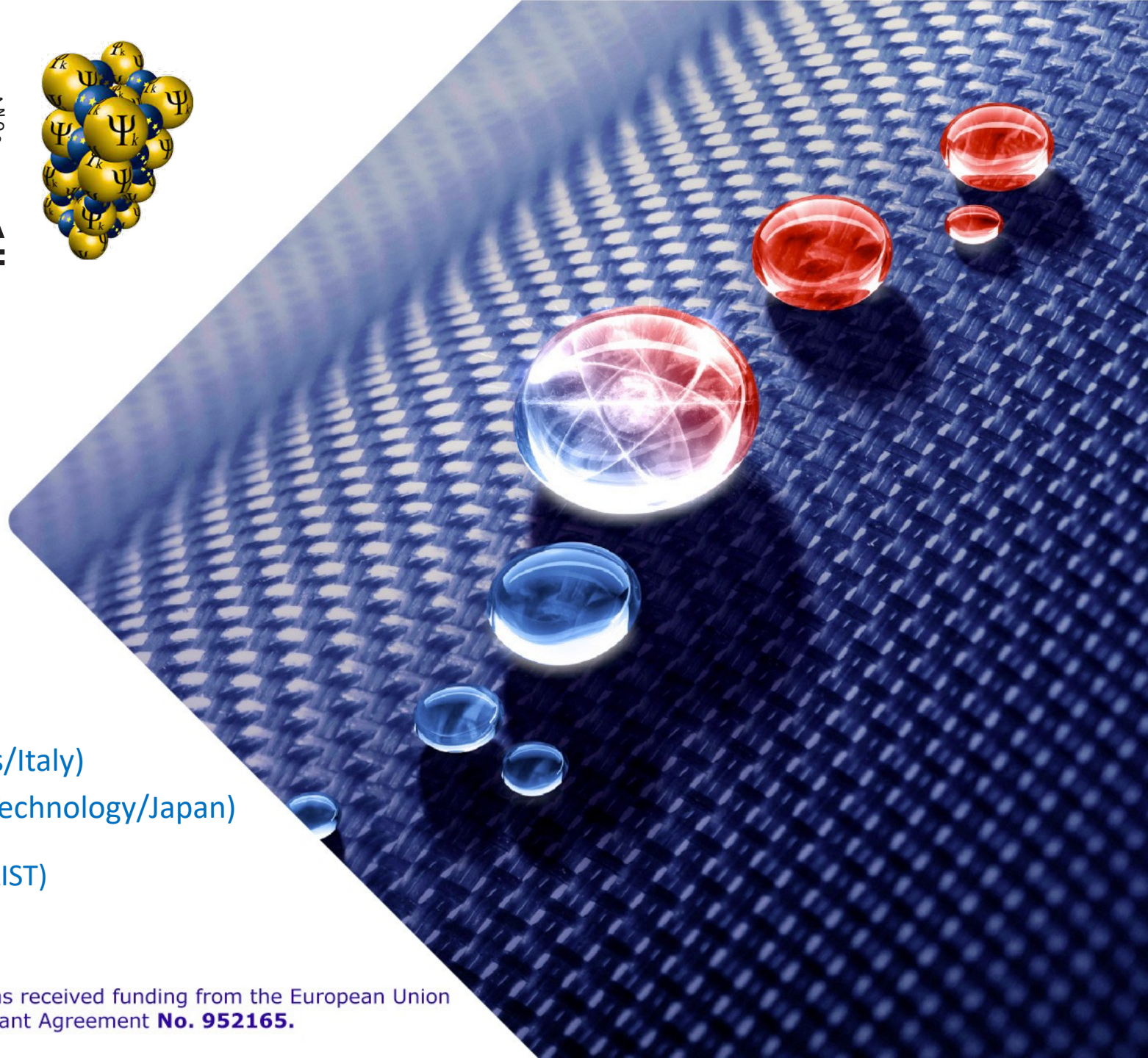
Targeting Real chemical accuracy at the EXascale

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)

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.**

compilation-and-installation
dft
general
lectures
lrdmc
random
turbo-genius
tutorials
user-manuals
vmc
vmc-optimization
wf-conversion

- Compilation and Installation: Compilations

- DFT: DFT calculations (prep.x)

- VMC: Single-shot VMC, also Force calculations

- VMC-optimization: Optimizations

- LRDMC: LRDMC.

- Turbo-Genius: Turbo-Genius python wrappers

- User manuals and Tutorials.

Others??

Targeting Real chemical accuracy at the EXascale

SISSA

Day2:

- Various ansatz (JDFT/JsAGPs/JAGPu/JAGP)
- Wavefunction conversion
- Wavefunction optimization

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

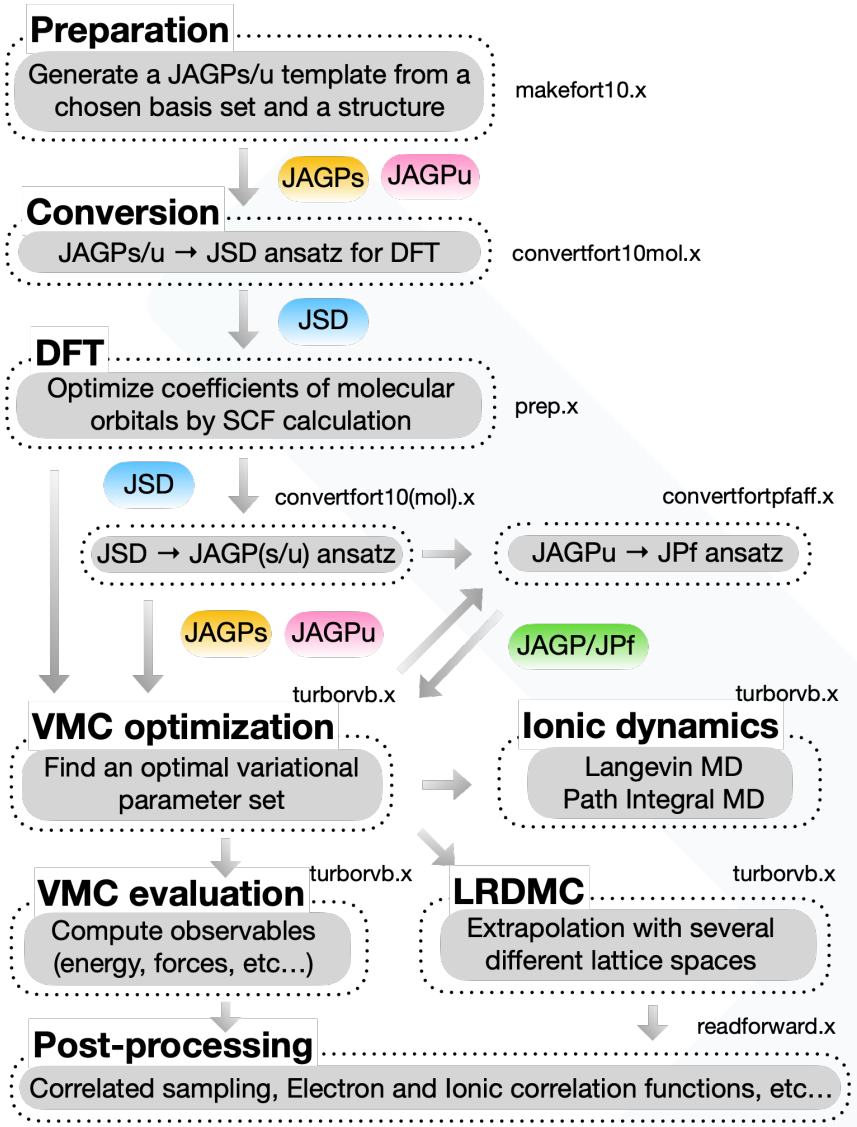
SISSA

Day2:

