCI within the same basis set

Largest CAS-SCF (2017)

THE JOURNAL OF CHEMICAL PHYSICS **147**, 184111 (2017)

Pushing configuration-interaction to the limit: Towards massively parallel MCSCF calculations

Konstantinos D. Vogiatzis,[1,a),b)] Dongxia Ma,[1,c)] Jeppe Olsen,[2] Laura Gagliardi,[1,a)] and Wibe A. de Jong[3,a)]
[1]*Department of Chemistry, Minnesota Supercomputing Institute, and Chemical Theory Center, University of Minnesota, 207 Pleasant Street Southeast, Minneapolis, Minnesota 55455-0431, USA*
[2]*Department of Chemistry, Aarhus University, Langelandsgade 140, 8000 Aarhus C, Denmark*
[3]*Computational Research Division, Lawrence Berkeley National Laboratory, Berkeley, California 94720, USA*

(Received 12 June 2017; accepted 20 October 2017; published online 14 November 2017)

A new large-scale parallel multiconfigurational self-consistent field (MCSCF) implementation in the open-source NWChem computational chemistry code is presented. The generalized active space approach is used to partition large configuration interaction (CI) vectors and generate a sufficient number of batches that can be distributed to the available cores. Massively parallel CI calculations with large active spaces can be performed. The new parallel MCSCF implementation is tested for the chromium trimer and for an active space of 20 electrons in 20 orbitals, which can now routinely be performed. Unprecedented CI calculations with an active space of 22 electrons in 22 orbitals for the pentacene systems were performed and a single CI iteration calculation with an active space of 24 electrons in 24 orbitals for the chromium tetramer was possible. The chromium tetramer corresponds to a CI expansion of one trillion Slater determinants (914 058 513 424) and is the largest conventional CI calculation attempted up to date. *Published by AIP Publishing.*
https://doi.org/10.1063/1.4989858

The Full CI Hamiltonian is very sparse

- Each row $\langle I|$ of H has non-zeros when $|J\rangle$ differs by less than 3 MOs (Slater-Condon rules)
- Each row has at most $\mathcal{O}(N_o^2 N_v^2)$ non-zero elements
- H is symmetric, the same applies to columns
- Davidson's algorithm involves computing $\langle I|\hat{H}|\Psi\rangle$
- Sparse matrix-vector multiplication: $\mathcal{O}(N_d \times N_o^2 N_v^2)$

FCI has seen a breakthrough in 2007-2012

- DMRG[a]
- FCI-QMC : Stochastic solution of FCI equations.[b]
  - First row diatomics cc-pV5Z.[c]
- Selected Configuration Interaction
  - Scaling is still $\mathcal{O}(N!)$, but pre-factor is killed.
  - Much larger active spaces are possible today

[a] G. K.-L. Chan , arXiv:0711.1398 (2007)
[b] G.H. Booth , J. of Chem. Phys. 131, 054106 (2009).
[c] D. Cleland , J. Chem. Theory Comput. 8, 4138 (2012)

Modern CAS-SCF

OpenMolcas: From Source Code to Insight

Preprint submitted on 06.06.2019, 09:41 and posted on 06.06.2019, 18:08 by Ignacio Fdez. Galván, Morgane Vacher, Ali Alavi, Celestino Angeli, Jochen Autschbach, Jie J. Bao, Sergey I. Bokarev, Nikolay A. Bogdanov, Rebecca K. Carlson, Liviu F. Chibotaru, Joel Creutzberg, Nike Dattani, Mickaël G. Delcey, Sijia Dong, Andreas Dreuw, Leon Freitag, Luis Manuel Frutos, Laura Gagliardi, Frédéric Gendron, Angelo Giussani, Leticia Gonzalez, Gilbert Grell, Meiyuan Guo, Chad E. Hoyer, Marcus Johansson, Sebastian Keller, Stefan knecht, Goran Kovačević, Erik Källman, Giovanni Li Manni, Marcus Lundberg, Yingjin Ma, Sebastian Mai, João Pedro Malhado, Per Åke Malmqvist, Philipp Marquetand, Stefanie A. Mewes, Jesper Norell, Massimo Olivucci, Markus Oppel, Quan Manh Phung, Kristine Pierloot, Felix Plasser, Markus Reiher, Andrew M. Sand, Igor Schapiro, Prachi Sharma, Christopher J. Stein, Lasse Kragh Sørensen, Donald G. Truhlar, Mihkel Ugandi, Liviu Ungur, Alessio Valentini, Steven Vancoillie, Valera Veryazov, Oskar Weser, Per-Olof Widmark, Sebastian Wouters, J. Patrick Zobel, Roland Lindh

In this article we describe the OpenMolcas environment and invite the computational chemistry community to collaborate. The open-source project already includes a large number of new developments realized during the transition from the commercial MOLCAS product to the open-source platform. The paper initially describes the technical details of the new software development platform. This is followed by brief presentations of many new methods, implementations, and features of the OpenMolcas program suite. These developments include novel wave function methods such as stochastic complete active space self-consistent field, density matrix renormalization group (DMRG) methods, and hybrid multiconfigurational wave function and density functional theory models. Some of these implementations include an array of additional options and functionalities. The paper proceeds and describes developments related to explorations of potential energy surfaces. Here we present methods for the optimization of conical intersections, the simulation of adiabatic and nonadiabatic molecular dynamics and interfaces to tools for semiclassical and quantum mechanical nuclear dynamics.

CATEGORIES

- Computational Chemistry and Modeling
- Theory - Computational
- Chemoinformatics - Computational Chemistry
- Spectroscopy (Physical Chem.)
- Physical and Chemical Properties

KEYWORD(S)

Multiconfigurational methods
Molecular dynamics
Wave function analysis
Spectroscopy
Basis sets

JCTC Journal of Chemical Theory and Computation

Article

Cite This: J. Chem. Theory Comput. 2017, 13, 5468-5478

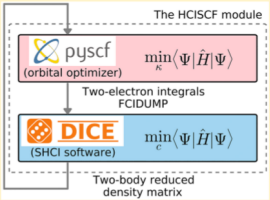
pubs.acs.org/JCTC

Cheap and Near Exact CASSCF with Large Active Spaces

James E. T. Smith,* Bastien Mussard, Adam A. Holmes, and Sandeep Sharma*

Department of Chemistry and Biochemistry, University of Colorado Boulder, Boulder, Colorado 80309, United States

ABSTRACT: We use the recently developed Heat-bath Configuration Interaction (HCI) algorithm as an efficient active space solver to perform multiconfiguration self-consistent field calculations (HCISCF) with large active spaces. We give a detailed derivation of the theory and show that difficulties associated with non-variationality of the HCI procedure can be overcome by making use of the Lagrangian formulation to calculate the HCI relaxed two-body reduced density matrix. HCISCF is then used to study the electronic structure of butadiene, pentacene, and Fe−porphyrin. One of the most striking results of our work is that the converged active space orbitals obtained from HCISCF are relatively insensitive to the accuracy of the HCI calculation. This allows us to obtain nearly converged CASSCF energies with an estimated error of less than 1 mHa using the orbitals obtained from the HCISCF procedure in which the integral transformation is the dominant cost. For example, an HCISCF calculation on the Fe−porphyrin model complex with an active space of (44e, 44o) took only 412 s per iteration on a single node containing 28 cores, out of which 185 s was spent in the HCI calculation and the remaining 227 s was used mainly for integral transformation. Finally, we also show that active space orbitals can be optimized using HCISCF to substantially speed up the convergence of the HCI energy to the Full CI limit because HCI is not invariant to unitary transformations within the active space.



Selected Configuration Interaction (SCI)

- Select determinants on-the-fly
  - with perturbation theory (CIPSI[1])
  - or based only on the matrix elements of $\hat{H}$ (SHCI[2])
- Target spaces : Full-CI, MR-CISD, large CAS, ...
- Use PT2 to estimate the missing part

[1] B. Huron, J.P. Malrieu, and P. Rancurel, J. Chem. Phys. 58, 5745 (1973).
[2] A.A. Holmes, C.J. Umrigar, and S. Sharma, J. Chem. Phys. 147, 164111 (2017)

CIPSI Algorithm

Start with $\mathcal{D}_0 = \{|\mathrm{HF}\rangle\}$ and $|\Psi_0\rangle = |\mathrm{HF}\rangle$.

CIPSI Algorithm

Start with $\mathcal{D}_0 = \{|\mathrm{HF}\rangle\}$ and $|\Psi_0\rangle = |\mathrm{HF}\rangle$.

1. $\forall\, |i\rangle \in \{\hat{T}_{\mathrm{SD}}|\Psi_n\rangle\} \setminus \{\mathcal{D}_n\}$, compute $e_i = \frac{\langle i|\mathcal{H}|\Psi_n\rangle^2}{E(\Psi_n) - \langle i|\mathcal{H}|i\rangle}$

CIPSI Algorithm

Start with $\mathcal{D}_0 = \{|\mathrm{HF}\rangle\}$ and $|\Psi_0\rangle = |\mathrm{HF}\rangle$.

1. $\forall |i\rangle \in \{\hat{T}_{\mathrm{SD}}|\Psi_n\rangle\} \setminus \{\mathcal{D}_n\}$, compute $e_i = \frac{\langle i|\mathcal{H}|\Psi_n\rangle^2}{E(\Psi_n) - \langle i|\mathcal{H}|i\rangle}$
2. if $|e_i| > \epsilon_n$, select $|i\rangle$

CIPSI Algorithm

Start with $\mathcal{D}_0 = \{|\mathrm{HF}\rangle\}$ and $|\Psi_0\rangle = |\mathrm{HF}\rangle$.

1. $\forall |i\rangle \in \{\hat{T}_{\mathrm{SD}}|\Psi_n\rangle\} \setminus \{\mathcal{D}_n\}$, compute $e_i = \frac{\langle i|\mathcal{H}|\Psi_n\rangle^2}{E(\Psi_n) - \langle i|\mathcal{H}|i\rangle}$
2. if $|e_i| > \epsilon_n$, select $|i\rangle$
3. Estimated energy : $E(\Psi_n) + E_{\mathrm{PT2}}(\Psi_n) = E(\Psi_n) + \sum_i e_i$

CIPSI Algorithm

Start with $\mathcal{D}_0 = \{|\mathrm{HF}\rangle\}$ and $|\Psi_0\rangle = |\mathrm{HF}\rangle$.

1. $\forall |i\rangle \in \{\hat{T}_{\mathrm{SD}}|\Psi_n\rangle\} \setminus \{\mathcal{D}_n\}$, compute $e_i = \frac{\langle i|\mathcal{H}|\Psi_n\rangle^2}{E(\Psi_n) - \langle i|\mathcal{H}|i\rangle}$
2. if $|e_i| > \epsilon_n$, select $|i\rangle$
3. Estimated energy : $E(\Psi_n) + E_{\mathrm{PT2}}(\Psi_n) = E(\Psi_n) + \sum_i e_i$
4. $\mathcal{D}_{n+1} = \mathcal{D}_n \cup \{\cup_{i(\mathrm{selected})}|i\rangle\}$

CIPSI Algorithm

Start with $\mathcal{D}_0 = \{|\mathrm{HF}\rangle\}$ and $|\Psi_0\rangle = |\mathrm{HF}\rangle$.

1. $\forall |i\rangle \in \{\hat{T}_{\mathrm{SD}}|\Psi_n\rangle\} \setminus \{\mathcal{D}_n\}$, compute $e_i = \frac{\langle i|\mathcal{H}|\Psi_n\rangle^2}{E(\Psi_n) - \langle i|\mathcal{H}|i\rangle}$
2. if $|e_i| > \epsilon_n$, select $|i\rangle$
3. Estimated energy : $E(\Psi_n) + E_{\mathrm{PT2}}(\Psi_n) = E(\Psi_n) + \sum_i e_i$
4. $\mathcal{D}_{n+1} = \mathcal{D}_n \cup \{\cup_{i(\mathrm{selected})}|i\rangle\}$
5. Minimize $E(\Psi_{n+1})$ (Davidson),
   $\Psi_{n+1} = \Psi_n + \sum_{i(\mathrm{selected})} c_i|i\rangle$

CIPSI Algorithm

Start with $\mathcal{D}_0 = \{|\mathrm{HF}\rangle\}$ and $|\Psi_0\rangle = |\mathrm{HF}\rangle$.

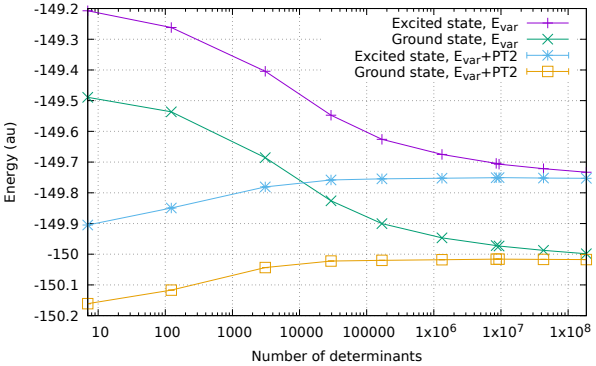
1. $\forall |i\rangle \in \{\hat{T}_{\mathrm{SD}}|\Psi_n\rangle\} \setminus \{\mathcal{D}_n\}$, compute $e_i = \frac{\langle i|\mathcal{H}|\Psi_n\rangle^2}{E(\Psi_n) - \langle i|\mathcal{H}|i\rangle}$
2. if $|e_i| > \epsilon_n$, select $|i\rangle$
3. Estimated energy : $E(\Psi_n) + E_{\mathrm{PT2}}(\Psi_n) = E(\Psi_n) + \sum_i e_i$
4. $\mathcal{D}_{n+1} = \mathcal{D}_n \cup \{\cup_{i(\mathrm{selected})}|i\rangle\}$
5. Minimize $E(\Psi_{n+1})$ (Davidson),
   $\Psi_{n+1} = \Psi_n + \sum_{i(\mathrm{selected})} c_i|i\rangle$
6. Choose $\epsilon_{n+1} < \epsilon_n$

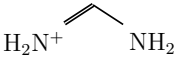
CIPSI Algorithm

Start with $\mathcal{D}_0 = \{|\mathrm{HF}\rangle\}$ and $|\Psi_0\rangle = |\mathrm{HF}\rangle$.

1. $\forall |i\rangle \in \{\hat{T}_{\mathrm{SD}}|\Psi_n\rangle\} \setminus \{\mathcal{D}_n\}$, compute $e_i = \frac{\langle i|\mathcal{H}|\Psi_n\rangle^2}{E(\Psi_n) - \langle i|\mathcal{H}|i\rangle}$
2. if $|e_i| > \epsilon_n$, select $|i\rangle$
3. Estimated energy : $E(\Psi_n) + E_{\mathrm{PT2}}(\Psi_n) = E(\Psi_n) + \sum_i e_i$
4. $\mathcal{D}_{n+1} = \mathcal{D}_n \cup \{\cup_{i(\mathrm{selected})}|i\rangle\}$
5. Minimize $E(\Psi_{n+1})$ (Davidson),
   $\Psi_{n+1} = \Psi_n + \sum_{i(\mathrm{selected})} c_i|i\rangle$
6. Choose $\epsilon_{n+1} < \epsilon_n$
7. Iterate

Convergence of CIPSI

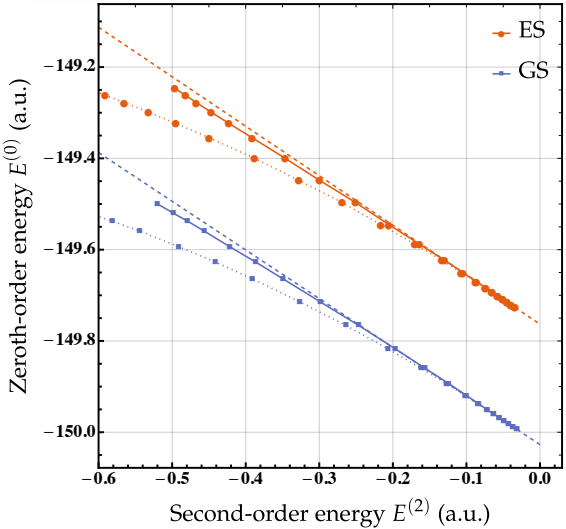




aug-cc-pVDZ

- When $N_d = N_{\mathrm{FCI}}$, $E_{\mathrm{PT2}} = 0$, CI is solved *exactly*.
- Every CI problem can be solved by iterative perturbative selection

Extrapolated FCI energies



exFCI : Extrapolate $E = f(E_{\mathrm{PT2}})$ at $E_{\mathrm{PT2}} = 0$, estimates the complete CI solution.

