Software strategy and goals of TREX CoE

Targeting Real chemical accuracy at the EXascale

Claudia Filippi

MESA+ Institute for Nanotechnology, Universiteit Twente, The Netherlands

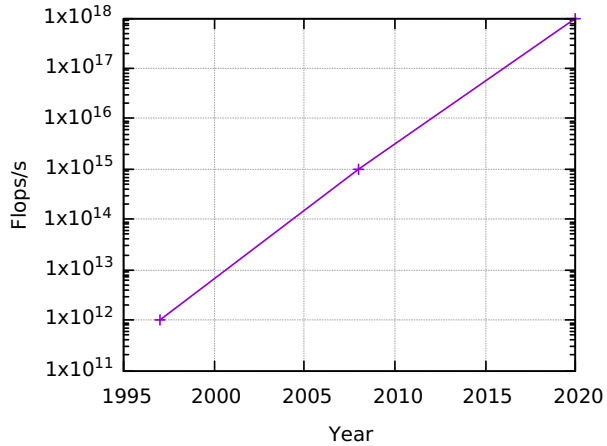
10th ABINIT International Developer Workshop, May 31-June 4, 2021

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.**

The race to the next supercomputer (1)

Worldwide competition for faster supercomputers

- 1997 : Teraflops/s[1]
- 2008 : Petaflops/s
- 2020? : Exaflops/s



– So far, exponential increase of computational power

– Moore's Law is ending

– Technological search for alternatives (quantum computing?)

[1] flops/s: floating point operations per second

The race to the next supercomputer (2)

- 1997 : Terascale : Distributed parallelism
- 2008 : Petascale : Multi-core chips or accelerators
- 2020? : Exascale : Hybrid architectures

Transition to exascale will be painful

– Memory per core decreases

– Network becomes slow versus computation

– Heterogeneous machines (accelerators)

Very few applications will scale

→ run high throughput computing (HTC) workloads

TREX CoE: Targeting Real chemical accuracy at the EXascale

TREX CoE: Targeting Real chemical accuracy at the EXascale

- Started in October 2020
- Focus → quantum Monte Carlo (QMC) methods
  - Massively parallelisable method: multiple QMC "trajectories"
  - Small I/O and memory, (often) little communication
  - Difficulty: take advantage of parallelism withing a trajectory
- Objective → make codes ready for exascale systems
- How → provide libraries instead of re-writing codes!
  - One library for high-performance QMC (QMCkl)
  - One library for exchanging info between codes (TREXIO)

TREX CoE: Targeting Real chemical accuracy at the EXascale

Scientists in quantum chemistry, physics, and machine learning

+ Software and HPC experts + Tech and communication SMEs

+ Representative of user communities

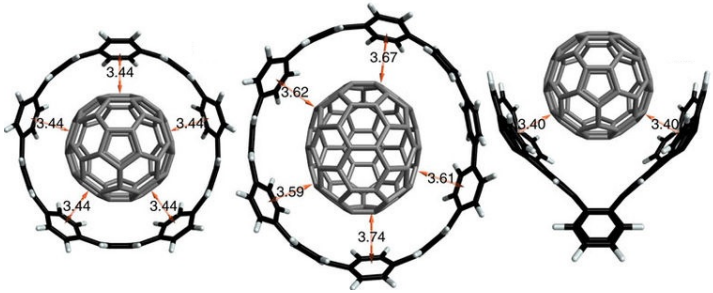
Few words about real-space quantum Monte Carlo methods

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)

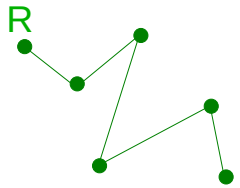
Simplest flavor: Variational Monte Carlo

Quantum observables $\rightarrow$ expectations values $\rightarrow$ integrals

Use Monte Carlo to compute expectation values

$$E = \frac{\langle\Psi|\mathcal{H}|\Psi\rangle}{\langle\Psi|\Psi\rangle} = \int \mathrm{d}\mathbf{R}^{3\mathrm{N}} \, \frac{\mathcal{H}\Psi(\mathbf{R})}{\Psi(\mathbf{R})} \, \boxed{\frac{|\Psi(\mathbf{R})|^2}{\int \mathrm{d}\mathbf{R}^{3\mathrm{N}}|\Psi(\mathbf{R})|^2}}$$

$$\approx \frac{1}{M}\sum_{i=1}^{M} E_L(\mathbf{R}_i) \quad \longleftarrow \text{ Sampled by Metropolis}$$