- Various ansatz (JDFT/JsAGPs/JAGPu/JAGP)
- Wavefunction conversion
- Wavefunction optimization

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.**

A typical workflow in TurboRVB (in detail)



= Workflow =

1. Prepare a structure and basis set **makefort10.x** **Day1**

2. DFT Construct a reasonable initial WF! **prep.x** **Day1**

3. VMC-opt Optimize the wavefunction **turborvb.x** **Day2**

4. VMC Do a VMC run. **turborvb.x** **Day3**

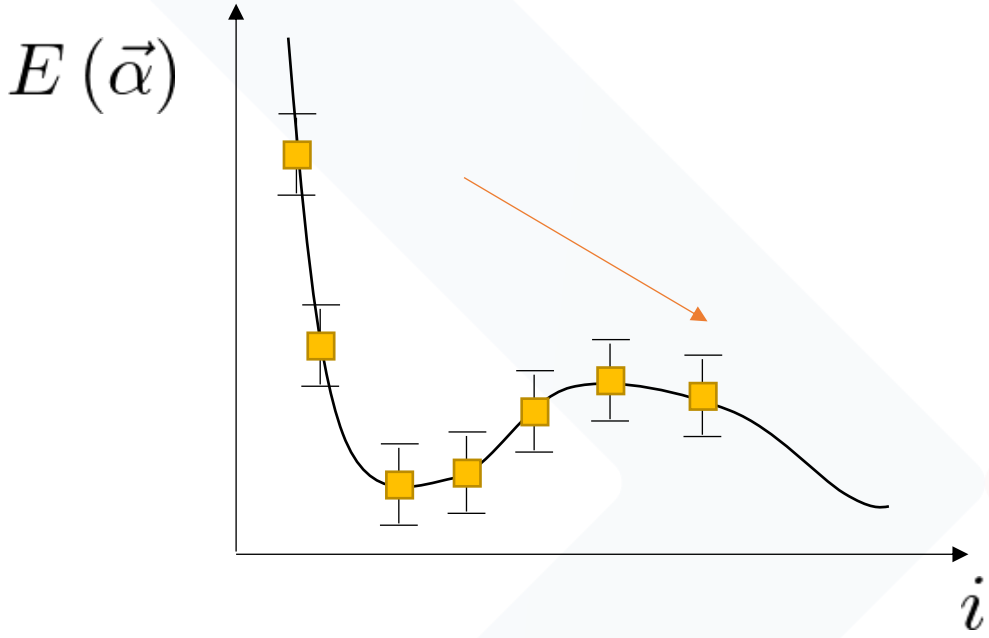
5. LRDMC LRDMC with the optimized WF. **turborvb.x** **Day4**

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.,

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

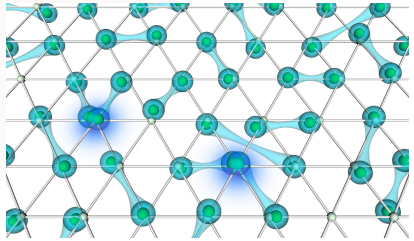
Ansatz Hierarchy

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

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

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

More complex.

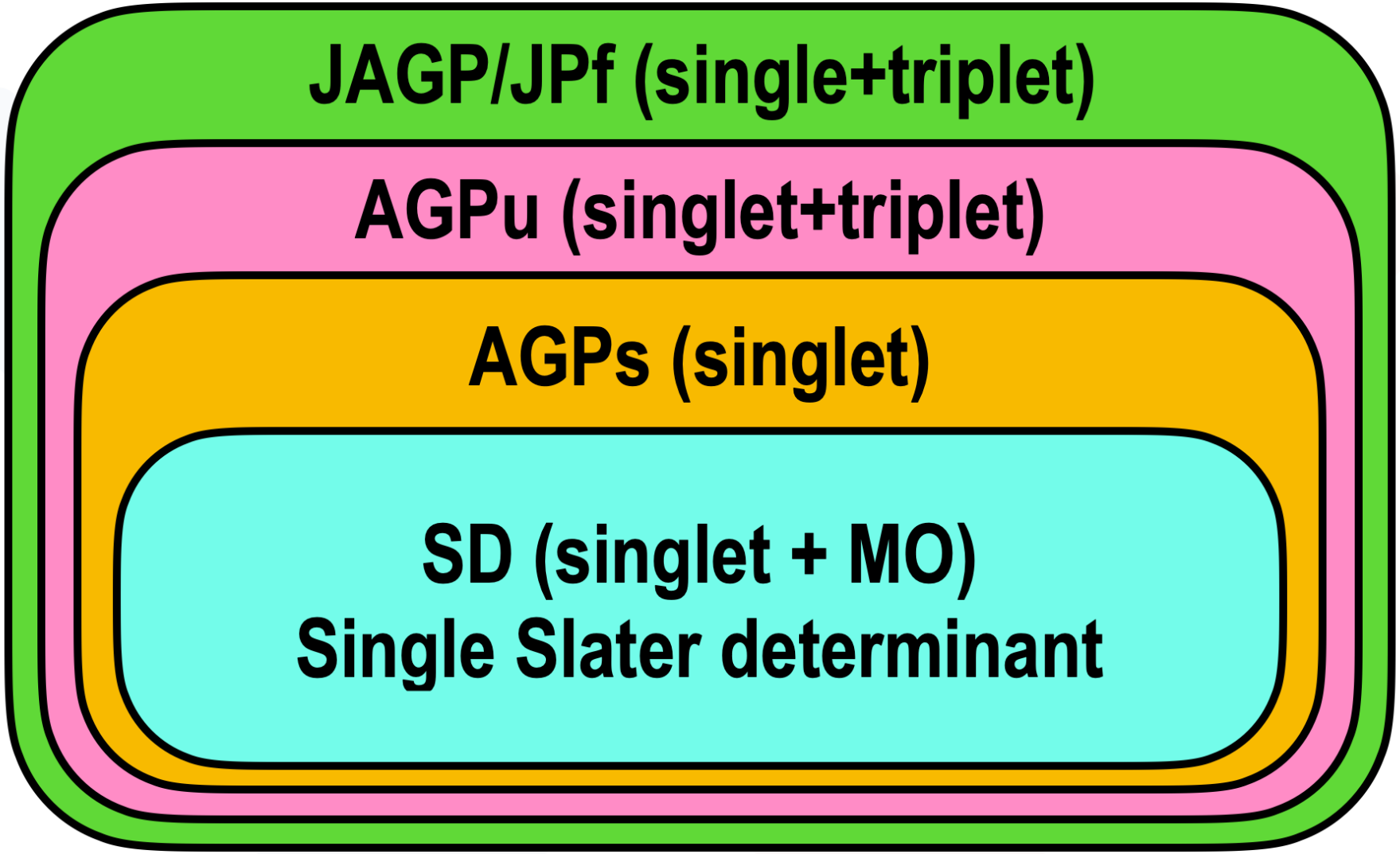


c.f. P.W. Anderson

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”.