Consistent energies

The error of $E_{\mathrm{FCI}} \sim E + E_{\mathrm{PT2}}$ is proportional to E_{PT2}

$$E_{\mathrm{FCI}} = E + (1+\alpha)\, E_{\mathrm{PT2}}$$

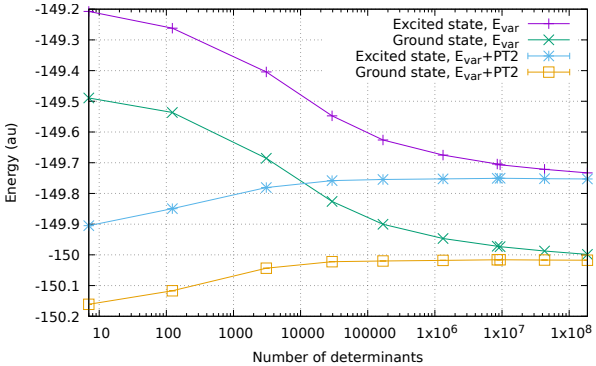
For 2 states

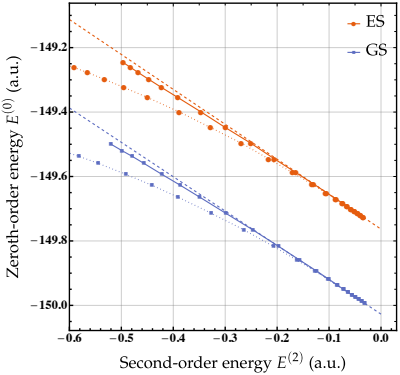
$$\begin{aligned} E^{(1)}_{\mathrm{FCI}} &= E^{(1)} + (1+\alpha^{(1)})\, E^{(1)}_{\mathrm{PT2}} \\ E^{(2)}_{\mathrm{FCI}} &= E^{(2)} + (1+\alpha^{(2)})\, E^{(2)}_{\mathrm{PT2}} \end{aligned}$$

If $\alpha^{(1)} = \alpha^{(2)}$ and $E^{(1)}_{\mathrm{PT2}} = E^{(2)}_{\mathrm{PT2}}$

$$E^{(2)}_{\mathrm{FCI}} - E^{(1)}_{\mathrm{FCI}} = E^{(2)} - E^{(1)}$$

Consistent energies





- $-(1+\alpha)$ is the slope of the extrapolation curve
- $\alpha^{(1)} \sim \alpha^{(2)}$ can be obtained using state-average orbitals

Multideterminant QMC

Wave function

- In a CI calculation:

$$|\Psi\rangle = \sum_I c_I |I\rangle$$

- In QMC:

$$\begin{aligned}\Psi(\mathrm{r}_1, \ldots, \mathrm{r}_N) &= \left(\sum_k c_k D_k(\mathrm{r}_1, \ldots, \mathrm{r}_N)\right) e^{J(\mathrm{r}_1, \ldots, \mathrm{r}_N)} \\ &= \sum_k c_k \left(D_K(\mathrm{r}_1, \ldots, \mathrm{r}_N) e^{J(\mathrm{r}_1, \ldots, \mathrm{r}_N)}\right)\end{aligned}$$

Computationally expensive

- We need to evaluate *all* the Slater determinants at each MC step
- Compacting the wave function is desirable

Evaluation of a determinant

1. Build the Slater Matrix $A_{ij} = \phi_j(\mathrm{r}_i)$:

$$\mathrm{A} = \begin{bmatrix} \phi_1(\mathrm{r}_1) & \phi_2(\mathrm{r}_1) & \dots & \phi_N(\mathrm{r}_1) \\ \phi_1(\mathrm{r}_2) & \phi_2(\mathrm{r}_2) & \dots & \phi_N(\mathrm{r}_2) \\ \vdots & & \ddots & \vdots \\ \phi_1(\mathrm{r}_N) & \phi_2(\mathrm{r}_N) & \dots & \phi_N(\mathrm{r}_N) \end{bmatrix}$$

2. LU factorization (`dgetrf`) : $\mathrm{A} = \mathrm{P}\,\mathrm{L}\,\mathrm{U}$, costs $\mathcal{O}(N^3)$
3. $\det A = \prod_i U_{ii}$

Evaluation of the derivatives of a determinant

$$\frac{\nabla_i(\det A)}{\det A} = \sum_j \nabla_i \phi_j(\mathrm{r}_i).A^{-1}_{ji}$$

$$\frac{\Delta_i(\det A)}{\det A} = \sum_j \Delta_i \phi_j(\mathrm{r}_i).A^{-1}_{ji}$$

Inverse of A (`dgetri`) : costs $\mathcal{O}(N^3)$

Shermann-Morrison-Woodbury

A and A^{-1} are known, u and v are column vectors,

$$\left(A + uv^\dagger\right)^{-1} = A^{-1} - \frac{A^{-1}uv^\dagger A^{-1}}{1 + v^\dagger A^{-1}u}.$$

Costs $\mathcal{O}(N^2)$.

Single orbital change:

$$u = \begin{bmatrix} \phi_k(r_1) - \phi_l(r_1) \\ \vdots \\ \phi_k(r_N) - \phi_l(r_N) \end{bmatrix}, v = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix},$$

Computational scaling

$$\Psi(\mathrm{r}) = \sum_{k}^{N_d} c_k D_k = \sum_{i}^{N_{d\uparrow}} \sum_{j}^{N_{d\downarrow}} C_{ij}\, D_{i\uparrow}(\mathrm{r}_\uparrow)\, D_{j\downarrow}(\mathrm{r}_\downarrow)$$

- $\mathrm{D}_\uparrow(\mathrm{r}_\uparrow)$: vector of $N_{d\uparrow}$ elements
- $\mathrm{D}_\downarrow(\mathrm{r}_\downarrow)$: vector of $N_{d\downarrow}$ elements
- C : $N_{d\uparrow} \times N_{d\downarrow}$ matrix. The matrix contains N_d non-zero elements

C is constant in a QMC calculation $\Longrightarrow$ preprocessing.

Computational scaling

At every MC step, we need to evaluate:

$$
\begin{aligned}
\Psi &= (D_\uparrow{}^\dagger(C)D_\downarrow) \\
\nabla_i\Psi &= \nabla_i D_\uparrow{}^\dagger.(CD_\downarrow) \text{ or } (D_\uparrow{}^\dagger C).\nabla_i D_\downarrow \\
\Delta_i\Psi &= \Delta_i D_\uparrow{}^\dagger.(CD_\downarrow) \text{ or } (D_\uparrow{}^\dagger C).\Delta_i D_\downarrow \\
V_{\text{pseudo}}^{\text{non-loc}}\Psi &= V_{\text{pseudo}}^{\text{non-loc}} D_\uparrow{}^\dagger.(CD_\downarrow) \text{ or } (D_\uparrow{}^\dagger C).V_{\text{pseudo}}^{\text{non-loc}} D_\downarrow
\end{aligned}
$$

($\uparrow$ electrons and $\downarrow$ electrons)

Computational scaling

$\mathcal{O}(N_{d\uparrow} \times N^2_{\text{elec}\uparrow})$

$\mathrm{D}_\uparrow$ and $\mathrm{D}_\downarrow$, $\nabla\mathrm{D}_\uparrow$ and $\nabla\mathrm{D}_\downarrow$, $\Delta\mathrm{D}_\uparrow$ and $\Delta\mathrm{D}_\downarrow$

$\mathcal{O}(N_d)$, tiny prefactor

- Sparse vector-matrix product $\mathrm{D}_\uparrow^\dagger.\mathrm{C}$: N_d operations, returns a $N_{d\downarrow}$ vector

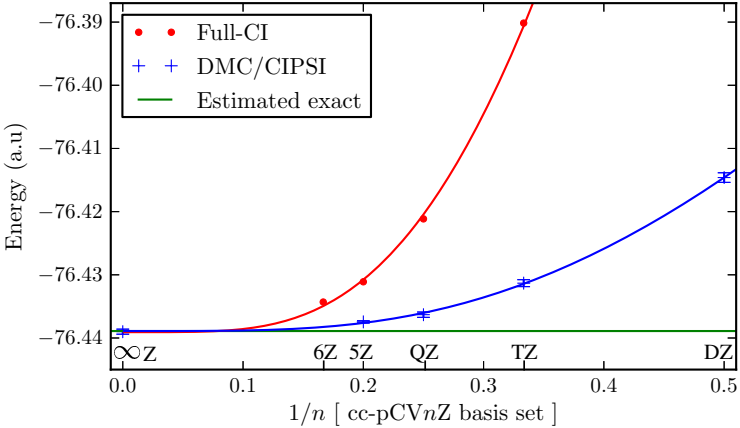
$\mathcal{O}(N_{\text{elec}\downarrow} \times N_{d\uparrow})$

- Dot product with $\mathrm{D}_\downarrow$: $N_{d\downarrow}$ operations, produces a scalar
- Matrix product with $\nabla\mathrm{D}_\downarrow$: $3N_{\text{elec}\downarrow} \times N_{d\downarrow}$ operations, produces a $3N_{\text{elec}\downarrow}$ vector
- Matrix product with $\Delta\mathrm{D}_\downarrow$: $N_{\text{elec}\downarrow} \times N_{d\downarrow}$ operations, produces a $N_{\text{elec}\downarrow}$ vector
- Matrix product with $V^{\text{non-loc}}_{\text{pseudo}}\mathrm{D}_\downarrow$: $N_{\text{elec}\downarrow} \times N_{d\downarrow}$ operations, produces a $N_{\text{elec}\downarrow}$ vector

Use large CIPSI wave functions as trial wave functions for DMC[3]:

H_2O

- best estimate of the exact energy
- $\sim 10^6$ Slater determinants



[3] Caffarel et al, (2016), J. Chem. Phys., 144:15(151103)

Effect of the Jastrow factor on the CI wave function

- Adding a Jastrow factor on top of a CI wave function:
  - The N-electron basis is no more orthonormal

$$\langle D_I\, e^J | D_K\, e^J \rangle \neq \delta_{IK}$$

  - Double-counting of correlation
    - Dynamic correlation from the determinants
    - Dynamic correlation from the Jastrow
  - The CI coefficients are no more optimal

Effect of the Jastrow factor on the CI wave function

- Re-optimizing the CI coefficients in the presence of the Jastrow:
  - Increases large coefficients
  - Reduces small coefficients

- Solving H.C $= E$ S.C is difficult:
  - Statistical errors in matrix elements of H and S
  - Determinants with tiny CI coefficients have a negligible contribution to Ψ^2
  - The error on $\langle K|\hat{H}|L\rangle$ is often larger than the expectation value when c_K is small.

Excited states in QMC

- Nodal surfaces (DMC energies) are determined by the determinant expansion.
- Accurate energy differences need balanced wave function qualities between the states

Two different strategies:

1. Stochastic optimization
   - Use a deterministic method which gives a qualitatively good description (minimal CAS-SCF)
   - Reoptimize all the parameters: MOs, CI, Jastrow
2. Deterministic optimization
   - Use a deterministic method which gives a reasonable ΔE (MR-CI, CIPSI)
   - Run a DMC without modifying the wave function.

Excited states in QMC

	Pros	Cons
Deterministic optimization	Very good quality control Smooth potential energy surfaces	Very large expansions Limited to small systems
Stochastic optimization	Compact wave functions Can be applied to large systems	Noisy optimization Harder to get balanced energies

Good strategy towards large systems: The best of both worlds

- Small CIPSI expansions in a large active space : $\Longrightarrow$ compact
- Enforcing constant E_{PT2} for selecting determinants $\Longrightarrow$ $\Delta E \sim \Delta E_{\mathrm{FCI}}$ consistent quality for both states
- Optimize a Jastrow factor in QMC
- Re-optimize all parameters in QMC

Targeting Real chemical accuracy at the EXascale

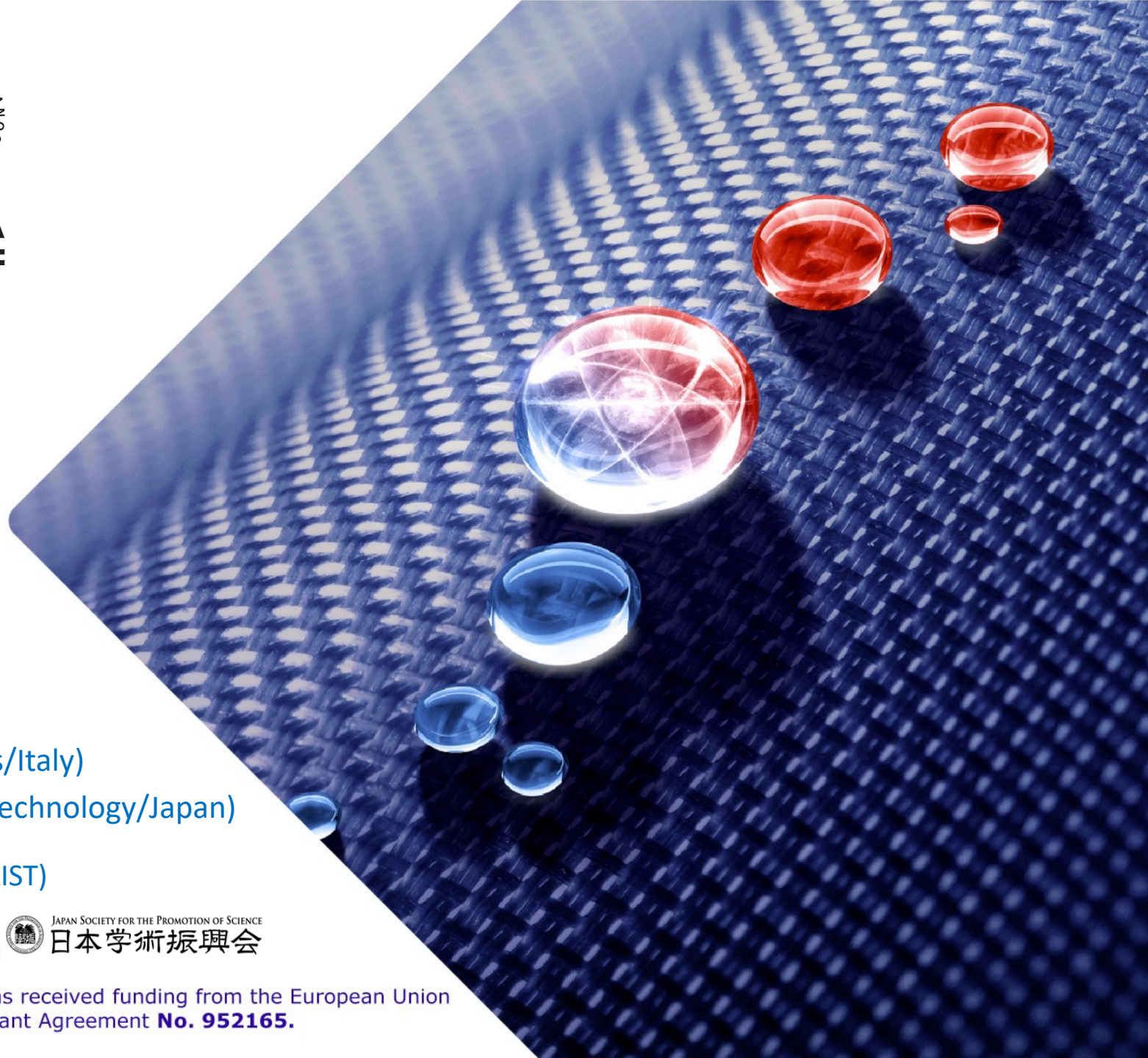
SISSA

TurboRVB and Turbo-Genius: Overview and Workflow

Kosuke Nakano

- SISSA (International School for Advanced Studies/Italy)
- JAIST (Japan Advanced Institute of Science and Technology/Japan)

(Prof. Sorella group/SISSA) (Prof. Maezono group/JAIST)



JAIST Japan Advanced Institute of Science and Technology 1990

Japan Society for the Promotion of Science 日本学術振興会

Targeting Real Chemical Accuracy at the Exascale project has received funding from the European Union Horizon 2020 research and innovation programme under Grant Agreement **No. 952165.**

Targeting Real chemical accuracy at the EXascale

SISSA

Day 3 and 4:
- Overview
- Hands-on session

Targeting Real Chemical Accuracy at the Exascale project has received funding from the European Union Horizon 2020 research and innovation programme under Grant Agreement **No. 952165.**

Targeting Real chemical accuracy at the EXascale

SISSA

Day 3 and 4:
- Overview
- Hands-on session

Targeting Real Chemical Accuracy at the Exascale project has received funding from the European Union Horizon 2020 research and innovation programme under Grant Agreement **No. 952165.**

Packages: TurboRVB and Turbo-Genius

TurboRVB
Quantum Monte Carlo Package SISSA

QMC engines (DFT, VMC-optimization, VMC, LRDMC)

K. Nakano, C. Attaccalite, M. Barborini, L. Capriotti, M. Casula, E. Coccia, M. Dagrada, Y. Luo, G. Mazzola, A. Zen, and S. Sorella, ***J. Chem. Phys.*** 152, 204121 (2020)

TurboGenius

User friendly python wrappers.

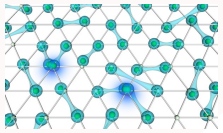
K. Nakano et al., ***in preparation*** (2022)

Wavefunction ansatz

$\Psi\left(\mathbf{r}_1, \mathbf{r}_2, \ldots \mathbf{r}_N\right)$ should be anti-symmetric under exchange of electron positions.

Slater determinant: the most straightforward ansatz $\Psi_{VMC} = Det\left|\phi_1, \phi_2, \cdots, \phi_N\right|$

- Linear combination of the slater determinants. $\Psi_{VMC} = Det\left|\phi_1, \phi_2, \cdots, \phi_N\right| +$
- Geminal functions (i.e., considering pairs of electrons.)
- Backflow functions (i.e., increase in the effective mass of electrons.)
- Fermi-net (i.e., anti-symmetric neural network.)



c.f. P.W. Anderson

More complex.

The more complex an ansatz is, the better energy we could get. However, the computational cost also increases.

One should increase the number of variational parameters, considering “physics”.

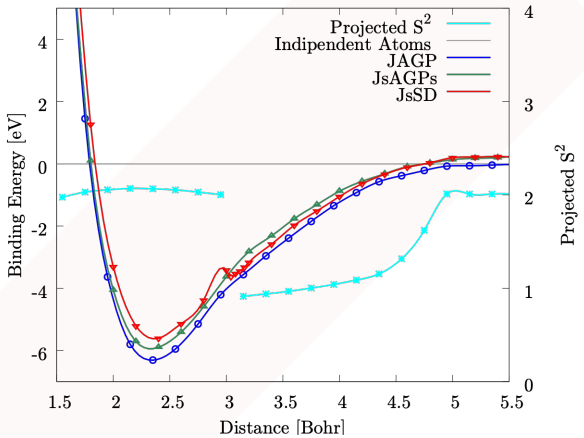
C_2 molecule

Double-bond?? Quadruple-bond??, spin-singlet.

S. Shaik, et al. Nat. Chem. 4 195–200 (2012)

DMC results

Wavefunction	C atom (Ha)	C_2 molecule (Ha)	Binding (eV)
Jastrow Slater	−37.82966(4)	−75.8672(1)	5.656(3)
Jastrow Geminal (Singlet)	−37.8364(1)	−75.8938(2)	6.01(1)
Jastrow Geminal (Singlet + broken sym.)	−37.8364(1)	−75.8935(2)	6.00(1)
Jastrow Geminal (All-pairing, Pfaffian)	− 37.8363(1)	−75.9045(2)	6.31(1)
Estimated exact	-37.8450	−75.9045(2)	6.44(2) (Exp.)



More complex.

C.G, T.S, **K.Nakano**, and S.S. J. Chem. Theory Comput. 16, 6114 (2020)

CCSD(T) with the V5Z basis = 6.24 eV

DMC gives a more accurate result than CCSD(T) for the challenging molecule!

TurboRVB paper

The Journal of Chemical Physics

TurboRVB: A many-body toolkit for *ab initio* electronic simulations by quantum Monte Carlo

Cite as: J. Chem. Phys. **152**, 204121 (2020); https://doi.org/10.1063/5.0005037
Submitted: 19 February 2020 . Accepted: 20 March 2020 . Published Online: 29 May 2020

Kousuke Nakano, Claudio Attaccalite, Matteo Barborini, Luca Capriotti, Michele Casula, Emanuele Coccia, Mario Dagrada, Claudio Genovese, Ye Luo, Guglielmo Mazzola, Andrea Zen, and Sandro Sorella

COLLECTIONS

Paper published as part of the special topic on Collection
Note: This article is part of the JCP Special Topic on Electronic Structure Software.