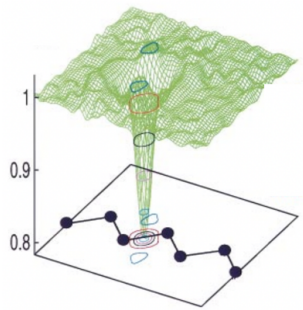
Random walk in 3N dimensions, $\mathbf{R} = (\mathbf{r}_1, \ldots, \mathbf{r}_N)$

Just a trick to compute integrals in many dimensions

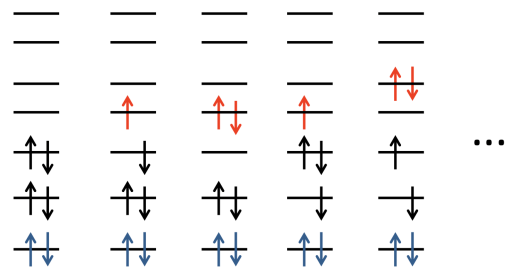
Key role of many-body wave function

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

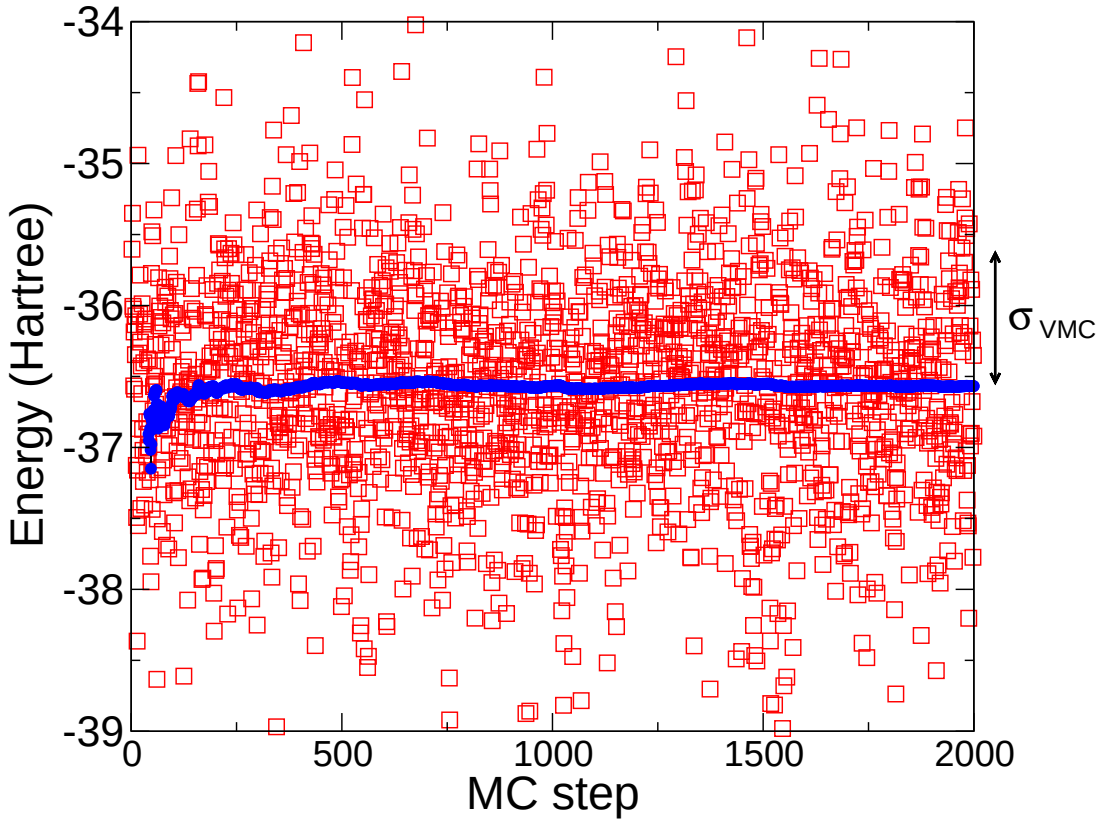
– Explicit dependence on electron-electron distances r_{ij}