Ansatz hierarchy in TurboRVB



The most general ansatz is represented as the Pfaffian.

$$\Phi_{\mathrm{Pf}} = \mathrm{Pf}(G) \equiv \frac{1}{2^{N/2}(N/2)!} \sum_{P \in S_N} \epsilon_P G_{P(1),P(2)} \cdots G_{P(N-1),P(N)}$$

$$G = \left[\begin{array}{c|c} G_{uu} & G_{ud} \\ \hline G_{du} & G_{dd} \end{array}\right]$$

$$g_{uu}(\mathbf{i},\mathbf{j}) = f_{uu}(\mathbf{r}_i,\mathbf{r}_j)\,|\uparrow\uparrow\rangle \qquad g_{dd}(\mathbf{i},\mathbf{j}) = f_{dd}(\mathbf{r}_i,\mathbf{r}_j)\,|\downarrow\downarrow\rangle$$

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

→ JAGP/JPf

Considering only different spins

$$\mathrm{Pf}\left(\begin{array}{c|c} 0 & G_{ud} \\ \hline -G_{ud}^T & 0 \end{array}\right) = (-1)^{\frac{N/2\times(N/2-1)}{2}} \det(G_{ud})$$

$$\Phi_{\mathrm{AGPs/AGPu}} = \det(G_{ud})$$

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

$$g_{ud}(\mathbf{i},\mathbf{j}) = f_S(\mathbf{r}_i,\mathbf{r}_j)\frac{|\uparrow\downarrow\rangle - |\downarrow\uparrow\rangle}{\sqrt{2}} + f_T(\mathbf{r}_i,\mathbf{r}_j)\frac{|\uparrow\downarrow\rangle + |\downarrow\uparrow\rangle}{\sqrt{2}} \quad \rightarrow \text{JAGPu}$$

Paring functions are parametrized:

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

They are not indices for the electrons, but for the basis!!

They are variational parameters, and written in fort.10!!

```
#          Nonzero values of detmat
           1       5      9.4217531017743 91E-002
           1       6      9.4217531017743 91E-002
           1       7      9.4217531017743 91E-002
```

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!

Coefficients of the Determinant part (JAGPu case)

```
#          Nonzero values of detmat
        1       1       8.321544938822982E-001 <- singlet
        ....
        1       5       9.421753101774391E-002 <- singlet
        1       6       9.421753101774391E-002
        1       7       9.421753101774391E-002
        2       1       3.485892384239842E-003 <- triplet
        2       2       3.589529849283749E-001 <- singlet
        2       3       2.489548797987997E-002 <- singlet
        3       1       1.112333456889842E-003 <- triplet
        3       2       2.585777744345490E-001 <- triplet
        3       3       3.936485649473937E-002 <- singlet
```

$A_{\{a,l\},\{b,m\}}$

- is symmetric for the singlet part
- is skew-symmetric for the triplet part

$$A_S = \begin{pmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ & A_{22} & \dots & A_{2n} \\ & & \ddots & \vdots \\ & & & A_{nn} \end{pmatrix} \quad A_T = \begin{pmatrix} 0 & -A_{21} & \dots & -A_{n1} \\ A_{21} & 0 & \dots & -A_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n1} & A_{n2} & \dots & 0 \end{pmatrix}$$

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

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

→ JAGPu

The most general paring function (Pfaffian ansatz).

```
# This is a C2-dimer case
# The number of basis set for each carbon is 4

#       Nonzero values of detmat

                                          <- (1,1) is always zero (G is skew-symmetric).
        1          2 -2.917621798712210E-002  <- A_{up,up}, triplet
        1          3  8.326474500954891E-003  <- A_{up,up}, triplet
        1          4 -0.228326252284219       <- A_{up,up}, triplet
        1          5  0.470855192339553       <- A_{up,up}, triplet
        1          6 -3.285258904186700E-002  <- A_{up,up}, triplet
        1          7 -5.097409720647310E-003  <- A_{up,up}, triplet
        1          8  5.679495868355650E-002  <- A_{up,up}, triplet
        1          9  0.684164602152446       <- A_{up,dn}, singlet
        ....
        1         16 -0.104285811627841       <- A_{up,dn}, singlet
        2          3 -2.076224374212450E-002  <- A_{up,up}, triplet
        ...
        2          8 -4.145465677435435E-003  <- A_{up,up}, triplet
        2          9  3.735515724267560E-003  <- A_{up,dn}, triplet
        2         10  0.520587530210701       <- A_{up,dn}, singlet
        ...
        2         16  4.428757569068110E-003  <- A_{up,dn}, singlet
        ...
        9         10  4.813787735439980E-003  <- A_{dn,dn}, triplet
        ...
       15         16  9.827312227017149E-003  <- A_{dn,dn}, triplet
```

From index 1 to 8 -> spin-up
From index 9 to 16 -> spin-dn

$$A = \left[\begin{array}{c|c} A_{\mathrm{up-up}} & A_{\mathrm{up-dn}} \\ \hline A_{\mathrm{dn-up}} & A_{\mathrm{dn-dn}} \end{array}\right]$$

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

We have 4 basis for each C atoms.
=> 16 * 16 matrix

$$A_{\mathrm{up-up}} = \begin{pmatrix} 0 & A_{1,2} & \cdots & A_{1,8} \\ -A_{1,2} & 0 & \cdots & A_{2,8} \\ \vdots & \vdots & \ddots & \vdots \\ -A_{1,8} & -A_{2,8} & \cdots & 0 \end{pmatrix}$$

$$A_{\mathrm{dn-dn}} = \begin{pmatrix} 0 & A_{9,10} & \cdots & A_{9,16} \\ -A_{9,10} & 0 & \cdots & A_{10,16} \\ \vdots & \vdots & \ddots & \vdots \\ -A_{9,16} & -A_{10,16} & \cdots & 0 \end{pmatrix}$$

The most general paring function (Pfaffian ansatz).

$$A = \left[\begin{array}{c|c} A_{\mathrm{up-up}} & A_{\mathrm{up-dn}} \\ \hline A_{\mathrm{dn-up}} & A_{\mathrm{dn-dn}} \end{array}\right]$$