View Online Export Citation CrossMark

» Home View page source

Home

TurboRVB
Quantum Monte Carlo Package SISSA

News

- K. Nakano et al. have published a paper in Phys. Rev. B 103, L121110 (2021).
  This paper has been selected as an **Editors' Suggestion**.
- Our TurboRVB workshop will be held on **12-16 July 2021** at SISSA (Italy)!
  Please have a look at Summer School on Quantum Monte Carlo methods 2021.
  Online registration can be done from the TREX website.
- C. Genovese and S. Sorella have published a paper in J. Chem. Phys. 153, 164301 (2020).
- C. Genovese et al. have published a paper in J. Chem. Theory Comput. 16 6114-6131 (2020).
- We have published a TurboRVB review paper in J. Chem. Phys. 152, 204121 (2020).

K. Nakano, C. Attaccalite, M. Barborini, L. Capriotti, M. Casula, E. Coccia, M. Dagrada, Y. Luo, G. Mazzola, A. Zen, and S. Sorella, ***J. Chem. Phys.*** 152, 204121 (2020)

What we can do with TurboRVB?

Please visit our website :-) All the papers and Ph.D thesis using TurboRVB are listed here.

TurboRVB website

Updated on 03/06/2021

Search docs

CONTENTS:

- News
- Developers
- Source code
- Workshops
- Positions
- Publications
  - 2021
  - 2020
  - 2019
  - 2018
  - 2017
  - 2016
  - 2015
  - 2014
  - 2013
  - 2012
  - 2011
  - 2010
  - 2009

» Publications View page source

Publications

TurboRVB

Quantum Monte Carlo Package / SISSA

2021

- Atomic forces by quantum Monte Carlo: Application to phonon dispersion calculations, K. Nakano, T. Morresi, M. Casula, R. Maezono, and S. Sorella, Phys. Rev. B 103, L121110 (2021). Selected as an **Editors' Suggestion**

2020

- Ground-state properties of the hydrogen chain: insulator-to-metal transition, dimerization, and magnetic phases, M. Motta, C. Genovese, F. Ma, Z. Cui, R. Sawaya, G.K. Chan, N. Chepiga, P. Helms, C. Jiménez-Hoyos, A.J. Millis, U. Ray, E. Ronca, H. Shi, S. Sorella, E.M. Stoudenmire, S.R. White, and S. Zhang (Simons Collaboration on the Many-Electron Problem), Phys. Rev. X 10, 031058 (2020).

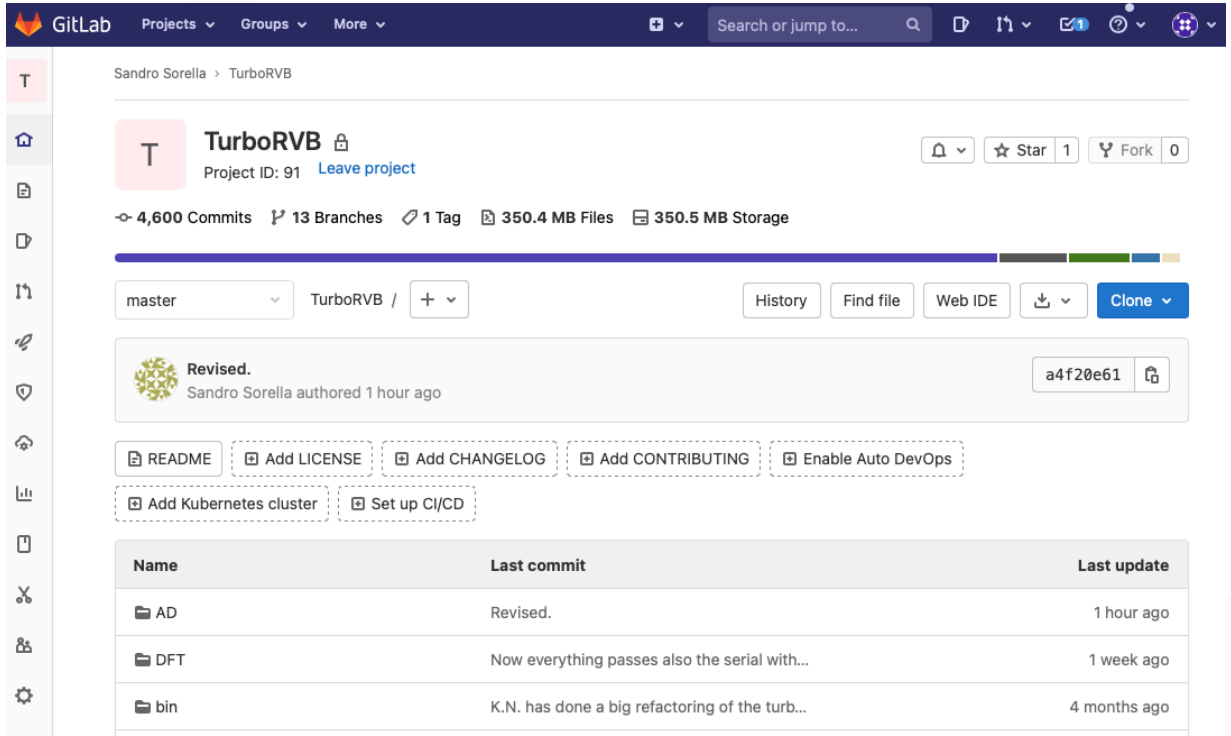
» PhD thesis View page source

PhD thesis

- **Dr. Claudio Genovese**, 2020: Geminal Power in QMC, pdf
- **Dr. Félix Mouhat**, 2018: Fully quantum dynamics of protonated water clusters, pdf
- **Dr. Nicolas Dupuy**, 2016: Corrélations électroniques des acènes vers la limite de longue taille : Étude par Monte Carlo quantique (Electronic correlations in the acenes to ards the long-size limit: a Monte Carlo study), pdf
- **Dr. Henri Hay**, 2016: Étude de la structure et des propriétés des polymorphes de SiO_2 et B_2O_3 par méthodes ab initio (Structural properties of SiO_2 and B_2O_3 polymorphs by ab initio methods), pdf
- **Dr. Mario Dagrada**, 2016: Improved quantum Monte Carlo simulations: from open to extended systems, pdf
- **Dr. Nicolas Dévaux**, 2015: Étude par Monte Carlo quantique de la transition α-γ du Cérium (Quantum Monte Carlo study of the α-γ transition in Cerium), pdf
- **Dr. Guglielmo Mazzola**, 2014: Metallization and dissociation in high pressure liquid hydrogen by an efficient molecular dynamics with quantum Monte Carlo, pdf
- **Dr. Ye Luo**, 2014: Ab initio molecular dynamics of water by quantum Monte Carlo, pdf

SISSA Gitlab server

Where can we download TurboRVB and Turbo-Genius?



From SISSA-gitlab server.

https://git-scm.sissa.it

A request: to kousuke_1123@icloud.com

For the time being, turborvb and turbo-genius are inhouse codes, so please DO NOT distribute to the public.

Within a year, all the codes will be public under an appropriate license (maybe BSD) :-)

Targeting Real chemical accuracy at the EXascale

SISSA

Day 3 and 4:

- Overview
- Hands-on session

Targeting Real Chemical Accuracy at the Exascale project has received funding from the European Union Horizon 2020 research and innovation programme under Grant Agreement **No. 952165.**

TurboRVB/Turbo-Genius manual and tutorials

TurboRVB userguides
Updated on 05/07/2021
Search docs

User Manuals
Tutorials
Developer Manuals

» TurboRVB user manuals View page source

TurboRVB user manuals

- User Manuals
  - TurboRVB Manual
    - Installation
    - TurboRVB in a nutshell
    - Wavefunction
    - makefort10
    - DFT driver
    - QMC driver
    - Tools and input examples
    - Workflow
  - Turbo-Genius Manual
    - Installation
    - Turbo-Genius in a nutshell
- Tutorials
  - TurboRVB tutorials
    - 00 TurboRVB install
    - 01_01Hydrogen_dimer
    - 01_02Hydrogen_dimer
    - 02_01Lithium_dimer
    - 98 Wavefuntion optimization

From SISSA-gitlab server.

https://git-scm.sissa.it

Any contributor is welcome!!!

They are composed by sphinx. All the tutorial in this school is included here.

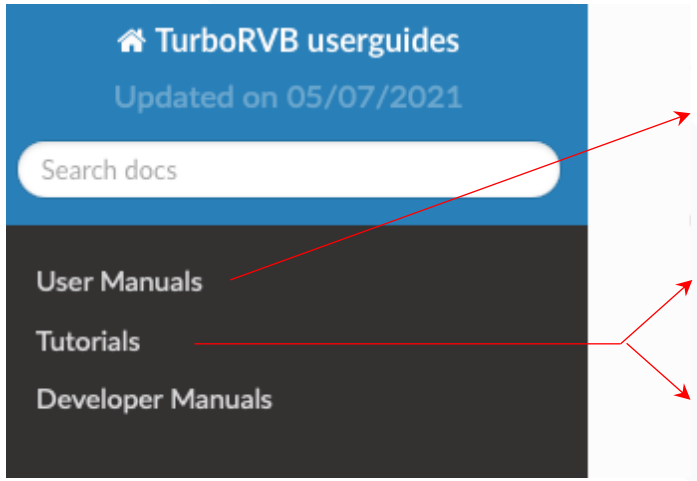
TurboRVB/Turbo-Genius manual and tutorials

If you want to see the userguide, please let us know. We will give you the permission.

git clone git@git-scm.sissa.it:sorella/turborvb_userguides.git

Any contributor is welcome!!!

Open /turborvb_userguides -> build -> html -> index.html



- User Manuals.
- TurboRVB tutorials.
- Turbo-Genius tutorials (for the hands-on session).

on your local comp. (for the TREX summer school).

Compilation (TurboRVB)

We strongly recommend Intel, IBM, and Fujitsu Fortran compilers. (Not gfortran).

1. Legacy make:

<u>You do not have to do this for the hands-on session!!!</u>

Copy a config file: config/make_XXX.inc make.inc

Copy a make.txt file: src/make.txt_standard src/make.txt

Compile TurboRVB: ./makeall (serial) or ./makeall-mpi

If you want to clean it: make cleanall

2. Modern CMake:

```
mkdir build
cd build
cmake  -DXXXX = YYYY etc...
make install    # copy generated binaries to ./bin directory.
```

Fugaku, Hokusai (RIKEN)

Marconi, Marconi100 (CINECA)

SISSA-cluster (SISSA)

Kagayaki (JAIST)

A typical workflow in QMC

(Input) Atomic positions, basis sets, pseudo potentials…

1) Pre-process:

WFT (Gaussian16, PySCF)
or **DFT** (Quanaum Espresso)

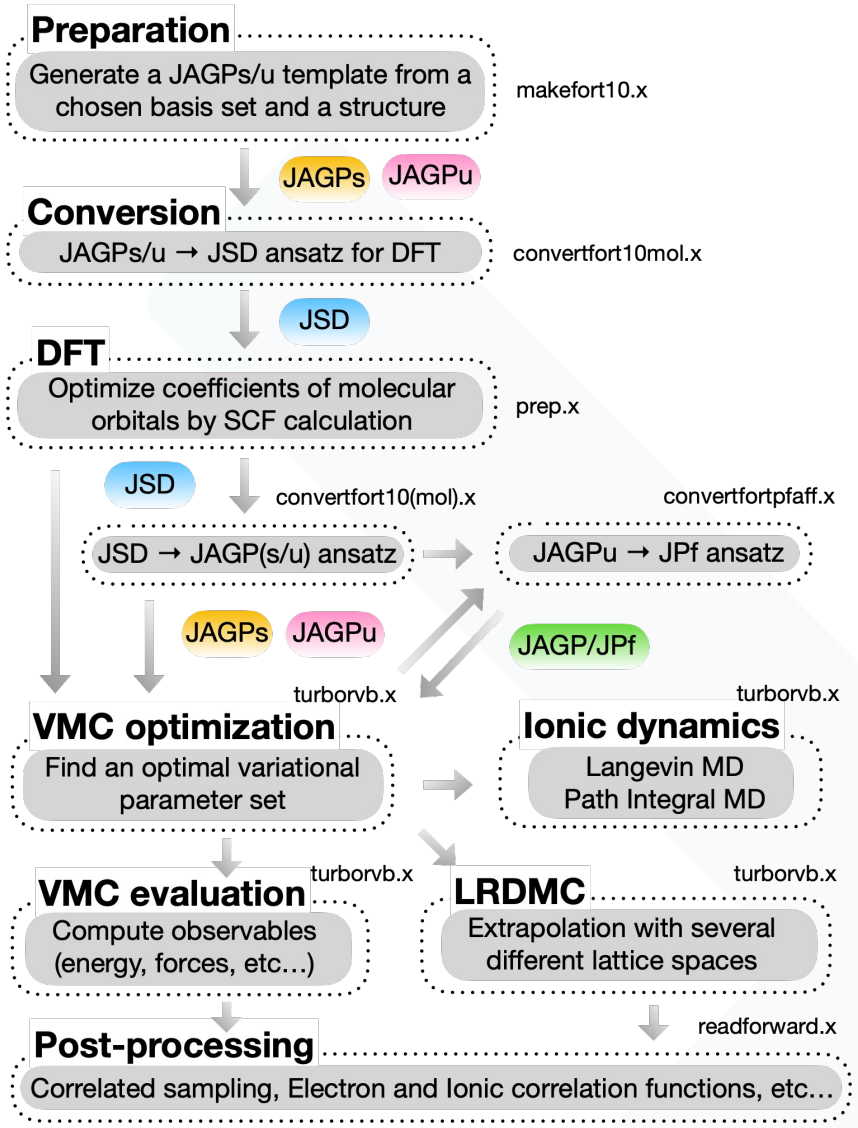
The obtained one-particle WFs

2) Post-process:

QMC (TurboRVB, etc…)

(Output) Many-body WFs, its energy, forces, etc…

A typical workflow in TurboRVB (TurboGenius)



= Workflow =

1. Prepare a structure and basis set **makefort10**

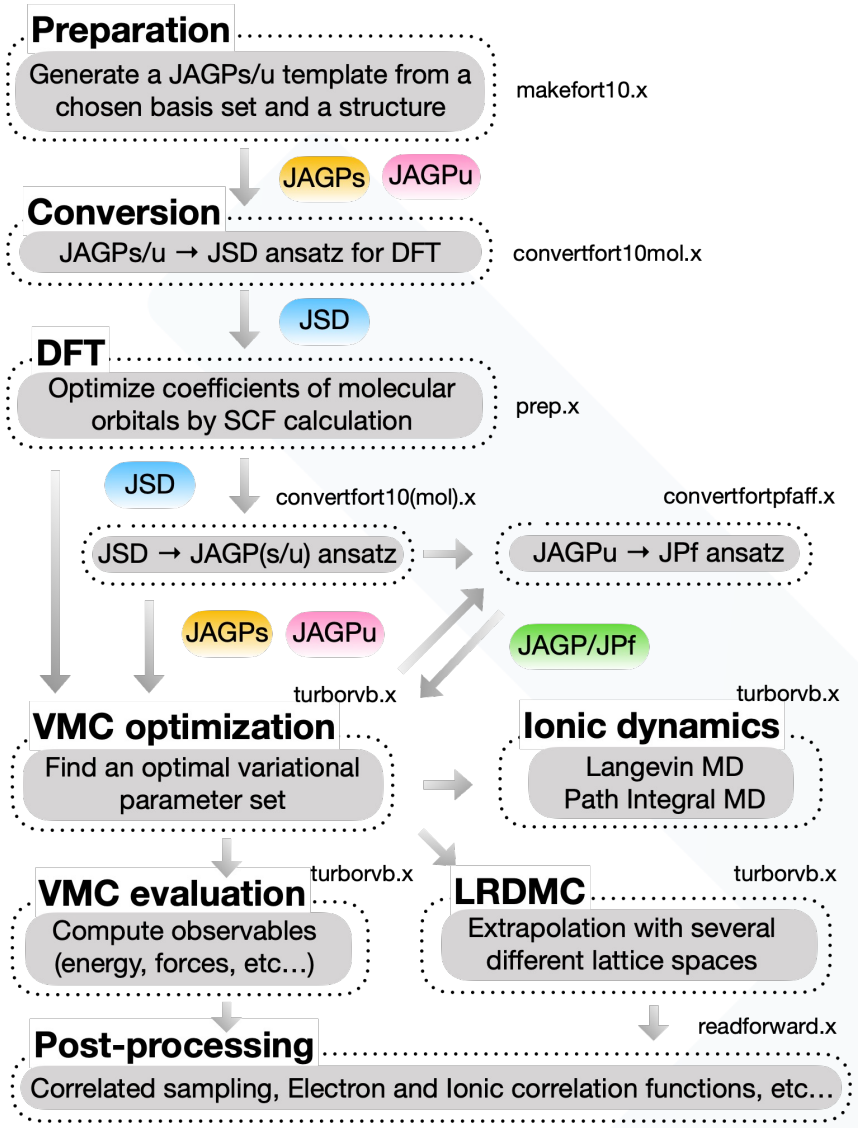
2. DFT Construct a reasonable initial WF! **prep**

3. VMC-opt Optimize the wavefunction **turborvb**

4. VMC Do a VMC run. **turborvb**

5. LRDMC LRDMC with the optimized WF. **turborvb**

A typical workflow in TurboRVB (TurboGenius)



= Workflow =

1. Prepare a structure and basis set **makefort10**

2. DFT Construct a reasonable initial WF! **prep**

3. VMC-opt Optimize the wavefunction **turborvb**

4. VMC Do a VMC run. **turborvb**

5. LRDMC LRDMC with the optimized WF. **turborvb**

Wavefunction (makefort.10.x)

Input: makefort10.input → Binary: makefort10.x → Output:fort10_new

makefort10.x is a tool for generating JAGP WF(fort.10) from makefort10.input.

```
# Ion coordinates
 N1 Z1                x1     y1     z1
 N2 Z2                x2     y2     z2
   ..                ..     ..     ..
 Nn Zn                xn     yn     zn
```

Structural information.

```
#       Parameters atomic wf
        1          4          300
        1    2.0   1.0  3.231  7.54
```

Basis-set information.

```
posunits='crystal'
natoms=2
ntyp=1
complexfort10=.false.
pbcfort10=.true.
!yes_pfaff=.true.
celldm(1)=4.648726266579395
celldm(2)=1.0
celldm(3)=4.065040650406504
celldm(4)=1.5707963267948966
celldm(5)=1.5707963267948966
celldm(6)=2.0943951023931953
yes_tilted=.true.
nxyz(1)=3
nxyz(2)=3
nxyz(3)=1
phase(1)=0.0
phase(2)=0.0
phase(3)=0.0
phasedo(1)=0.0
phasedo(2)=0.0
phasedo(3)=0.0
```

makefort10.input file

makefort10.x

```
# fort.10 of the C2-dimer (the Pfaffian ansatz with the Filippi pseudo potential.)
# Nelup  #Nel  # Ion
        4          -8          2
# Shell Det.   # Shell Jas.
        50         43
# Jas 2body  # Det   #  3 body atomic par.
        -22        1482        42
# Det mat. =/0  # Jas mat. =/0
        120        8370
# Eq. Det atomic par.  # Eq. 3 body atomic. par.
        741        21
# unconstrained iesfree,iessw,ieskinr,I/O flag
        8370       120         6           0
# Ion coordinates
4.00000000000000        6.00000000000000        0.000000000000000E+000
0.000000000000000E+000  -1.14999954166875
4.00000000000000        6.00000000000000        0.000000000000000E+000
0.000000000000000E+000   1.14999954166875
#  Constraints for forces: ion - coordinate
        1          1          1
        1          1          2
        1          1          3
        1          2          1
        1          2          2
        1          2          3
#          Parameters Jastrow two body
        -1  0.342214663461764
...
```

Wavefunction file (fort.10)

Wavefunction file (fort.10)

All the information (except for pseudo potential info.) is included in a single file, named “fort.10”

```
# fort.10 of the C2-dimer (the Pfaffian ansatz with the Filippi pseudo potential.)
# Nelup  #Nel  # Ion
        4          -8           2
# Shell Det.   # Shell Jas.
        50          43
# Jas 2body  # Det   #  3 body atomic par.
        -22        1482          42
# Det mat. =/0  # Jas mat. =/0
        120        8370
# Eq. Det atomic par.  # Eq. 3 body atomic. par.
        741          21
# unconstrained iesfree,iessw,ieskinr,I/O flag
        8370         120           6           0
# Ion coordinates
4.00000000000000        6.00000000000000       0.00000000000000E+000
0.00000000000000E+000  -1.14999954166875
4.00000000000000        6.00000000000000       0.00000000000000E+000
0.00000000000000E+000   1.14999954166875
#  Constraints for forces: ion - coordinate
        1           1           1
        1           1           2
        1           1           3
        1           2           1
        1           2           2
        1           2           3
#          Parameters Jastrow two body
        -1  0.342214663461764
...
```

“fort.10” can be generated by “makefort10.x” (see later).

fort.10 Header

Header:

```
# Nelup  #Nel  # Ion
          2          4          1
# Shell Det.   # Shell Jas.
          3          3
# Jas 2body  # Det   #  3 body atomic par.
         -8         16          8
# Det mat. =/0  # Jas mat. =/0
          6          6
# Eq. Det atomic par.  # Eq. 3 body atomic. par.
         13          8
# unconstrained iesfree,iessw,ieskinr,I/O flag
          4          4          0          0
```

Nelup: The number of spin up electrons in the system.