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

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

Typical variational Monte Carlo run

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



$E = \langle E_L(\mathbf{R}) \rangle = -36.542 \pm 0.001$ Hartree (40×20000 steps)

$\sigma^2_{\mathrm{E}} = \langle (E_L(\mathbf{R}) - E)^2 \rangle = 0.90$ Hartree

Beyond variational Monte Carlo

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

Projector Monte Carlo method

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

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

- ▷ Apply operator to initial Ψ

$$\Psi_0 = \lim_{\tau\to\infty} e^{-\tau(\mathcal{H}-E_{\mathrm{T}})}\Psi$$

  if we choose $E_{\mathrm{T}} \approx E_0$

How do we perform the projection stochastically?

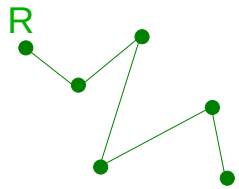
$$\Psi_0 = \lim_{\tau \to \infty} e^{-\tau(\mathcal{H} - E_T)} \Psi$$

Rewrite projection equation in integral form

$$\Psi(\mathbf{R}', t + \tau) = \int \mathrm{d}\mathbf{R} \, \langle \mathbf{R}' | e^{-\tau(\mathcal{H} - E_\mathrm{T})} | \mathbf{R} \rangle \, \Psi(\mathbf{R}, t)$$

Perform this integral by Monte Carlo integration

- ▷ Represent $\Psi(\mathbf{R}, t)$ as an ensemble of walkers
- ▷ Generate random walk by iterating integral equation

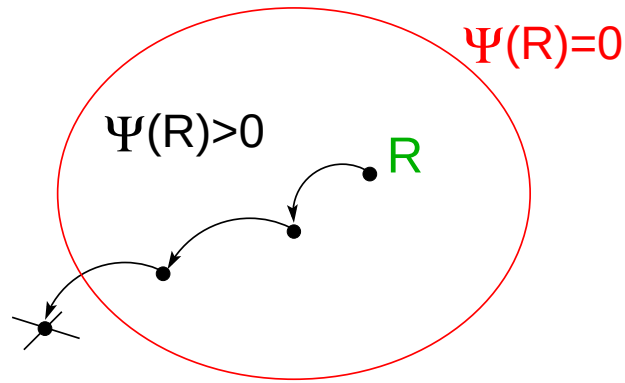


Note: Projection with other basis, e.g. determinants → FCIQMC

Diffusion Monte Carlo and the fermionic sign problem

Ψ is positive/negative $\Rightarrow$ Ψ is not a probability distribution

$\rightarrow$ Fixed-node approximation



Find best solution with same nodes as trial wave function Ψ

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 G2 set

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

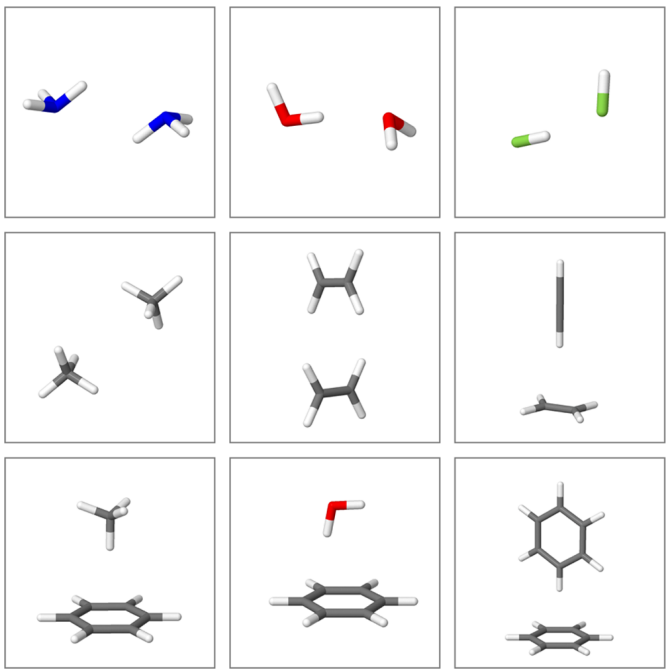
Petruzielo, Toulouse, Umrigar, J. Chem. Phys. **136**, 124116 (2012)