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

```
# This is a C2-dimer case
# The number of basis set for each carbon is 4

#       Nonzero values of detmat

                                          <- (1,1) is always zero (G is skew-symmetric).
       1          2 -2.917621798712210E-002  <- A_{up,up}, triplet
       1          3  8.326474500954891E-003  <- A_{up,up}, triplet
       1          4 -0.228326252284219       <- A_{up,up}, triplet
       1          5  0.470855192339553       <- A_{up,up}, triplet
       1          6 -3.285258904186700E-002  <- A_{up,up}, triplet
       1          7 -5.097409720647310E-003  <- A_{up,up}, triplet
       1          8  5.679495868355650E-002  <- A_{up,up}, triplet
       1          9  0.684164602152446       <- A_{up,dn}, singlet
       ....
       1         16 -0.104285811627841       <- A_{up,dn}, singlet
       2          3 -2.076224374212450E-002  <- A_{up,up}, triplet
       ...
       2          8 -4.145465677435435E-003  <- A_{up,up}, triplet
       2          9  3.735515724267560E-003  <- A_{up,dn}, triplet
       2         10  0.520587530210701       <- A_{up,dn}, singlet
       ...
       2         16  4.428757569068110E-003  <- A_{up,dn}, singlet
       ...
       9         10  4.813787735439980E-003  <- A_{dn,dn}, triplet
       ...
      15         16  9.827312227017149E-003  <- A_{dn,dn}, triplet
```

From index 1 to 8 -> spin-up
From index 9 to 16 -> spin-dn

We have 4 basis for each C atoms.
=> 16 * 16 matrix

$$A_{up-dn,S} = \begin{pmatrix} A_{1,9} & A_{1,10} & \cdots & A_{1,16} \\ & A_{2,10} & \cdots & A_{2,16} \\ & & \ddots & \vdots \\ & & & A_{8,16} \end{pmatrix}$$

$$A_{up-dn,T} = \begin{pmatrix} 0 & -A_{2,9} & \cdots & -A_{8,9} \\ A_{2,9} & 0 & \cdots & -A_{8,10} \\ \vdots & \vdots & \ddots & \vdots \\ A_{8,9} & A_{8,10} & \cdots & 0 \end{pmatrix}$$

Coefficients of the Jastrow part (spin-independent Jastrow)

```
#       Nonzero values of jasmat
        1       1  8.9714511763304 50E-009
        1       2  9.8538589183041 12E-002
        2       2  9.7037797224777 37E-002
```

$$J_{3/4}\left(\mathbf{r}_1\sigma_1,\ldots,\mathbf{r}_N\sigma_N\right)=\sum_{i<j}\left(\sum_{a,l}\sum_{b,m}M^{\sigma_i,\sigma_j}_{a,l,b,m}\chi_{a,l}(\mathbf{r}_i)\chi_{b,m}(\mathbf{r}_j)\right),$$

```
        ....
        1         45 -2.0427115443666 10E-004
        1         91 -6.9942723202272 31E-004    <- inhomogeneous onebody M.
```

$$J_1^{inh}\left(\mathbf{r}_1\sigma_1,\ldots,\mathbf{r}_N\sigma_N\right)=\sum_{i=1}^{N}\sum_{a=1}^{N_{\mathrm{at}}}\left(\sum_{l}M^{\sigma_i}_{a,l}\chi_{a,l}\left(\mathbf{r}_i\right)\right)$$

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

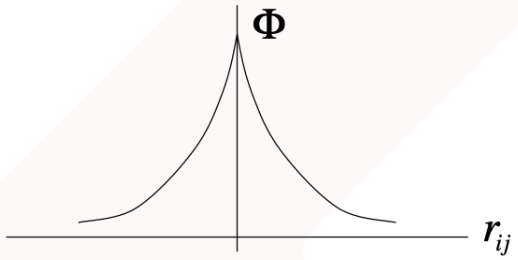
two-body and one-body parameters.

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

-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.

Targeting Real chemical accuracy at the EXascale

SISSA

Day2:

- Various ansatz (JDFT/JsAGPs/JAGPu/JAGP)
- Wavefunction conversion
- Wavefunction optimization

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.**

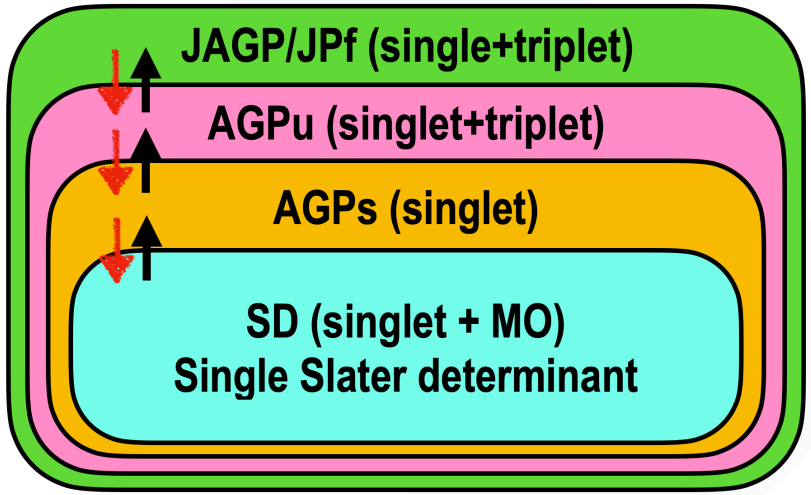
Conversion tools (convertfort10.x, convertpfaff)

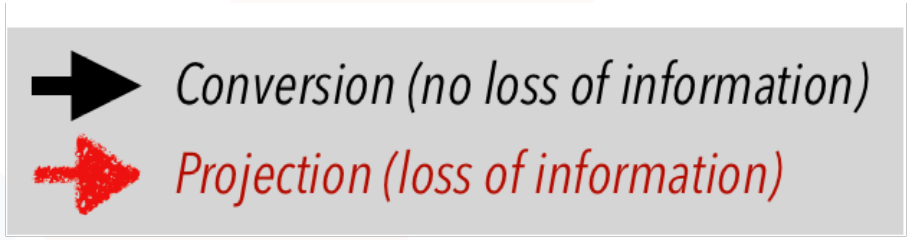
Input: convertfort.input, fort.10_in, fort.10_out → Binary: convertfort10.x → Output:fort10_new

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





Hybrid orbitals (convertfort10)

Hybrid orbitals (90000): In fort.10, 900000 indicates a hybrid orbital.

```
          #atom index,   the number of components, 900000
          #index of basis [1,2,....]
          #coefficients for basis [1,2,....]
          1          90      900000
          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  4.415629124781804E-004
-2.665471779107012E-003  6.462432564128138E-003 -2.835419348050690E-002
-1.801631060924397E-003  0.000000000000000E+000  0.488607316300470
 -1.08438499459258        1.00000000000000      -8.862166918962697E-002
-3.585186058053676E-006  1.336604741376926E-005 -1.034531513737405E-002
-9.682848944243672E-005  7.780031865084339E-005 -2.330742724611005E-002
 2.406091695918141E-004 -1.314165271323025E-004 -1.383980745357883E-002
-8.265782808695380E-005  6.484551172622954E-005 -6.461899158835088E-002
-1.224707476287876E-003  3.784895470189310E-004 -0.150215939301643
 3.279641185411990E-003  4.146447536354074E-004 -5.370349262603082E-002
-2.867011651206949E-003 -6.433419048360544E-004 -0.169339132045330
 0.000000000000000E+000  0.000000000000000E+000  5.388221636070670E-002
 3.865987502185416E-004  1.537139475756473E-004 -0.134079774055112
-4.310821717930281E-004 -2.609351310060059E-004  2.739963165919702E-002
 2.693581696773797E-002 -3.679623706959143E-005  1.393726496455493E-004
-1.728876043016380E-004  1.648834534792205E-004
```

makefort.10.input

```
ATOM_6
&shells
nshelldet=18
nshelljas=10
ndet_hyb=4
/
1   1   16
1   13.073594000000
```

$$\tilde{\Phi}_k^a = \sum_i^{N_{basis}^a} c_{i,k}^a \cdot \phi_i^a(\boldsymbol{r})$$

sp^2, sp^3, etc…

Hybrid orbitals can be added by “convertfort10.x”. We can decrease the number of variational parameters.