Nel: The total number of electrons in the system.

Ion: The number of nuclei in the system.

Jas 2body: Onebody and Twobody Jastrow types

The number of atomic forces.

The total number of determinant variational param.

The total number of Jastrow variational param.

Coordinates:

```
# Ion coordinates
 N1 Z1                x1      y1      z1
 N2 Z2                x2      y2      z2
   ..                ..      ..      ..
 Nn Zn                xn      yn      zn
```

- N: Atomic number

- Z: The number of valence electrons

- xn, yn, zn : atomic positions (Bohr)

Pseudo potential case N != Z

If you want to use a H-pseudo potential, please put N=1.0, Z=1.00001 (dummy).

Basis set in makefort10.input/fort.10

Basis set for the determinant part:

```
#              Parameters atomic wf
Shell_Multiplicity      Number of par.
Ion index                [par (1, NUMBER OF PAR.)]

#              Parameters atomic wf
               1           1          16
               1  0.500000000000000
               3           1          36
               1   1.00000000000000
               1           1          16
               2  0.300000000000000
               1           1          16
               3  0.300000000000000
               1           1          16
               4  0.300000000000000
               1           1          16
               5  0.300000000000000
```

$$\phi(r) \sim \exp(-Zr^2)$$

```
#       Parameters atomic wf
        1          4          300
        1    2.0   1.0  3.231  7.54
```

$$\phi(r) = 3.231 \cdot \exp(-2.0 \cdot r^2) + 7.54 \cdot \exp(-1.0 \cdot r^2)$$

Shell codes:
- 16 -> s orbital
- 36 -> p orbital
- 68 -> d orbital
- 48 -> f orbital
- 51 -> g orbital
- 72 -> h orbital
- 73 -> i orbital

Single Slater-determinant (convertfort10mol.x)

Input: convertfort10mol.input, fort.10_in → Binary: convertfort10mol.x → Output:fort10_new

convertfort10mol.x is a tool for adding molecular orbitals to fort.10_in.

This is used for converting a JAGP WF to a JSD WF. JAGPs → JSD

JSD Slater Determinant

$$f(\mathbf{r}_i, \mathbf{r}_j) = \sum_{k=1}^{M=N_{\mathrm{el}}/2} \lambda_k \tilde{\Phi}_k(\mathbf{r}_i) \tilde{\Phi}_k(\mathbf{r}_j)$$

with $\tilde{\Phi}_k = \sum_{i=1}^{N_{\mathrm{basis}}} c_{i,k} \cdot \phi_i(\boldsymbol{r})$

JAGPs

$$f(\mathbf{r}_i, \mathbf{r}_j) = \sum_{l,m,a,b} A_{\{a,l\},\{b,m\}} \psi_{a,l}(\mathbf{r}_i) \psi_{b,m}(\mathbf{r}_j)$$

DFT (prep.x) works only with molecular orbitals!! So, one should convert a WF from the JsAGPs to JSD.

Coefficients of the Determinant part (JAGPs case)

```
#          Nonzero values of detmat
           1       5       9.421753101774391E-002
           1       6       9.421753101774391E-002
           1       7       9.421753101774391E-002
```

$$A = \begin{pmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ & A_{22} & \dots & A_{2n} \\ & & \ddots & \vdots \\ & & & A_{nn} \end{pmatrix}$$

$$f\left(\mathbf{r}_i, \mathbf{r}_j\right) = \sum_{l,m,a,b} A_{\{a,l\},\{b,m\}} \psi_{a,l}\left(\mathbf{r}_i\right) \psi_{b,m}\left(\mathbf{r}_j\right)$$

$$g_{ud}(\mathbf{i},\mathbf{j}) \equiv g_s(\mathbf{i},\mathbf{j}) = f_S(\mathbf{r}_i,\mathbf{r}_j)\frac{|\uparrow\downarrow\rangle - |\downarrow\uparrow\rangle}{\sqrt{2}},$$

→ JAGPs

$A_{\{a,l\},\{b,m\}}$ is a symmetric matrix!

fort.10: details

Molecular orbitals (100000): In fort.10, 1000000 indicates a molecular orbital.

```
#always 1,   the number of components, 100000
#index of basis [1,2,....]
#coefficients for basis [1,2,....]
 1         180     1000000
 1           1           2           3           4           5
 6           7           8           9          10          11
 12          13          14          15          16          17
 18          19          20          21          22          23
 24          25          26          27          28          29
 30          31          32          33          34          35
 36          37          38          39          40          41
 42          43          44          45          46          47
 48          49          50          51          52          53
 54          55          56          57          58          59
 60          61          62          63          64          65
 66          67          68          69          70          71
 72          73          74          75          76          77
 78          79          80          81          82          83
 84          85          86          87          88          89
 90  0.438271164894104      -4.608166217803955E-002
0.189550578594208       7.299757003784180E-002 -0.129178702831268
-0.241831779479980      -7.793867588043213E-002 -0.143670558929443
-0.181271851062775      -0.265352427959442       0.374841809272766
```

$$\Phi_k = \sum_{i=1}^{N_{\text{basis}}} c_{i,k} \cdot \phi_i(\boldsymbol{r})$$

JSD

Molecular orbitals can be added by “convertfort10mol.x”. DFT works only with molecular orbitals.

Onebody and Twobody Jastrows

twobody: 1B and 2B Jastrows: Various Jastrow types are implemented (see the manual.)

Typically:

-6: Open/PBC with pseudo potentials

Only two-body parameter. $1\mathrm{b} = \frac{1}{2a}(1 - e^{-ar})$

i.e., electron-ion cusp conditions are enough.

-15: Open/PBC with all-electrons

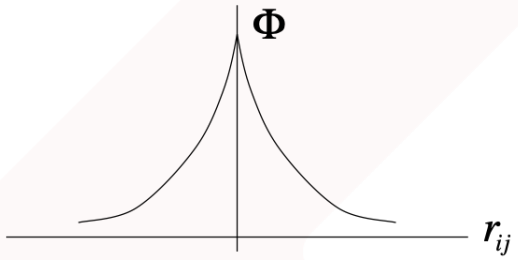
two-body and one-body parameters.

$$2\mathrm{b} = \frac{r}{2(1+br)} \quad 1\mathrm{b} = \frac{1}{2b}(1 - e^{-br})$$

-22: Open/PBC with JAGPu/JPf.

Only one-body or two-body and one-body parameters.

Spin-dependent Jastrow factors:



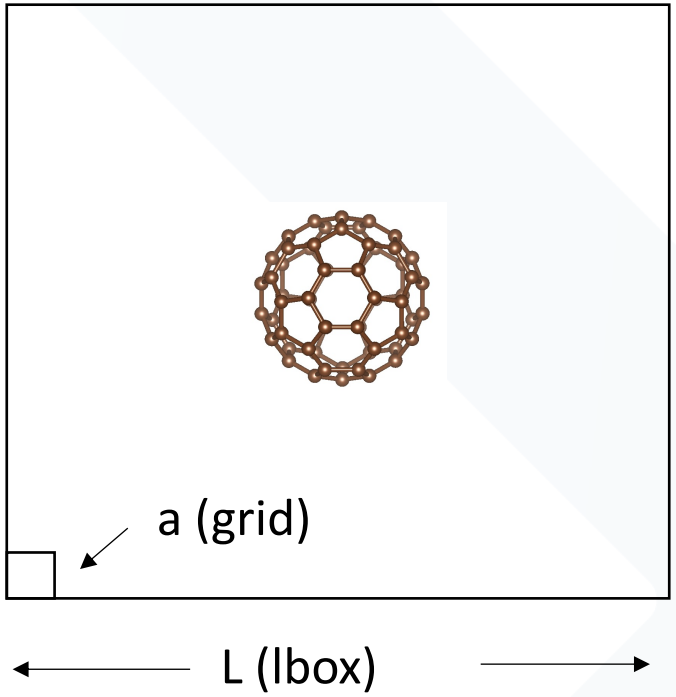
Electron-ion (1b).

Electron-Electron (2b).

To satisfy the cusp conditions.

Input: prep.input, fort.10 → Binary: prep-(serial, mpi).x → Output:fort10_new

Box and mesh sizes are so important for obtaining converged results in practice !!



$$H = \hat{T} + V_{\text{ele-ion}}(\vec{r}) + V_{\text{ele-ele}}(\vec{r}) + V_{\text{XC}}(\vec{r})$$

For a calculation with PPs, a~0.10 bohr is small enough.

For an all-electron calculation, a < 0.05 bohr is needed.
The double-grid algorithm should also be helpful.

If you have enough memories, we recommend
L ~ 20 Bohr for the safety.

Lz = cell length for a periodic system. Automatically set.

Built-in DFT code (prep.x)

Input: prep.input, fort.10 → Binary: prep-(serial, mpi).x → Output:fort10_new

prep.x is a built-in DFT code!!

Why built-in?

$$\tilde{\phi}_j^b(\mathbf{r}-\mathbf{R}_b) = \phi_j^b(\mathbf{r}-\mathbf{R}_b)\,\tilde{J}_1(\mathbf{r})$$

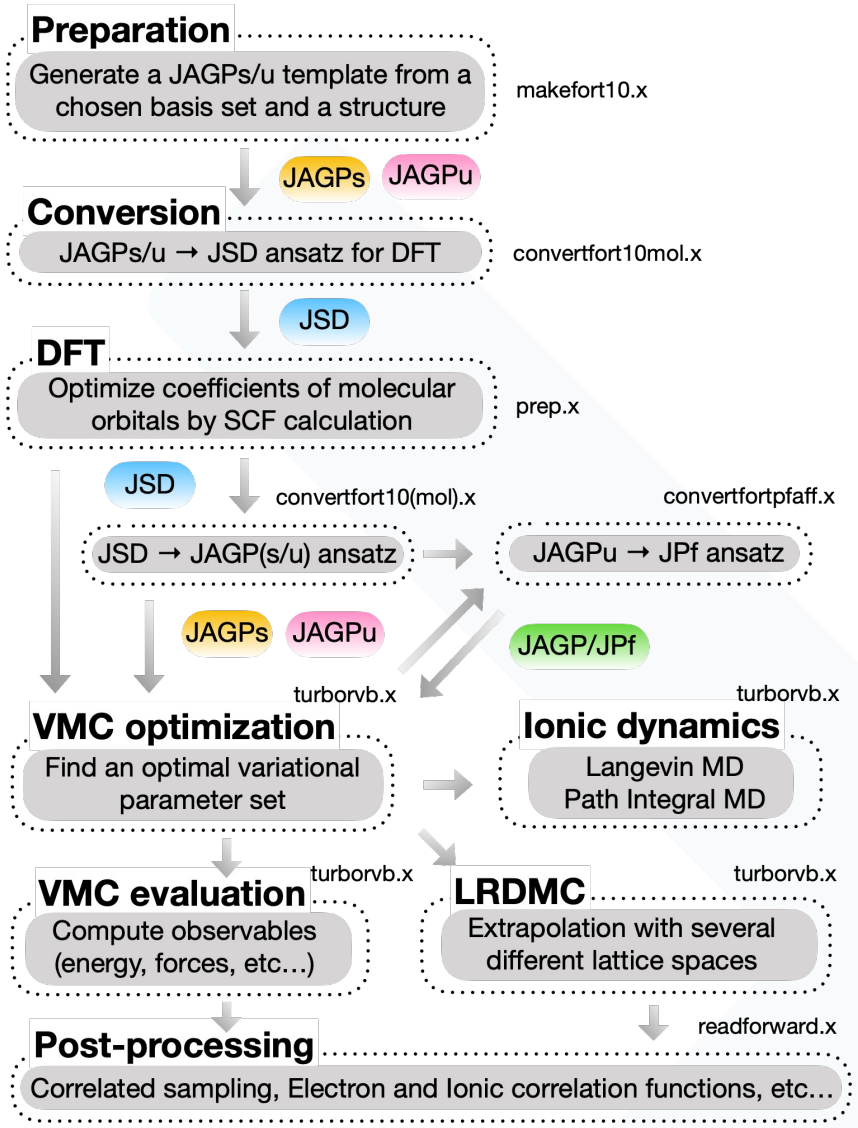
As mentioned before, the modified gaussian orbital is used.

So, we cannot exploit the analytical integration even though we employ the Gaussian primitive orbitals.

The CRYSTAL basis + cusp. for PBC cases.

We are also implementing converters for several QC codes (e.g., Gaussian) via TREX-IO.

A typical workflow in TurboRVB (TurboGenius)



= Workflow =

1. Prepare a structure and basis set **makefort10**

2. DFT Construct a reasonable initial WF! **prep**

3. VMC-opt Optimize the wavefunction **turborvb**

4. VMC Do a VMC run. **turborvb**

5. LRDMC LRDMC with the optimized WF. **turborvb**

Main QMC engine: turborvb-(serial, mpi).x

Input: datas(min, vmc, fn).input, fort.10 → Binary: turborvb-(serial, mpi).x → Output:fort10

turborvb.x is the main QMC engine in the turborvb package.

=VMC-opt, VMC, DMC, and LRDMC=

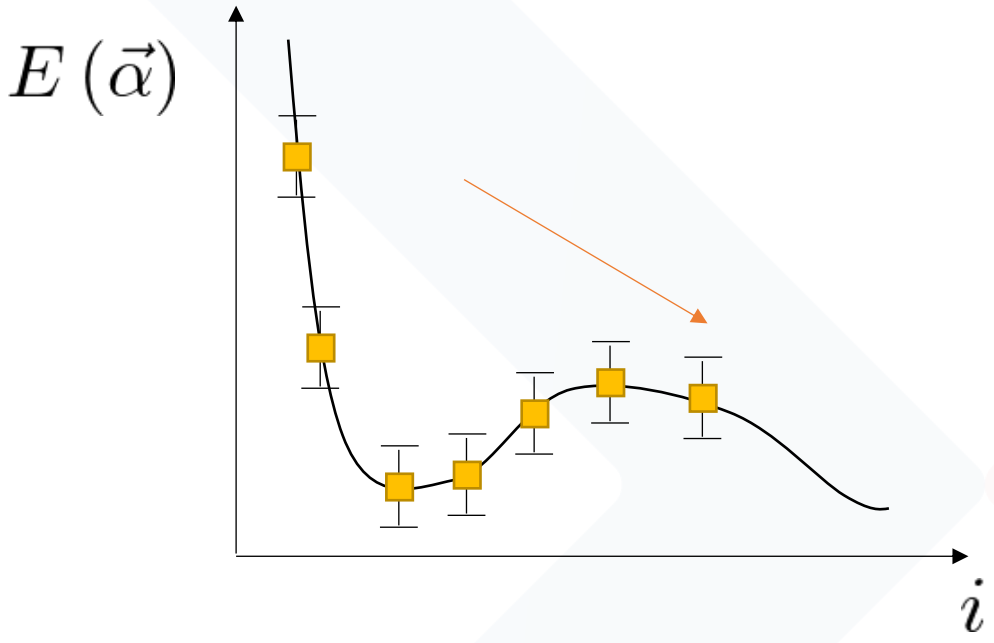
- Single-shot VMC run (itestr4=-2 in the &simulation namelist).
- VMC optimization (itestr4=-4,-5,-8,-9 in the &simulation namelist).
- Single-shot LRDMC run (itestr4=-6 in the &simulation namelist).
- Single-shot DMC run (itestr4=-5 in the &simulation namelist), but not maintained.

What the VMC optimization does?

$$E\left(\vec{\alpha}\right) = \frac{\int d\vec{R} \cdot \Psi^* \left(\vec{R}, \vec{\alpha}\right) \cdot \hat{\mathcal{H}} \Psi \left(\vec{R}, \vec{\alpha}\right)}{\int d\vec{R} \cdot \Psi^* \left(\vec{R}, \vec{\alpha}\right) \Psi \left(\vec{R}, \vec{\alpha}\right)} \geq E_0$$

The variational principle

This integral is evaluated using the MCMC method.