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.

- 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.

Targeting Real chemical accuracy at the EXascale

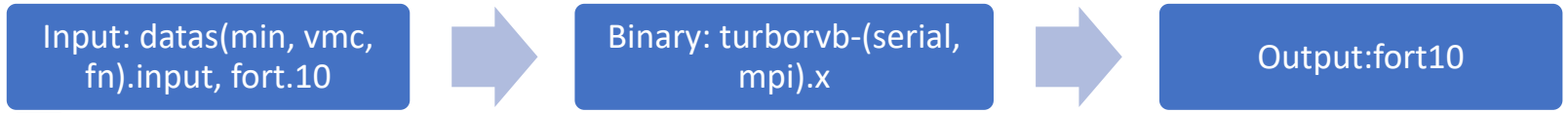
SISSA

Day2:

- Various ansatz (JDFT/JsAGPs/JAGPu/JAGP)
- Wavefunction conversion
- Wavefunction optimization

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.**

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



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

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

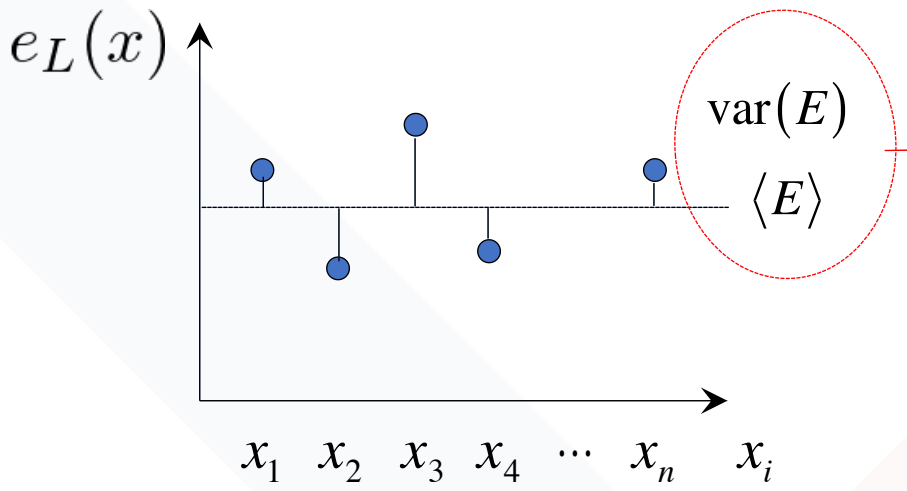
- Single-shot VMC run (itestr4=-2 in the &simulation namelist).
- VMC optimization (itestr4=-4,-5,-8,-9 in the &simulation namelist).
- Single-shot LRDMC run (itestr4=-6 in the &simulation namelist).
- Single-shot DMC run (itestr4=-5 in the &simulation namelist), but not maintained.

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

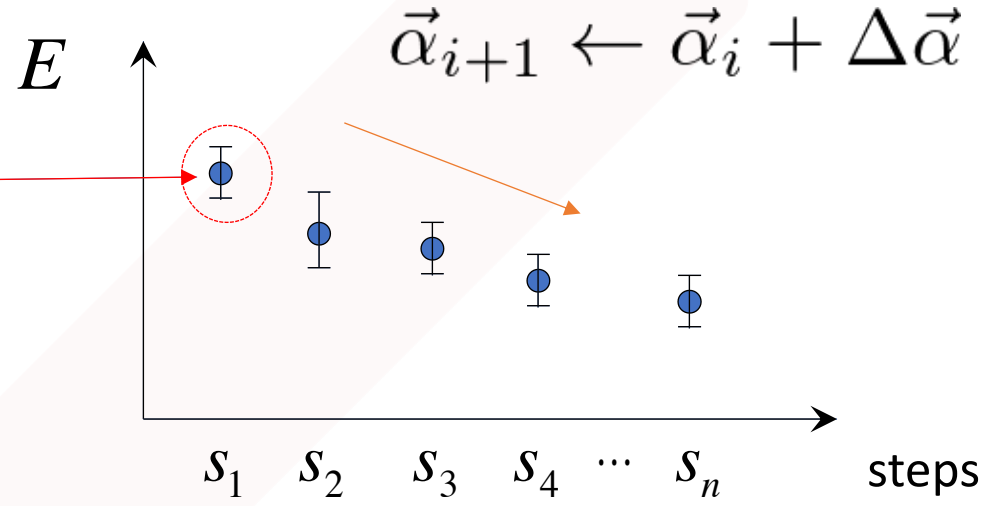
Each iteration



=

nweight × nw

Optimization



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

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

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

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

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

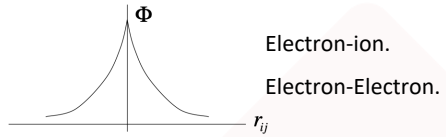
e.g., $s'_{i,i} = s_{i.i}(1+\varepsilon)$

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

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.

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!!

TurboRVB:

Both parts are optimized.

Stable nodal surface optimization

Independent of an initial DFT guess!?

Other QMC codes:

Only the Jastrow part is optimized.

Unstable nodal surface optimization

Dependent on an initial DFT guess!!

e.g., however, the optimization sometimes is stuck in a local minimum if the initialization was wrong etc…

Practical Optimization Procedure

Practical recommended optimization procedure. Avoid local minima…

1. Put a reasonable two-body Jastrow parameter (typically ~ 1.0 for twobody = -6, -15, -22),

%vi fort.10

2. Optimize homogenous and inhomogeneous one-body Jastrows (two-body and three-body are fixed),

iesfree=1, iesd=1 in the &parameters seciton.
twobodyoff=.true. and iesdtwobodyoff=.true. in the &optimization section.

3. Optimize three-body Jastrow part (two-body still fixed),

Remove !twobodyoff=.true.

4. Optimize two-body Jastrow part.

remove !twobodyoff=.true. and !iesdtwobodyoff=.true.

The other options

iessw=1 Opt. the determinant part.

iesup=1 Opt. the hyb. orbitals. / itestr4=-9/-4
+ Opt. the exponents (Det). / itestr4=-8/-5

iesm=1 Opt. the exponents (Jas).