Variational parameters!

$$\vec{\alpha}_{i+1} \leftarrow \vec{\alpha}_i + \Delta\vec{\alpha}$$

e.g.,

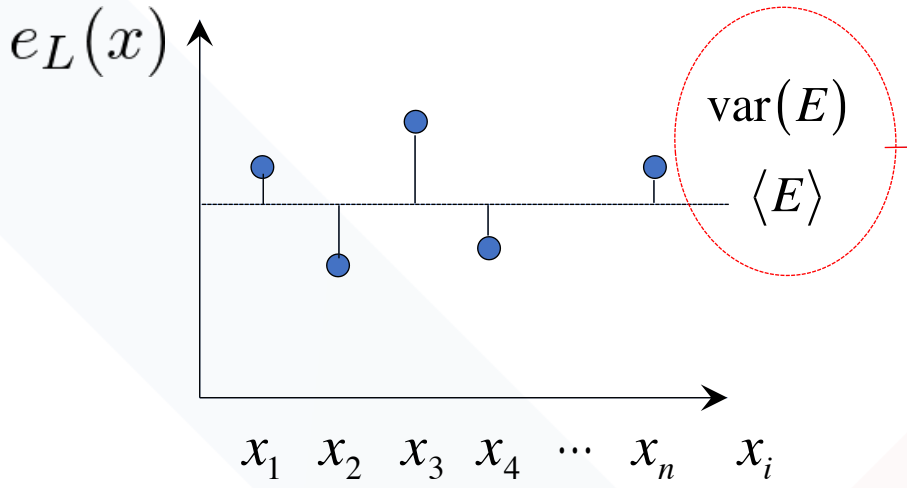
$$f_S\left(\mathbf{r}_i, \mathbf{r}_j\right) = \sum_{l,m,a,b} A_{\{a,l\},\{b,m\}} \psi_{a,l}\left(\mathbf{r}_i\right) \psi_{b,m}\left(\mathbf{r}_j\right).$$

The number of MCMC steps

```
&simulation
    itestr4=-4
    ngen=10000
    iopt=1
    maxtime=10800
/

&pseudo
/

&vmc
    epscut=0.0
/

&optimization
    nweight=100
    nbinr=5
    iboot=0
    tpar=0.35
    parr=-0.001
/

&readio
    !iread=3
/

&parameters
    !iessw=1
    !iesup=1
    !iesm=1
    !ieskin=1
    iesd=1
    iesfree=1
/
```

Relation between ngen and nweight

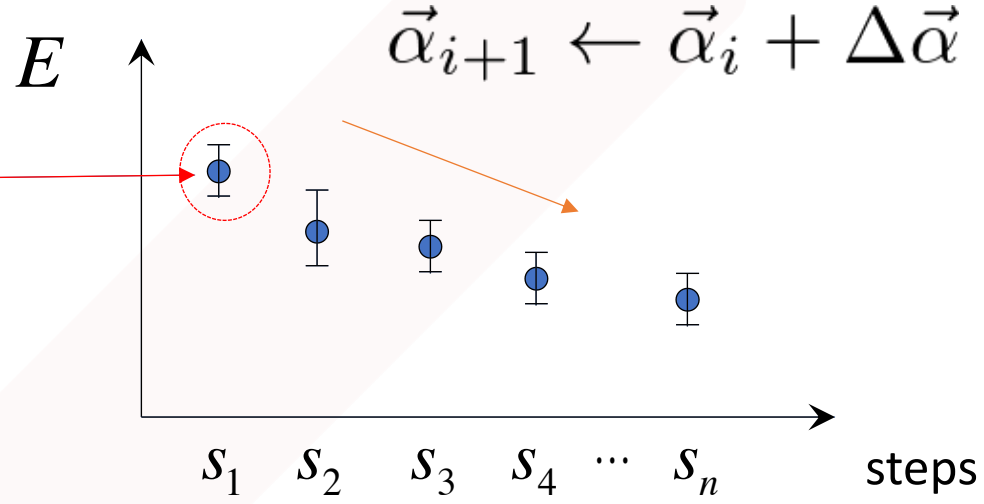
Each iteration



$$\| $$

nweight $\times$ nw

Optimization



Total optimization steps: $\dfrac{\mathrm{ngen}}{\mathrm{nweight}}$

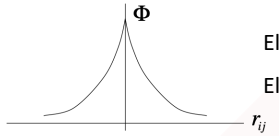
(default: the num. walker = the num. of MPI process)

The most difficult part in practice

3. VMC Optimize wavefunctions and VMC run.

A wavefunction reads $\Psi_{AS}(\vec{R}) \times \exp(J(\vec{R}))$ ⟶

Anti-symmetric part. Jastrow factor.



Electron-ion.

Electron-Electron.

To satisfy the cusp.

Jastrow factor = $\exp(J(\vec{R}))$ No effect on the nodal surface!! $\Psi(\vec{R}) \equiv \Psi(\vec{r_1}, \vec{r_2}, \cdots, \vec{r_N}) = 0$

Anti-symmetric part = $\Psi_{AS}(\vec{R})$ Determines the nodal surface. Its initial guess is taken from a DFT calculation!!

1st Step JSD 2nd Step JAGPs

Implemented optimization methods

Implemented optimization algorithms

-9, -5): Stochastic reconfiguration (natural gradient method)

S Sorella, et al., ***J. Chem. Phys.*** 127, 014105 (2007).

-4, -8): Linear method with the natural gradient

C.J. Umrigar, et al., ***Phys. Rev. Lett.*** 98, 110201 (2007).

In both cases, the most important parameters in practice are

1. tpar: Acceleration parameter (learning rate.)

e.g., $\alpha_k \to \alpha_k + \Delta \cdot \left(\boldsymbol{S'}^{-1} \mathbf{f} \right)_k$

tpar = 3.5d-1, and 1.0d-3 for -4 and -9, respectively.

2. parr: Regularization (c.f. LASSO)

e.g., $s'_{i,i} = s_{i.i}(1+\varepsilon)$

Depending on the accuracy your need. parr = ~ 1.0d-3

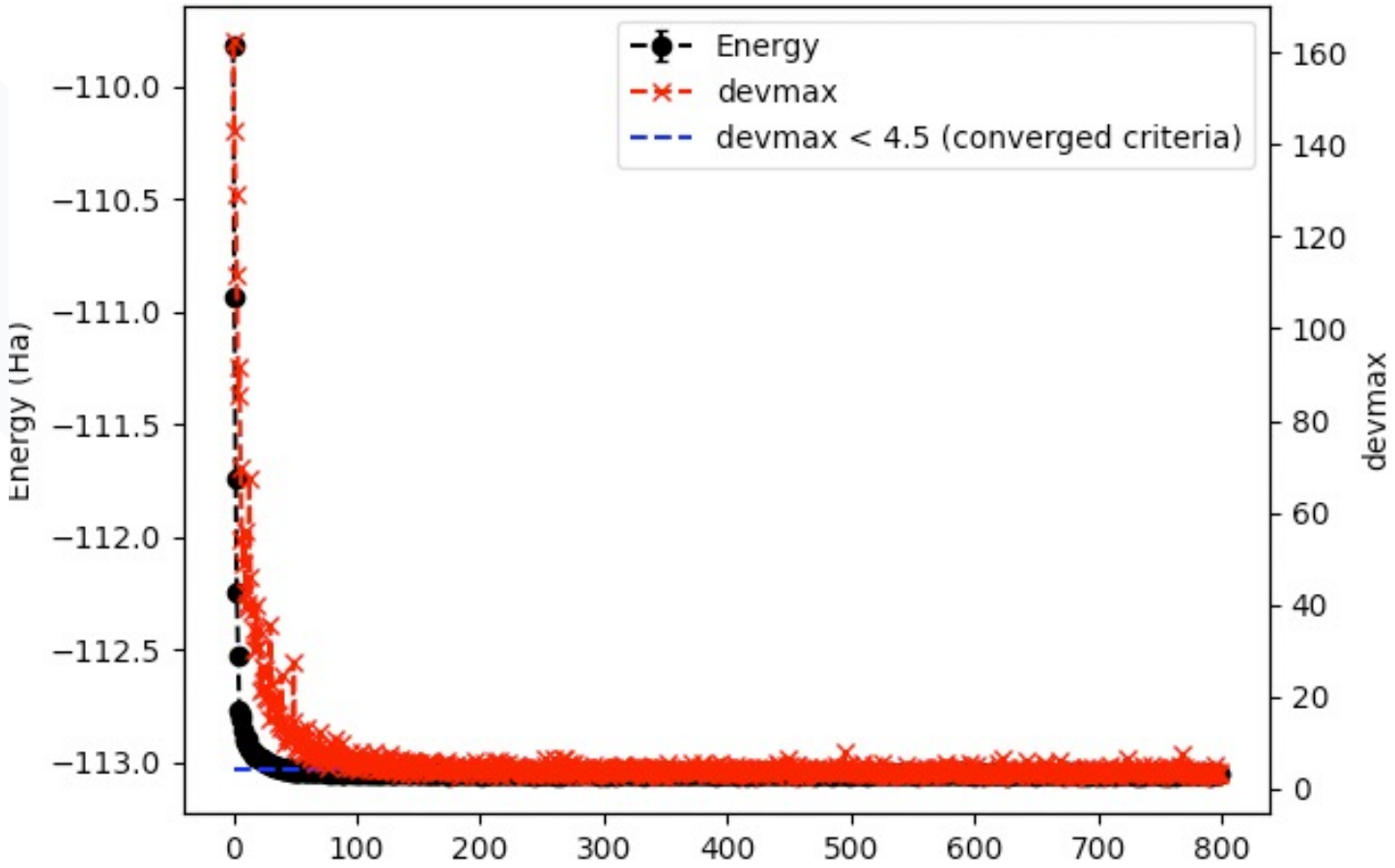
Optimization criteria

At least, ``devmax`` should be smaller than 4.0 after optimization. However, we also have experienced that this simple criteria is sometimes not sufficient to obtain a converged result.

The definition of ``devmax`` is: $devmax \equiv \max_k \left(\left| \frac{f_k}{\sigma_{f_k}} \right| \right)$

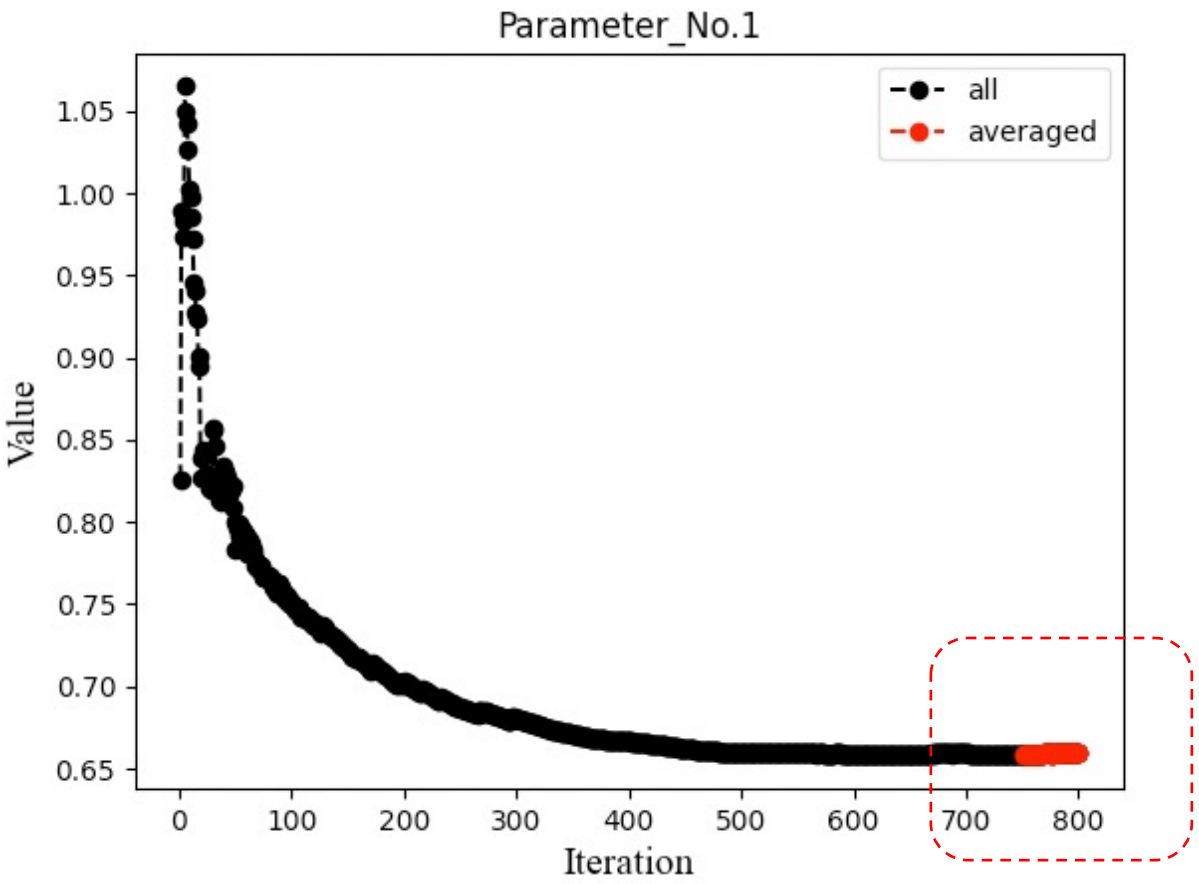
where σ_{f_k} represents the estimated error bar of a general force $f_k = -\frac{\partial E(\alpha)}{\partial \alpha_k} = -\frac{\partial}{\partial \alpha_k} \frac{\langle \Psi_\alpha | \hat{\mathcal{H}} | \Psi_\alpha \rangle}{\langle \Psi_\alpha | \Psi_\alpha \rangle}.$

Plotting Graphs (Turbo-Genius)



%turbo-genius.sh -j vmcopt -post -am interactive_detail

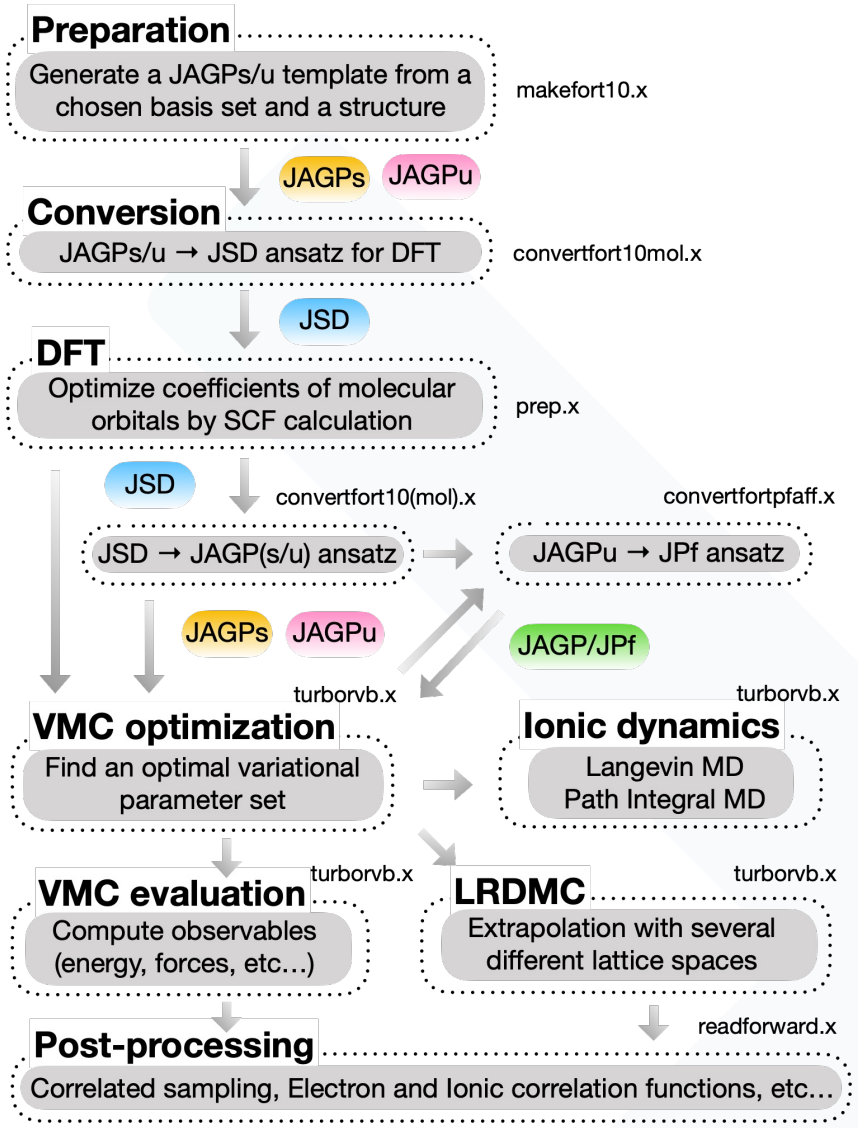
Averaging parameters (Turbo-Genius)



Averaged!

turbo-genius.sh -j vmcopt -post -am interactive_detail

A typical workflow in TurboRVB (TurboGenius)



= Workflow =

1. Prepare a structure and basis set **makefort10**

2. DFT Construct a reasonable initial WF! **prep**

3. VMC-opt Optimize the wavefunction **turborvb**

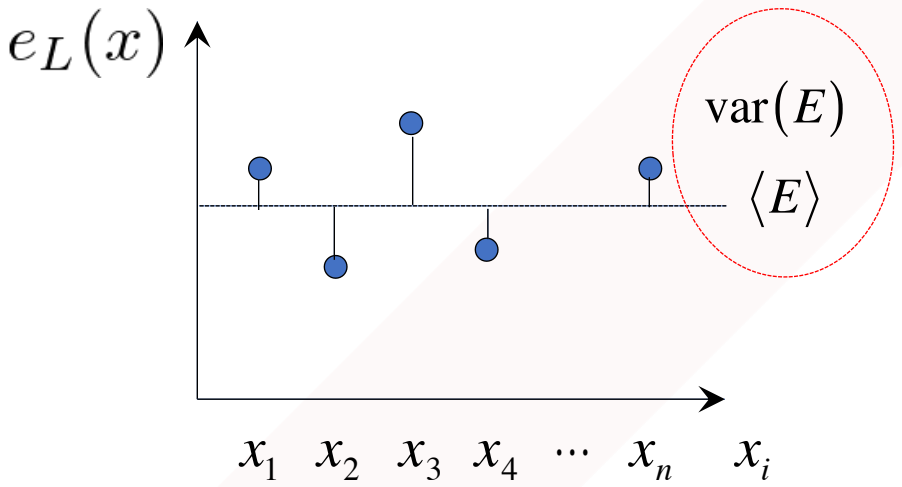
4. VMC Do a VMC run. **turborvb**

5. LRDMC LRDMC with the optimized WF. **turborvb**

VMC input file

```
&simulation
    itestr4=2
    ngen=10000
    maxtime=10800
    iopt=1
/

&pseudo
/

&vmc
    !epscut=0.0
/

&readio
    !iread=3
/

&parameters
    !ieskin=1
/
```

One-shot VMC



$$= \text{ngen} \times \text{nw}$$

ngen is the total number of Monte Carlo steps.