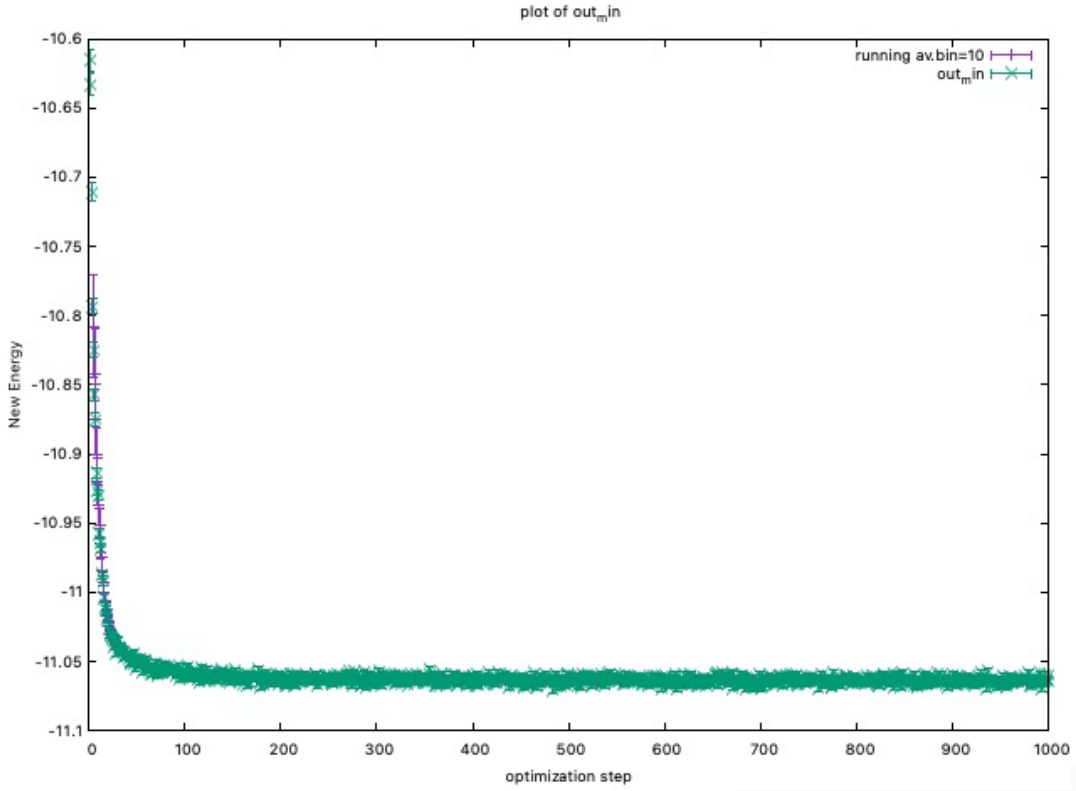
Optimization criteria

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.

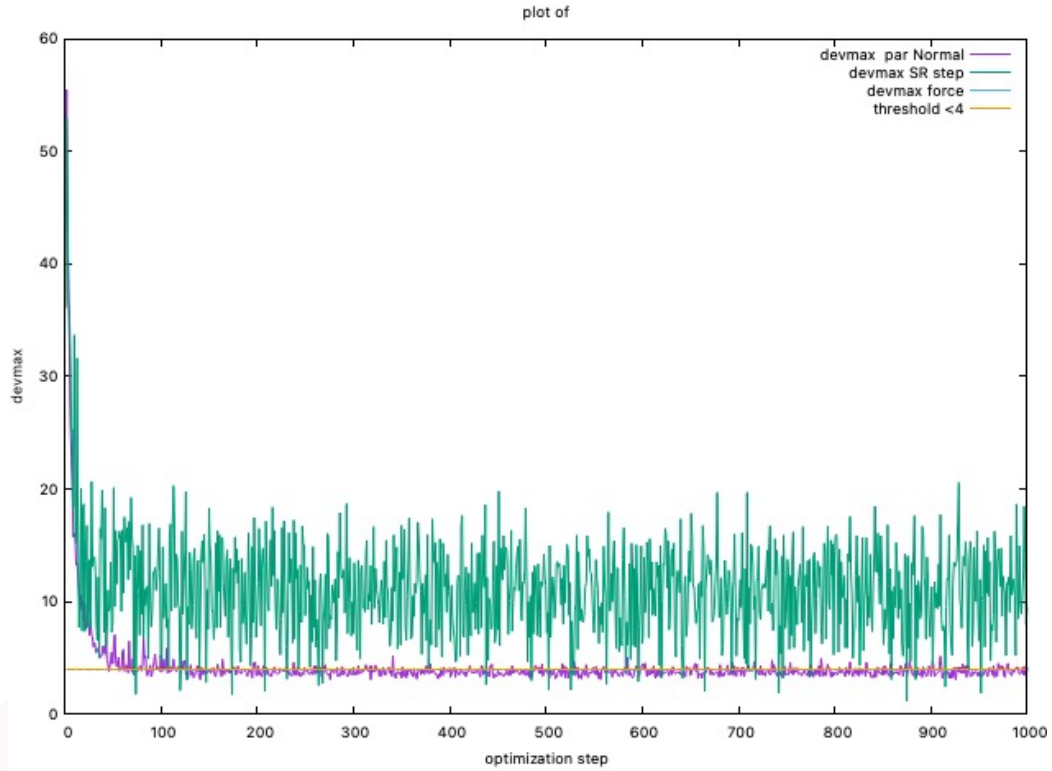
The definition of ``devmax`` is: $devmax \equiv \max_k \left(\left| \frac{f_k}{\sigma_{f_k}} \right| \right)$

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 (TurboRVB)