Default: nw is the number of MPI processes.

All information

```python
from scipy.io import FortranFile
import numpy as np

# check length of fort.12
f = FortranFile('fort.12', 'r')
a = f.read_reals(dtype='float64')
column_length = len(a)
f.close()

# start reading fort.12
head = ("head", "<i")
tail = ("tail", "<i")
dt = np.dtype([head, ("a", "<{}d".format(column_length)), tail])
fd = open('fort.12', "r")
fort12 = np.fromfile(fd, dtype=dt, count=-1)
data_length=len(fort12)
fd.close()
# end reading fort.12

print(fort12)
```

```python
# for ngen=10
>>> fort12
array([(40, [  1.        ,   1.        , -11.23924971, -11.23924971, 126.32073395], 40),
    (40, [  1.        ,   1.        , -11.4465321 , -11.4465321 , 131.02309712], 40),
    (40, [  1.        ,   1.        , -11.25058355, -11.25058355, 126.57563015], 40),
    (40, [  1.        ,   1.        , -11.88021352, -11.88021352, 141.13947319], 40),
    (40, [  1.        ,   1.        , -10.89686295, -10.89686295, 118.74162225], 40),
    (40, [  1.        ,   1.        , -11.8906161 , -11.8906161 , 141.38675112], 40),
    (40, [  1.        ,   1.        , -10.50040878, -10.50040878, 110.25858451], 40),
    (40, [  1.        ,   1.        , -10.85804034, -10.85804034, 117.89704005], 40),
    (40, [  1.        ,   1.        , -11.3042634 , -11.3042634 , 127.78637111], 40),
    (40, [  1.        ,   1.        , -10.86745849, -10.86745849, 118.10165397], 40)],
    dtype=[('head', '<i4'), ('a', '<f8', (5,)), ('tail', '<i4')])
```

e(L), etc… -> written in fort.12

Reblocking

forcevmc.sh “bin”, “init”, “pulay”, or

turbo-genius.sh –j vmc -post -reb “bin”, -eq “init”

“bin”: the length of reblocking size

“init”: the length of equilibration steps (init * bin)

“pulay”: the ratio of the pulay term (1 is OK)

pip0.d=energy

```
#cat pip0.d

number of bins read =        1496
Energy =  -1.1379192772188327        1.7589095174214898E-004
Variance square =   1.7369139136828382E-003   2.7618833870090571E-005
Est. energy error bar =   1.7510470092362484E-004   3.9800256121536918E-006
Est. corr. time  =   2.6420266523220208       0.10738159557488412
```

forcevmc.dat=forces

```
Force component 1
Force    =  6.004763869201490E-003  4.997922374161991E-005
6.273565633363322E-007
Der Eloc =  6.927675852724724E-003  4.999242839793062E-005
<OH> =  0.557134685159244       7.437283601136703E-005
<O><H> = -0.557596141151006       7.447559481785158E-005
2*(<OH> - <O><H>) = -9.229119835232336E-004  2.922997214772288E-006
```

Ionic Forces:

```
# Constraints for forces: ion - coordinate
          1      1      3
```

The number of forces, atom index, direction

F1,z for the first atom will be calculated.

```
# Constraints for forces: ion - coordinate
          2      1      1      2      -3
```

F1,x and F2,z will be calculated, assuming, F = F1,x = - F2,z.

The output value (forcevmc.dat) is the sum of two forces, i.e., (F = F1,x - F2,z.)

If you want to calculate forces, please set “ieskin=1” in the &parameter section in your VMC input.

forcevmc.dat

forcevmc.dat=forces

```
Force component 1
Force    =  6.004763869201490E-003  4.997922374161991E-005
6.273565633363322E-007
Der Eloc =  6.927675852724724E-003  4.999242839793062E-005
<OH> =  0.557134685159244        7.437283601136703E-005
<O><H> = -0.557596141151006        7.447559481785158E-005
2*(<OH> - <O><H>) = -9.229119835232336E-004  2.922997214772288E-006
```

Force (total) $$F_\alpha = -\langle \frac{d}{d\mathbf{R}_\alpha} E_L \rangle - 2\left(\langle E_L \cdot \frac{d}{d\mathbf{R}_\alpha} \log(J^{1/2}\Psi) \rangle - \langle E_L \rangle \cdot \langle \frac{d}{d\mathbf{R}_\alpha} \log(J^{1/2}\Psi) \rangle \right),$$

Der Eloc:

(Hellmann-Feynman term)

2*(<OH> - <O><H>)

(Pulay term)

where, J is the Jacobian of the warp transformation. S Sorella, L Capriotti, ***J. Chem. Phys.*** 133, 234111 (2010).

```
# Constraints for forces: ion - coordinate
          2      1      1      2      -3
```

The output value (forcevmc.dat) is the sum of two forces, i.e., (F = F1,x - F2,z.)

How to choose basis set for Force calc.? (PBC)

TurboRVB employs the CRYSTAL periodic basis for PBC calculations:

$$\psi^{\mathrm{PBC}}_{l,m,I}(\mathbf{r};\zeta) = \sum_{\mathbf{T}_s} \psi_{l,m,I}(\mathbf{r}+\mathbf{T}_s;\zeta)\, e^{-i\mathbf{k}_s\cdot\mathbf{T}_s}$$

-PBC, pseudo potential:

Unfortunately, provided basis sets for open systems are redundant for periodic cases, so we recommend that one should cut several smaller exponents, typically, smaller than 0.10.

-PBC, all-electron:

The same for all-electron cases. Basis sets provided for open systems such as Basis set exchange [https://www.basissetexchange.org] are usually redundant for a periodic case, so we recommend that one should cut several smaller exponents, typically, smaller than 0.10.

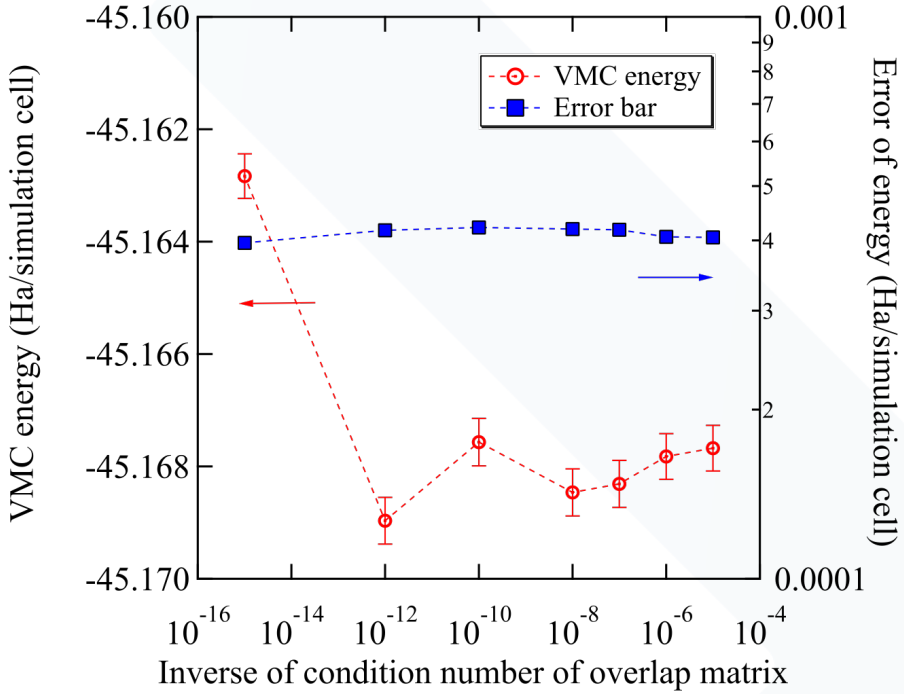
One can also use all-electron basis sets optimized for periodic systems such as ones provided in the CRYSTAL DFT code [https://www.crystal.unito.it/basis-sets.php].

Basis set redundancy for periodic cases

$$\Psi = J \times \Psi_{\mathrm{SD}} \qquad \phi_i^{\mathbf{R}} = \sum_{a,l} c_{i,\{a,l\}} \psi_{\{a,l\}}^{R_a}$$

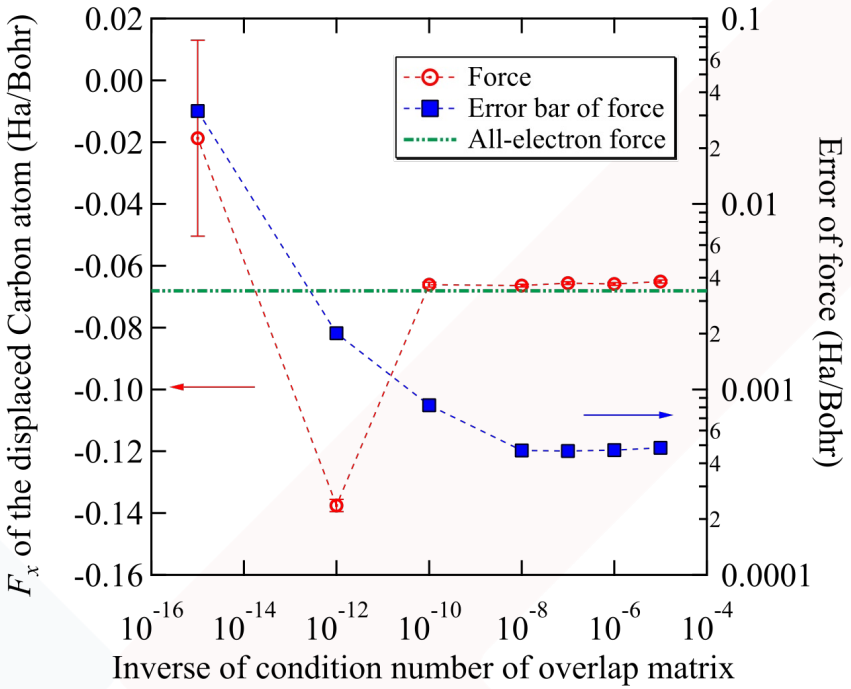
Linear dependency = the condition number of the overlap matrix (S).

- Diamond: Total Energy (E)

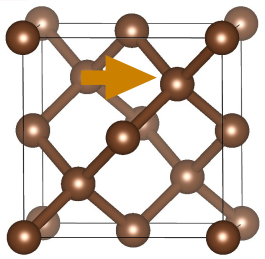


Large <- Lin. Dep. -> Small

- Diamond: Force (F)



Large <- Lin. Dep. -> Small



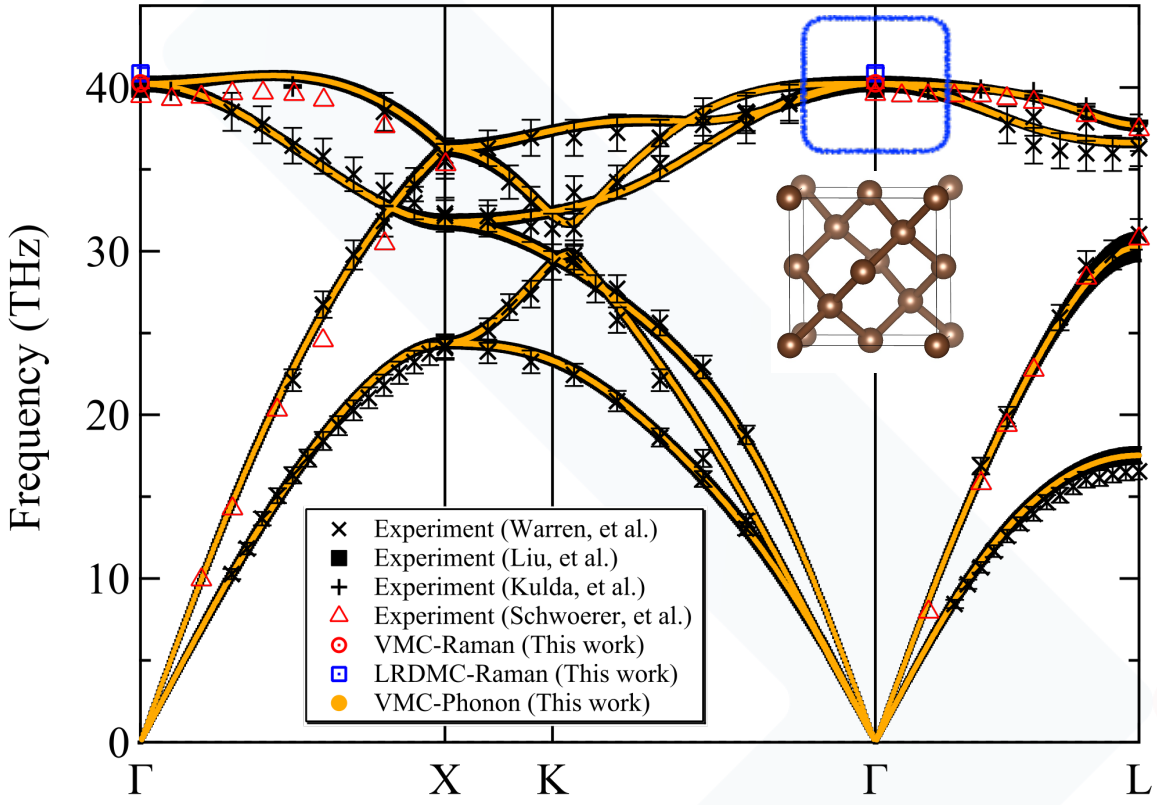
Only one C is displaced

K. Nakano *et al.*, ***Phys. Rev. B*** 103, L121110 (2021)

Recent progress: Phonon dispersion calculations

- Diamond: the conventional 2x2x2 supercell with the experimental lattice parameter
- The frozen phonon method implemented in Phonopy package.

A. Togo and I. Tanaka, *Scr. Mater.* 108, 1 (2015).



* including anharmonic corrections.

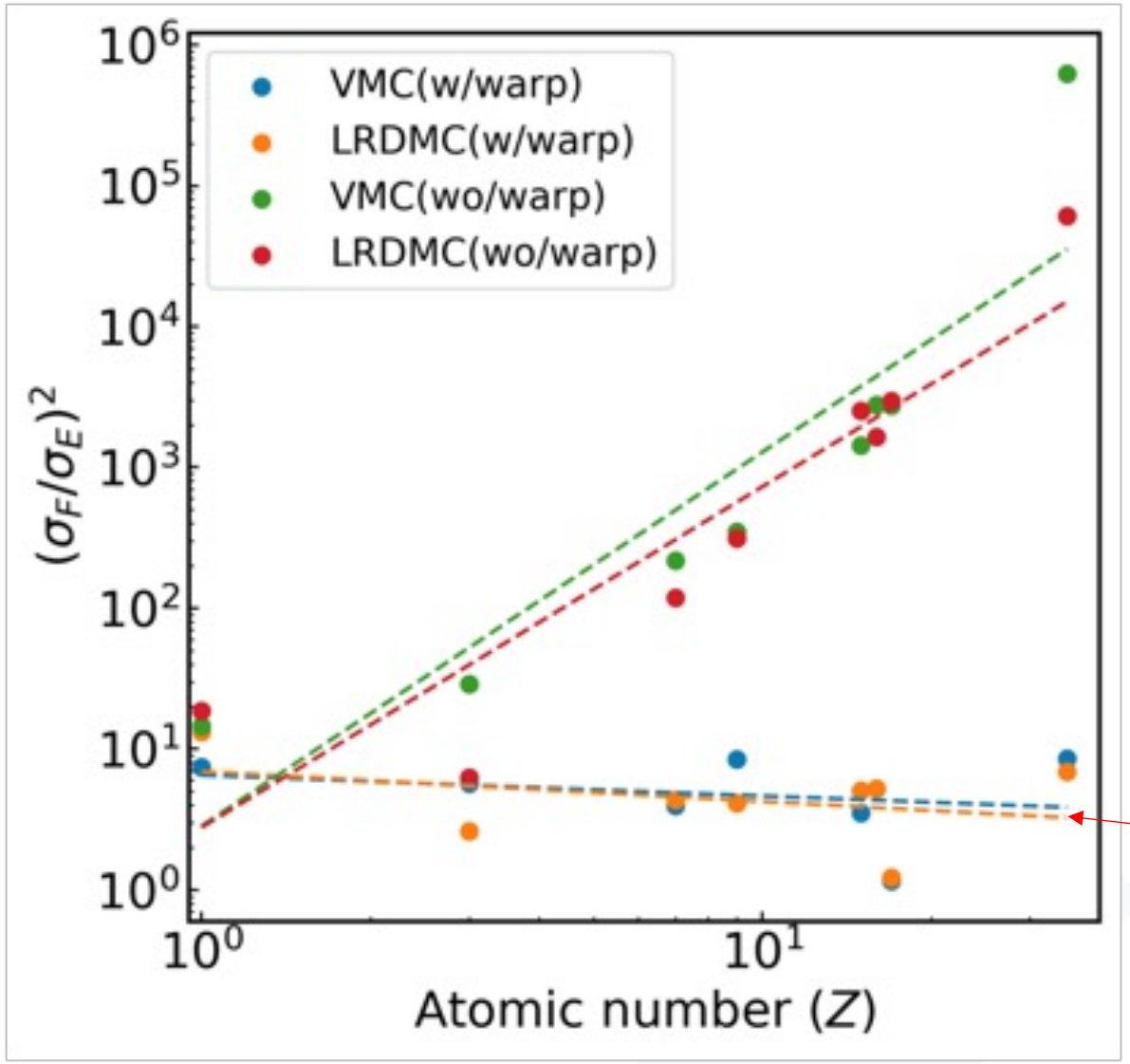
Raman Freq. (optical phonon at Γ)

DFT-LDA	38.55 THz
VMC	40.65(38) THz
Exp.	40.35 THz

** These are harmonic frequencies

K. Nakano *et al.*, ***Phys. Rev. B*** 103, L121110 (2021)

Importance of the Space-warp coordinate transformation



$(\sigma_F/\sigma_E)^2$ scales as $Z^{\sim 2.5}$ **without SWCT**,

consistent with QMCPACK group's paper

J. Tiihonen, et al. *J. Chem. Phys.* 154, 204111 (2021)

QMCPACK group shows that the scaling does not change even **with SWCT**…

No, the ratio is independent of Z !!

K. Nakano *et al.*, ***J. Chem. Phys.*** 156, 034101 (2022)

Energy derivative v.s. Force

$$\frac{dE}{dR_\alpha} = \langle \frac{\partial}{\partial R_\alpha} E_L \rangle + 2(\langle E_L \frac{\partial}{\partial R_\alpha} \log \Psi \rangle - \langle E_L \rangle \langle \frac{\partial}{\partial R_\alpha} \log \Psi \rangle)$$

$$+ \sum_{i=1}^{N_{\mathrm{par}}} \frac{\partial E}{\partial c_i} \frac{\partial c_i}{\partial R_\alpha}$$

Additional terms!!

JSD ? $\phi_i^{\mathbf{R}} = \sum_{a,l} c_{i,\{a,l\}} \psi_{\{a,l\}}^{R_a}$

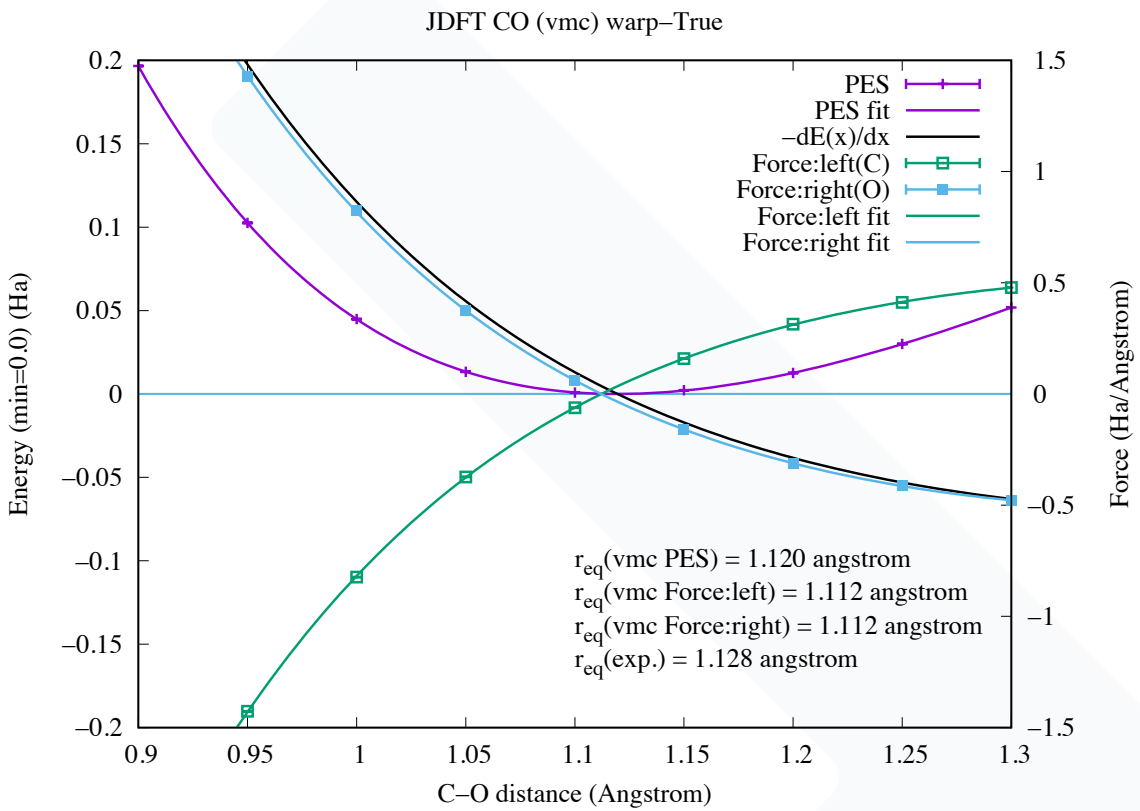
1. The system is already at a variational minimum. $\frac{\partial E}{\partial c_i} = 0$ → JAGPs ○

2. The variational parameters are not allowed to vary with changing the atomic pos. $\frac{\partial c_i}{\partial R_\alpha} = 0$

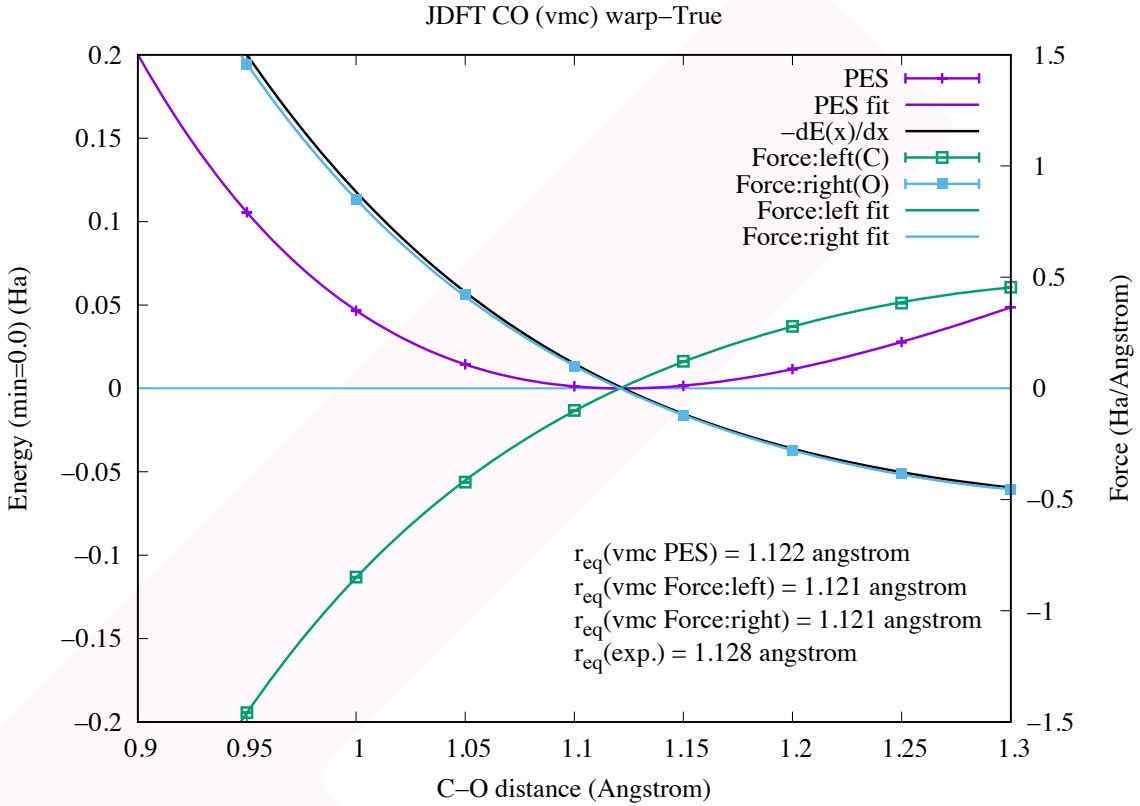
J. Tiihonen et al., ***J. Chem. Phys.*** 154, 204111 (2021)

Carbon monoxide: PES v.s. Force

All-electron calculations, VMC (**JDFT**). Jastrow factors were optimized for each C-O distance.

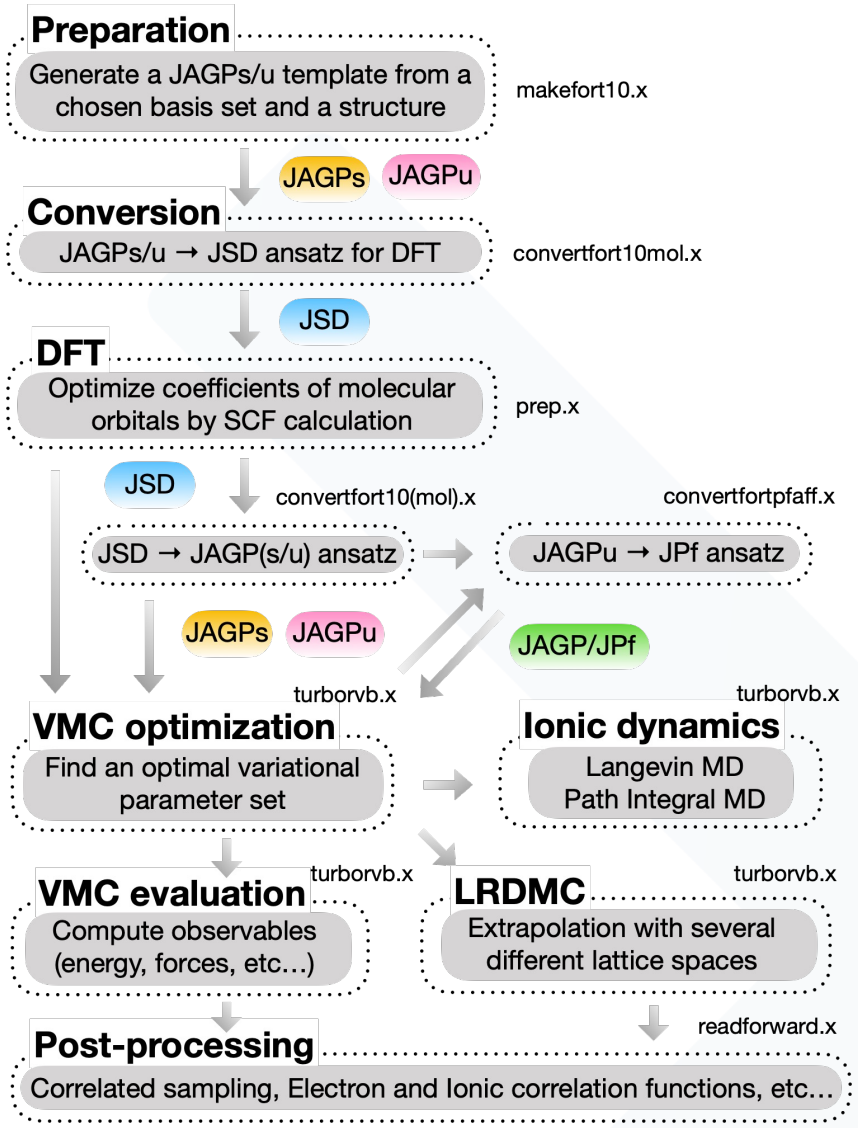


Basis = cc-pVDZ



Basis = cc-pVQZ

A typical workflow in TurboRVB (TurboGenius)



= Workflow =

1. Prepare a structure and basis set **makefort10**

2. DFT Construct a reasonable initial WF! **prep**

3. VMC-opt Optimize the wavefunction **turborvb**

4. VMC Do a VMC run. **turborvb**

5. LRDMC LRDMC with the optimized WF. **turborvb**

Lattice regularized DMC (LRDMC)

The projection technique to filer out the ground state from a trial wave function (typically, optimized by VMC).

M. Casula et al., ***Phys. Rev. Lett.*** 95, 100201 (2005)

$$|\Upsilon_0\rangle \propto \lim_{M\to\infty} \left(\mathbf{\Lambda} - \hat{\mathcal{H}}\right)^M |\Psi_\mathrm{T}\rangle$$

$$= \lim_{M\to\infty} (\lambda - E_0)^M \left[a_0 |\Upsilon_0\rangle + \sum_{n\neq 0} \left(\frac{\lambda - E_n}{\lambda - E_0}\right)^M a_n |\Upsilon_n\rangle \right],$$

Since $\dfrac{\lambda - E_n}{\lambda - E_0} < 1$ the projection filters out the ground state WF from a given trial WF

In TurboRVB, “etry” is the corresponding parameter. $\lambda = -2 \times \mathrm{etry}$

e.g., one can use a VMC energy for etry.

Lattice discretization

To apply the GFMC method for continuous systems.

M. Casula et al., ***Phys. Rev. Lett.*** 95, 100201 (2005)

$$\Delta_i f\left(x_i, y_i, z_i\right) \approx \Delta_i^a f\left(x_i, y_i, z_i\right)$$
$$\equiv \frac{1}{a^2}\left\{\left[f\left(x_i+a\right)-f\left(x_i\right)\right]+\left[f\left(x_i-a\right)-f\left(x_i\right)\right]\right\}$$
$$\leftrightarrow y_i \leftrightarrow z_i,$$

$$V^a(\mathbf{x}) = V(\mathbf{x}) + \frac{1}{2}\left[\frac{\sum_i\left(\Delta_i^a-\Delta_i\right)\Psi_{\mathrm{G}}(\mathbf{x})}{\Psi_{\mathrm{G}}(\mathbf{x})}\right].$$

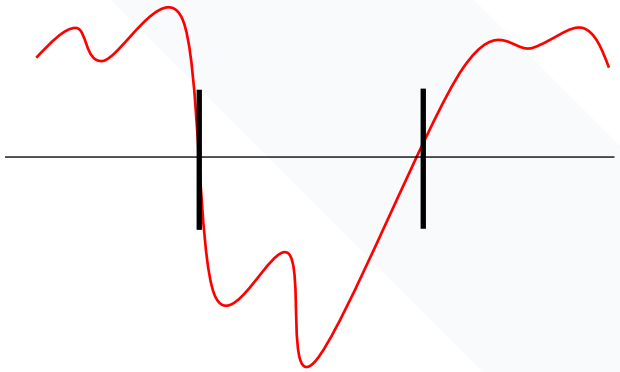
In TurboRVB, "**alat**" is the corresponding parameter. The unit is bohr.

Since the Trotter-Suzuki decomposition is not needed in the LRDMC framework, the "time-step" error does not exist in LRDMC unlike DMC, but this "lattice-space" error exists instead. We need extrapolation for alat. (later)

Fixed-node approximation

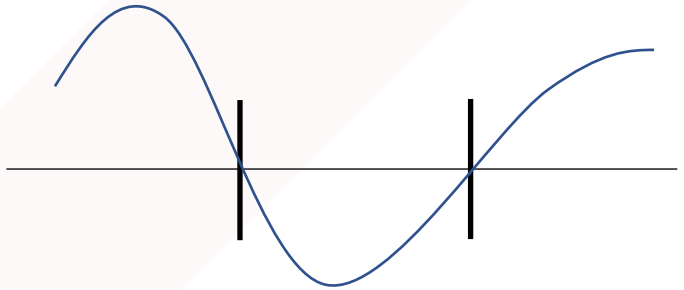
The Green's function cannot be made strictly positive for fermions; therefore, the fixed-node (FN) approximation has to be introduced in order to avoid the sign problem.

Trial WF



Projection.

Fixed-node WF



The nodal surface never changes during the simulation! i.e., Only the amplitude is relaxed.

Input file for Lattice regularized DMC (LRDMC)

M. Casula et al., ***Phys. Rev. Lett.*** 95, 100201 (2005)

```
&simulation
    itestr4=-6
    ngen=24100
    iopt=1
    maxtime=10800
/

&pseudo
/

&dmclrdmc
    tbra=0.1
    etry=-5.50
    Klrdmc=0.0
    alat=-0.40
    !iesrandoma=.true.
    !alat2=0.0
    gamma=0.0
    parcutg=1
/

&readio
    !iread=2
/

&parameters
    !ieskin=1
/
```

Important parameters:

Itestr4 = -6: LRDMC

ngen: The number of iterations (branchings)

tbra: Projection time

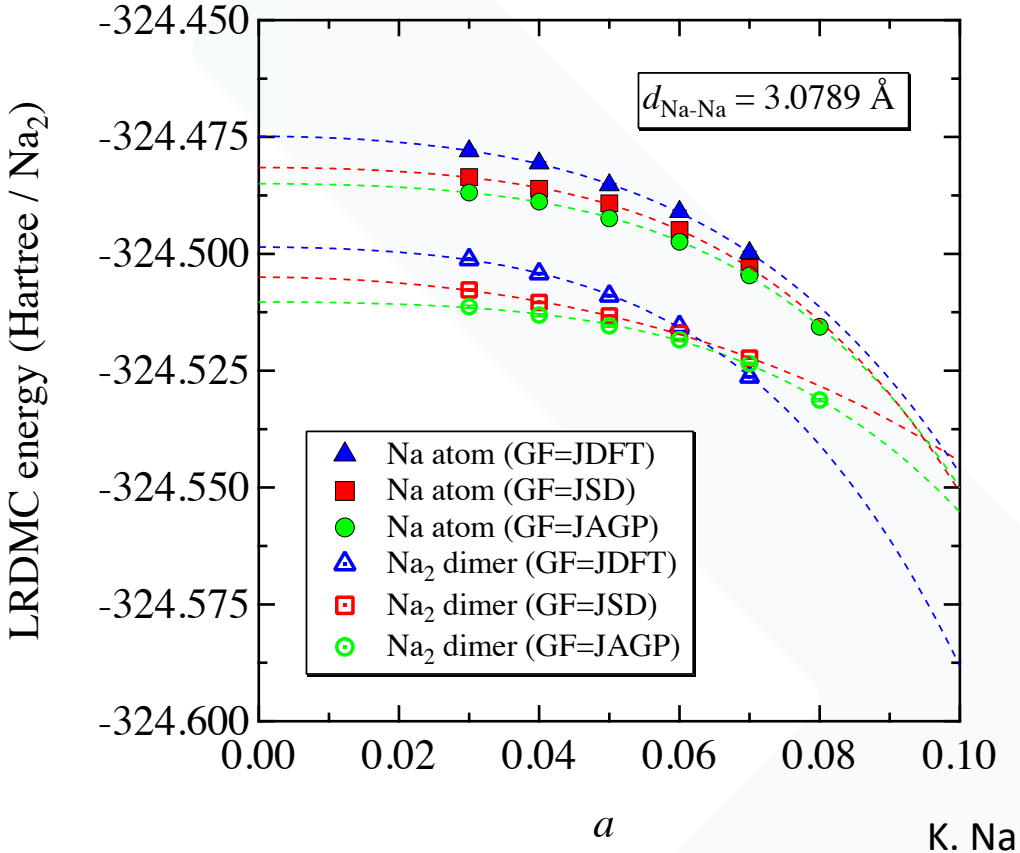
etry: Energy shift for the projection

alat: Coarser grid size (Bohr).

alat2: Denser grid size (Bohr).

$$\Delta_i f\left(x_i, y_i, z_i\right) \approx \Delta_i^a f\left(x_i, y_i, z_i\right)$$
$$\equiv \frac{1}{a^2}\left\{\left[f\left(x_i+a\right)-f\left(x_i\right)\right]+\left[f\left(x_i-a\right)-f\left(x_i\right)\right]\right\}$$

The extrapolation behaves well unlike the standard DMC!!



alat extrapolation with funvsa.x

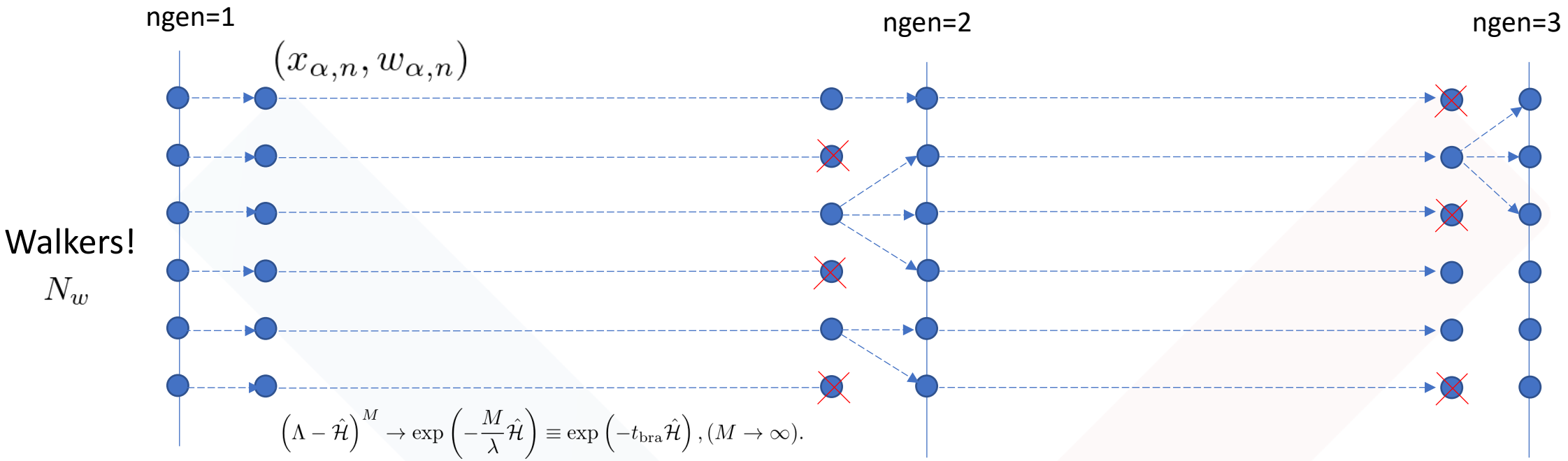
quartic: $E(a) = E(0) + k_1 \cdot a^2 + k_2 \cdot a^4$

quadratic: $E(a) = E(0) + k_1 \cdot a^2$

```
# See. Readme of funvsa.x in detail.
# 2=(up to a^4) number of data 4 1
2  5  4  1
0.10 -11.0850188375511 1.250592379643920E-004
0.20 -11.0854289356563 1.239503202184784E-004
0.30 -11.0855955871707 1.334024389855123E-004
0.40 -11.0860656088368 1.279739901272860E-004
0.50 -11.0868942724581 1.340429878094154E-004
```

K. Nakano et al., ***J. Chem. Theory Comput.*** 15, 4044-4055 (2019)

Branching and related parameters



weights and positions are updated.