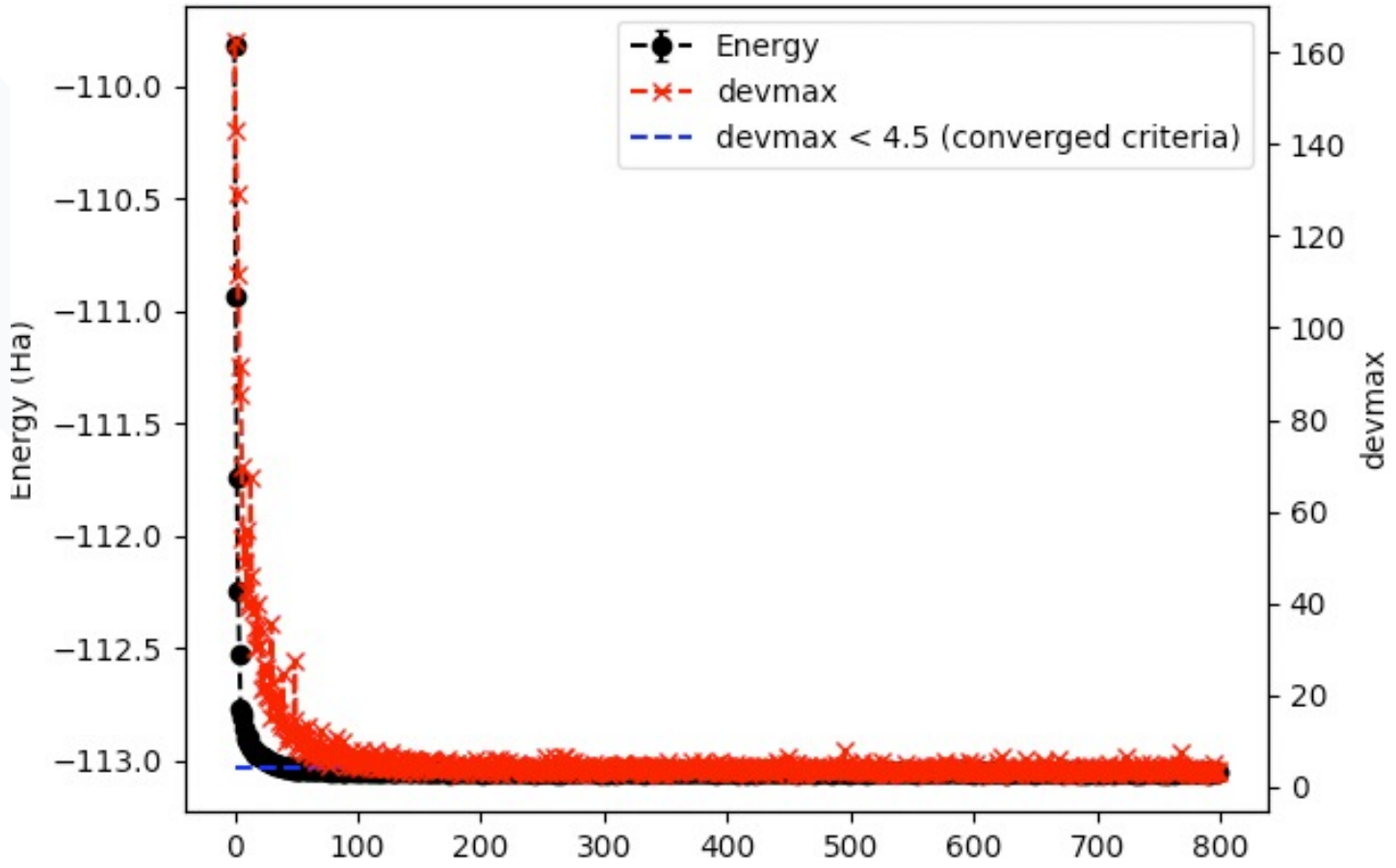


plot_Energy.sh



plot_devmax.sh

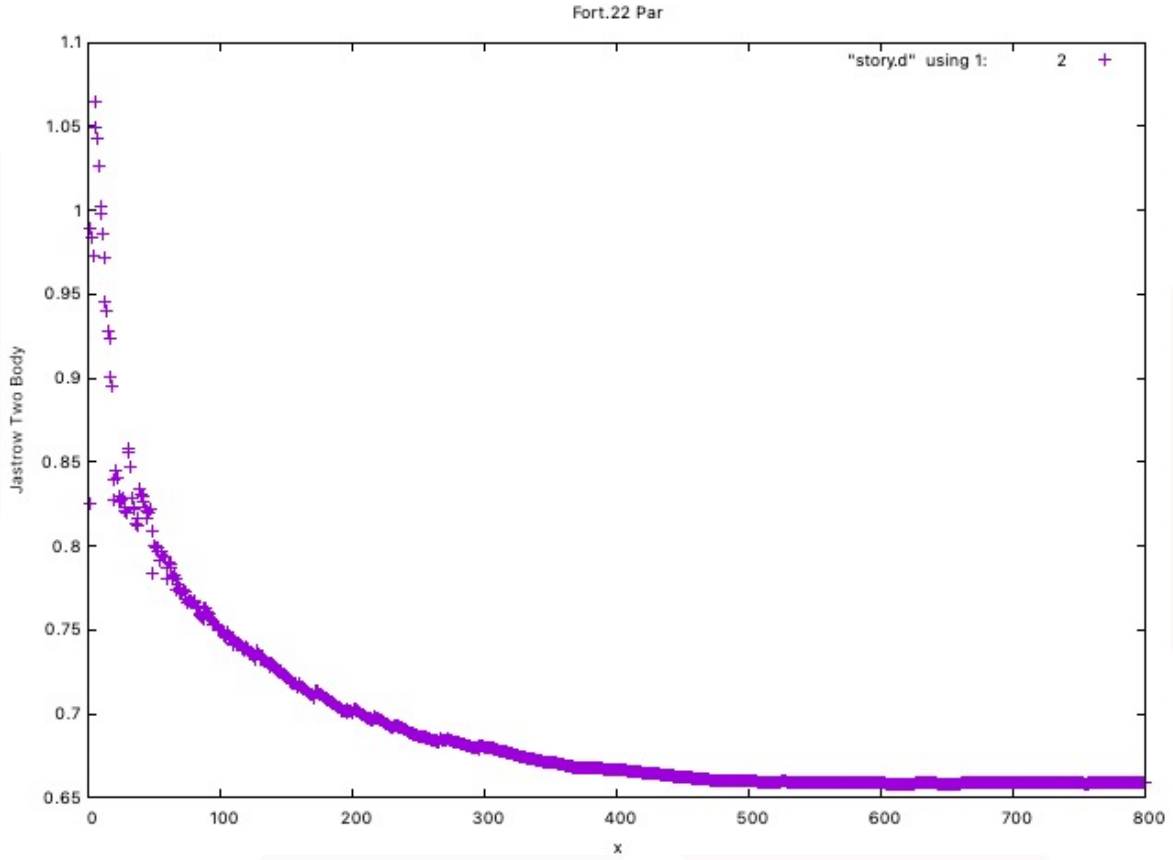
Plotting Graphs (Turbo-Genius)



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

Averaging parameters (TurboRVB)

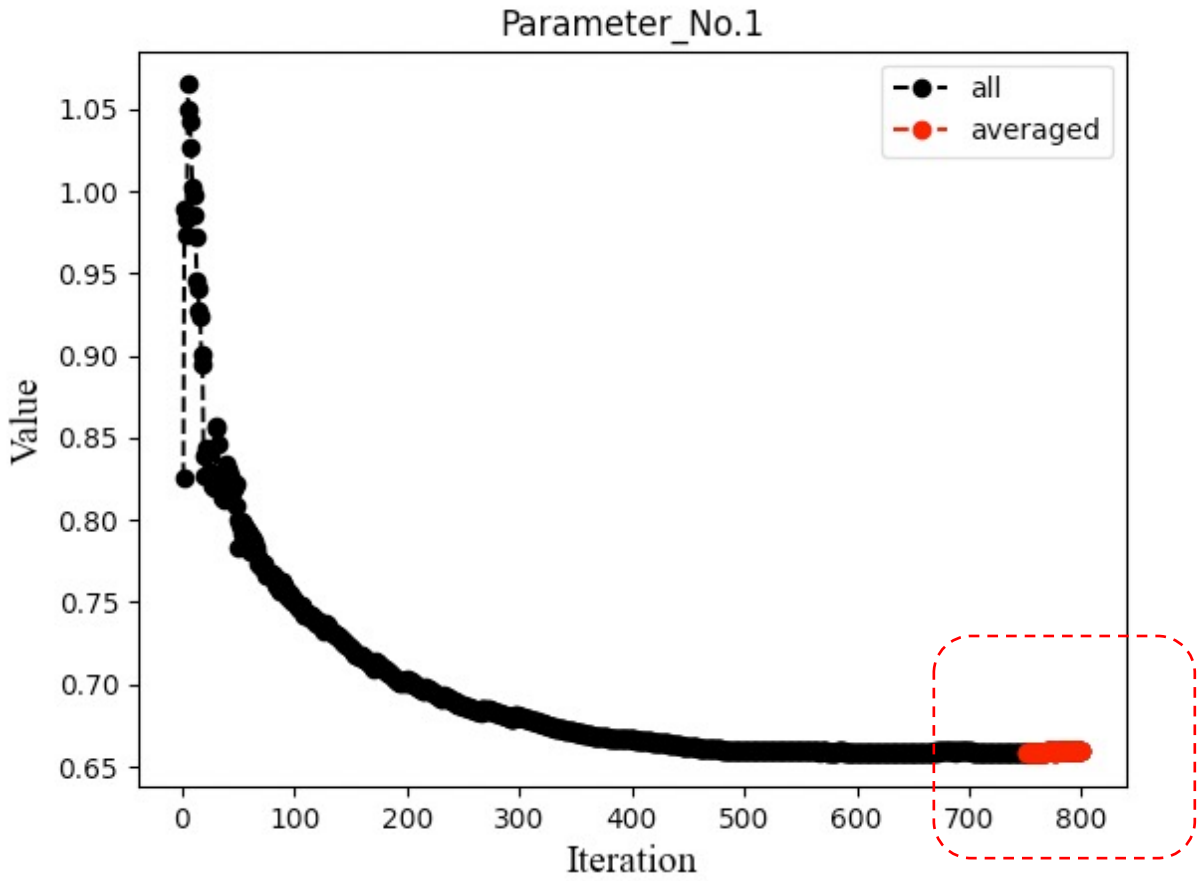
readalles.x enables us to visualize and average the latest several optimization steps.




```
% cat read.in
1 1 0 1
0
10
```

```
% readalles.x < read.in
```

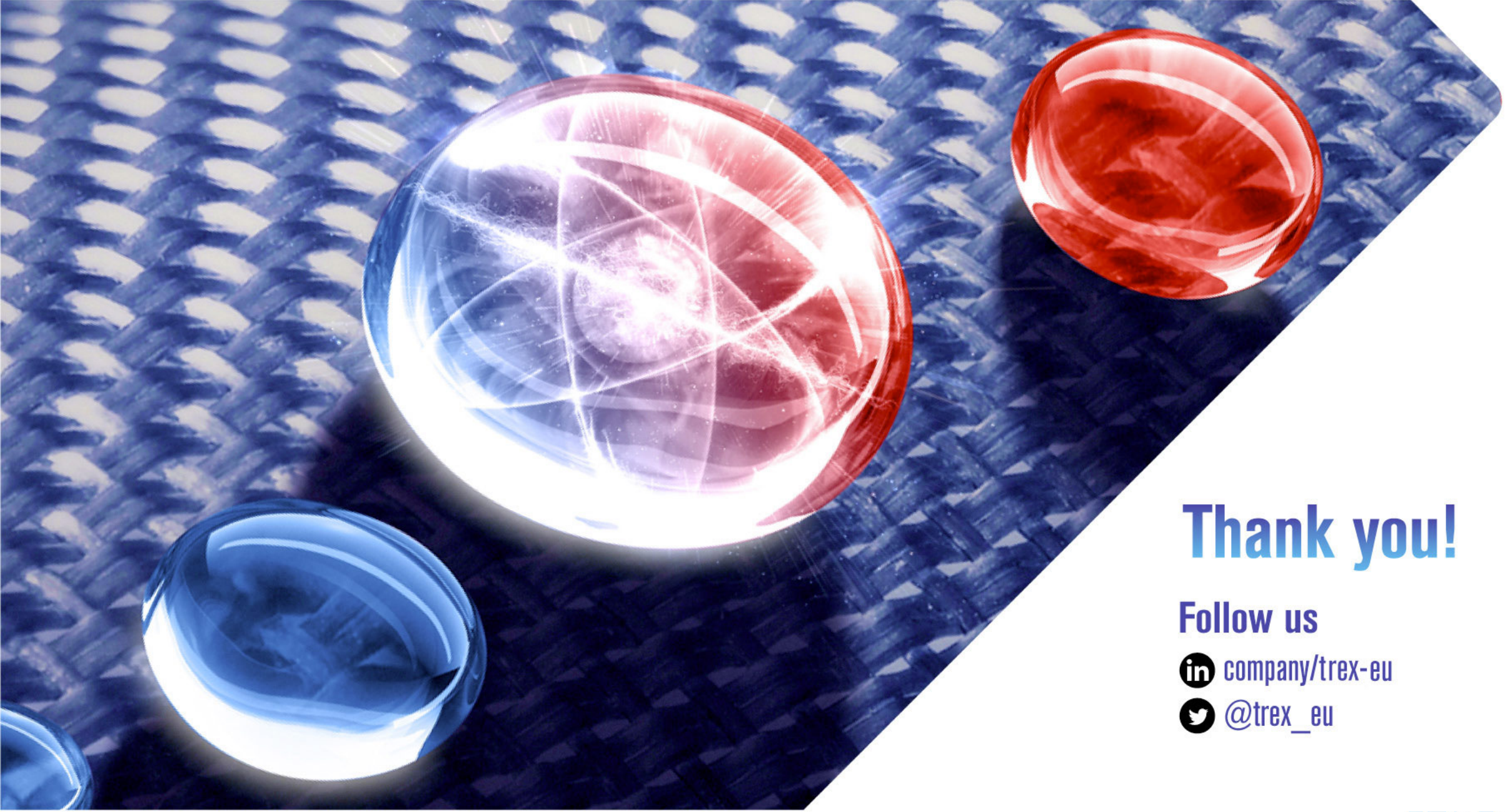
Averaging parameters (Turbo-Genius)



Averaged!

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

		Ansatz Level																
		JDFT				JsAGPs				JAGPu					JAGP(JPf)			
1. H_2-dimer	H_2	01 DFT Day1	02 VMCo Day2,3	03 VMC Day3	04 DMC Day4	06 Conv. Day2	08 VMCo Day2,3	09 VMC Day3	10 DMC Day4	NA					NA			
2. C_2-dimer	C_2	01-01 01-05 DFT Day1	01-06 VMCo Day2,3	01-07 VMC Day3	01-08 DMC Day4	02-01 Conv. Day2	02-02 VMCo Day2,3	02-03 VMC Day3	02-04 DMC Day4	03-01 03-03 DFT Day1	03-04 Conv. Day2	03-05 VMCo Day2,3	03-06 VMC Day3	03-06 DMC Day4	04-01 Conv. Day3	04-03 VMCo Day4	04-03 VMC Day4	04-03 DMC Day4
	C									03-01 DFT Day1					04-02 Conv. Day3			
3. H_2 on graphene	H_2 + Graphene	01 DFT Day1	02 VMCo Day2,3	03 VMC Day3	04 DMC Day4	NA				NA					NA			
	Graphene	05 DFT Day1	06 VMCo Day2,3	07 VMC Day3	08 DMC Day4													
	H_2	09 DFT Day1	10 VMCo Day2, 3	11 VMCo Day3	12 DMC Day4													



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