Branching!!

1. Set the new weights $w'_{\alpha,n} = \bar{w} \equiv \frac{1}{N_w}\sum_\beta w_{\beta,n}.$
2. Select the new walkers $p_{\alpha,n} = w_{\alpha,n}/\sum_\beta w_{\beta,n},$

- “ngen” is the number of branchings! (ngen).
- The branching is done every τ_{nbra} steps. (tbra).

too small tbra -> The weights are not update.

too large tbra -> Only few walkers survive.

Check your output! Av. num. of survived walkers/ # walkers in the branching 0.99 > 0.90!

Reblocking

forcefn.sh “bin”, “corr”, “init”, “pulay”, or

turbo-genius.sh –j lrdmc -post -reb “bin”, -eq “init” -col “corr”

pip0_fn.d=energy

```
% cat pip0_fn.d
number of bins read =        1201
Energy =  -11.0854289356563       1.239503202184784E-004
Variance square =  0.126708380716482       1.148750765092961E-003
Est. energy error bar =  1.234807072779590E-004  2.503947626011507E-006
Est. corr. time  =   1.85075908836029       7.596952532743223E-002
Energy (ave) = -11.0854159959592 1.144905833254917E-004
```

forcefn.dat=forces

```
Force component 1
Force    =  6.004763869201490E-003  4.997922374161991E-005
6.273565633363322E-007
Der Eloc =  6.927675852724724E-003  4.999242839793062E-005
<OH> =  0.557134685159244       7.437283601136703E-005
<O><H> = -0.557596141151006       7.447559481785158E-005
2*(<OH> - <O><H>) = -9.229119835232336E-004  2.922997214772288E-006
```

“bin”: the length of reblocking (binning) size

“corr”: correcting factor

“init”: the length of equilibration steps (init * bin)

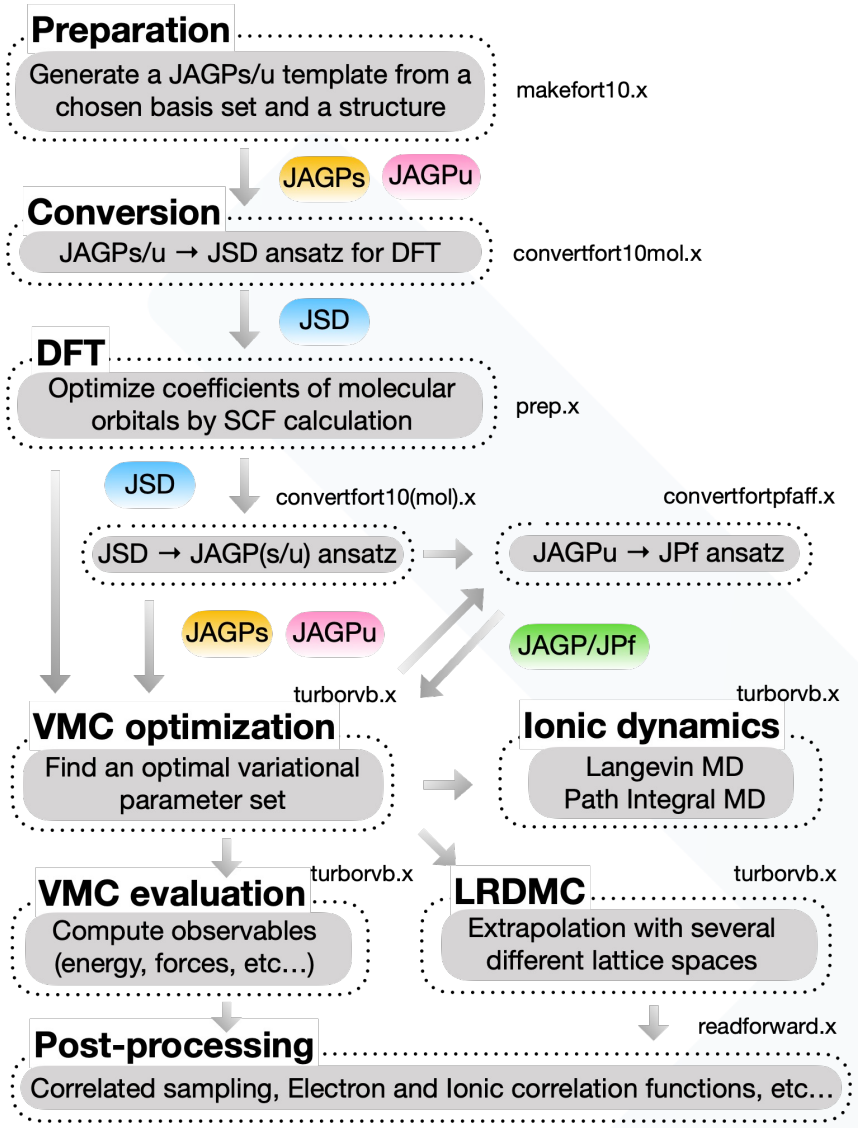
“pulay”: the ratio of the pulay term (1 is OK)

“corr”: correcting factor (p)

The average weights are stored and are set to one for all walkers after each branching.

$$E_0 \approx \frac{\sum_n \mathcal{G}_n^p e_L(x_n)}{\sum_n \mathcal{G}_n^p} \quad \mathcal{G}_n^p = \prod_{j=1}^{p} \bar{w}_{n-j},$$

A typical workflow in TurboRVB (TurboGenius)



= Workflow =

1. Prepare a structure and basis set **makefort10**

2. DFT Construct a reasonable initial WF! **prep**

3. VMC-opt Optimize the wavefunction **turborvb**

4. VMC Do a VMC run. **turborvb**

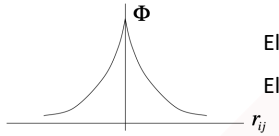
5. LRDMC LRDMC with the optimized WF. **turborvb**

The most difficult part in practice

3. VMC Optimize wavefunctions and VMC run.

A wavefunction reads $\Psi_{AS}(\vec{R}) \times \exp(J(\vec{R}))$ ⟶

Anti-symmetric part. Jastrow factor.



Electron-ion.

Electron-Electron.

To satisfy the cusp.

Jastrow factor = $\exp(J(\vec{R}))$ No effect on the nodal surface!! $\Psi(\vec{R}) \equiv \Psi(\vec{r_1}, \vec{r_2}, \cdots, \vec{r_N}) = 0$

Anti-symmetric part = $\Psi_{AS}(\vec{R})$ Determines the nodal surface. Its initial guess is taken from a DFT calculation!!

1st Step JSD 2nd Step JAGPs

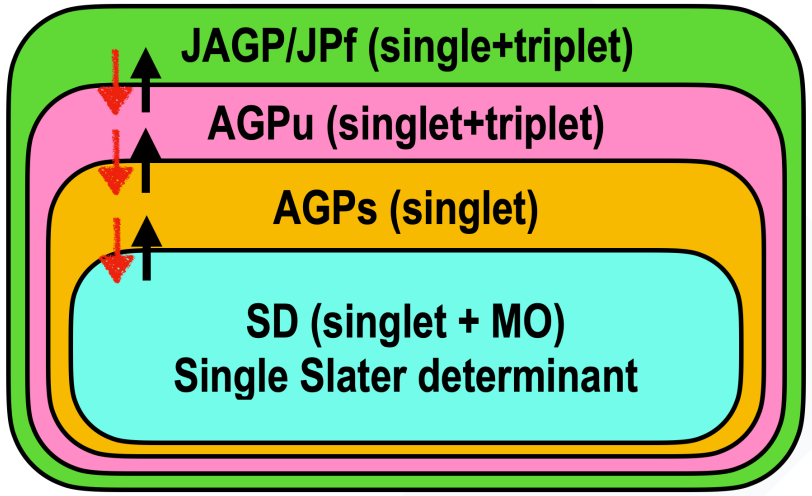
Conversion tools (convertfort10.x, convertpfaff)

Input: convertfort.input, fort.10_in, fort.10_out → Binary: convertfort10.x → Output:fort10_new

Convergfort10.x is a tool for converting a WF type., e.g., JSD → JAGPs JAGPu

Input: fort.10_in, fort.10_out → Binary: convertfortpfaff.x → Output:fort10_new

Convertpfaff.x is a tool for projecting a WF., e.g., JAGPu → JAGP/JPf



→ *Conversion (no loss of information)*

→ *Projection (loss of information)*

Correlated sampling

Input: readforward.input, fort.10, fort.10_corr → Binary: readforward.x → Output:corrsampling.dat

readforward.x enables us to calculate the difference in two WF using the correlated sampling.

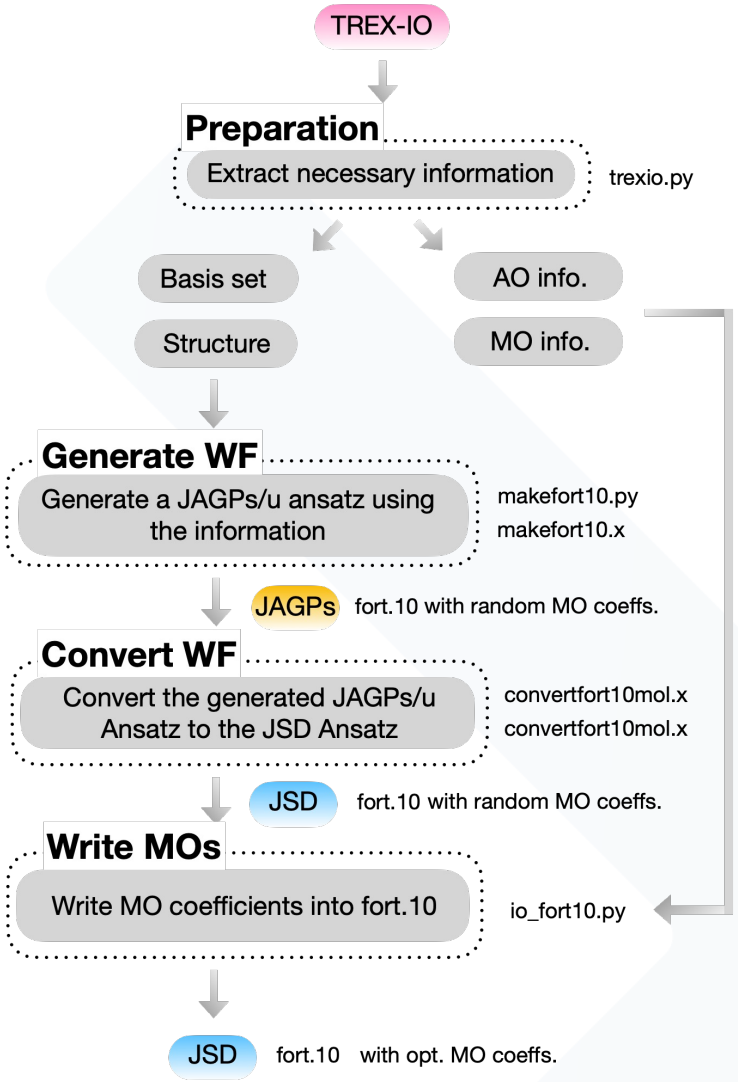
JSD ⟷ JAGPs

- The difference in energies
- The Overlap between the two WFs (the maximum is unity)

```
%cat corrsampling.dat

reference energy: E(fort.10)  -0.110045875E+02     0.252368934E-01
reweighted energy: E(fort.10_corr)  -0.110045875E+02     0.252368985E-01
reweighted difference: E(fort.10)-E(fort.10_corr)   -0.148834687E-07     0.316227766E-07
Overlap square : (fort.10,fort.10_corr)    0.999999998E+00     0.316227766E-07
```

If the overlap is unity, it means that the conversion has been done without losing any information.

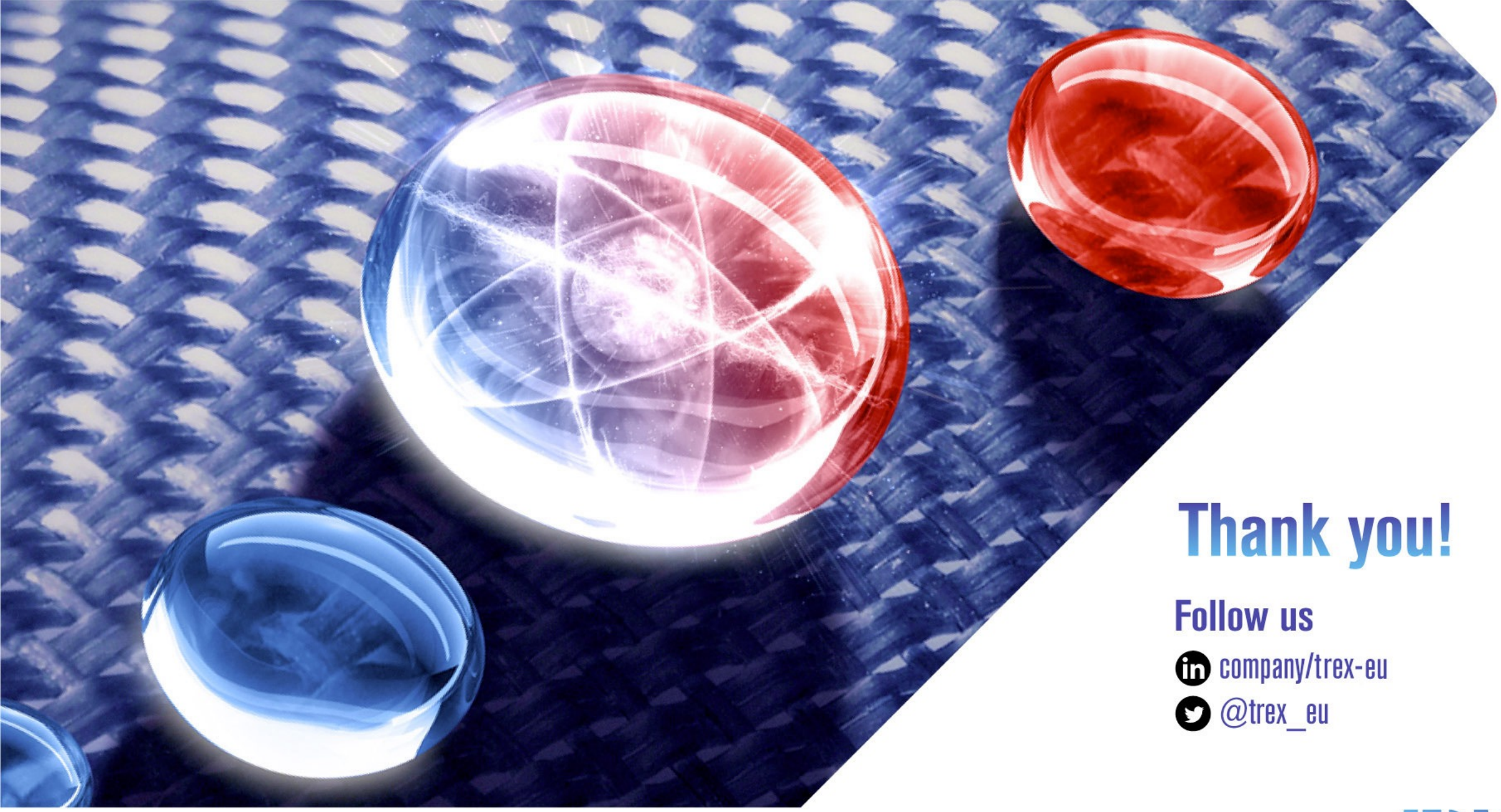
Workflow of the TREXIO-TurboRVB converter



First, the converter generates a TurboRVB WF file using only basis set and structure information stored in a TREX-IO file.

Then, the converter writes the MO information stored in a TREX-IO file into the generated WF file.

This is because of the complication of the TurboRVB WF format.



Thank you!

Follow us

company/trex-eu

@trex_eu

Targeting Real Chemical Accuracy at the Exascale project has received funding from the European Union Horizon 2020 research and innovation programme under Grant Agreement **No. 952165.**

TREX
Targeting Real chemical accuracy at the EXascale