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



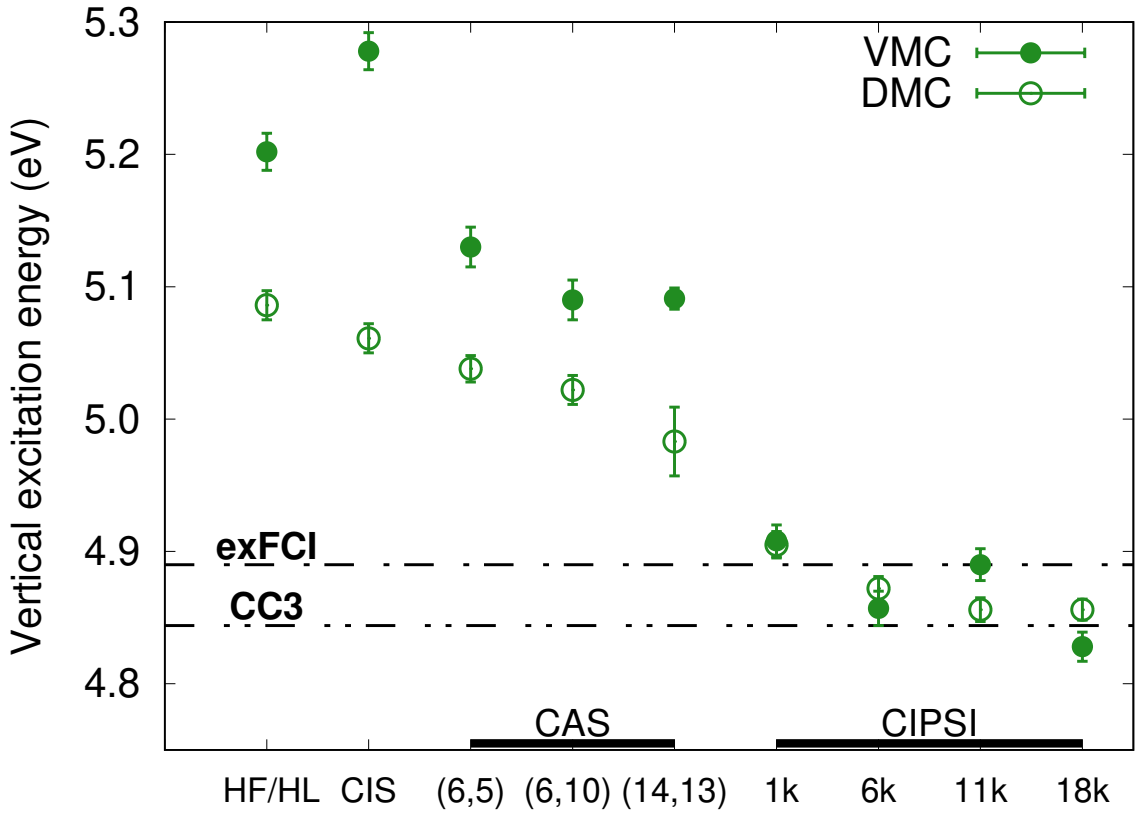
$\Delta_{\mathrm{MAD}} = 0.058$ kcal/mol

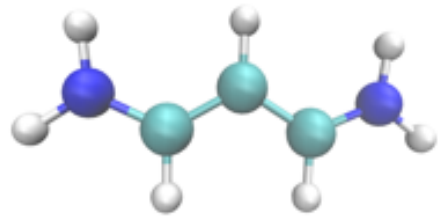
Dubecky *et al.*, JCTC **9**, 4287 (2013)

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

Diffusion Monte Carlo as a black-box approach?

Not really! 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!

Quantum Monte Carlo and exascale: why now?

Ease in paralellization of QMC is not sufficient for accurate results

$\rightarrow$ A big computer is not enough!

Recent methodological advances $\rightarrow$ new prospects

– Efficient computation of analytical energy derivatives

$\rightarrow$ QMC 'internally consistent" method

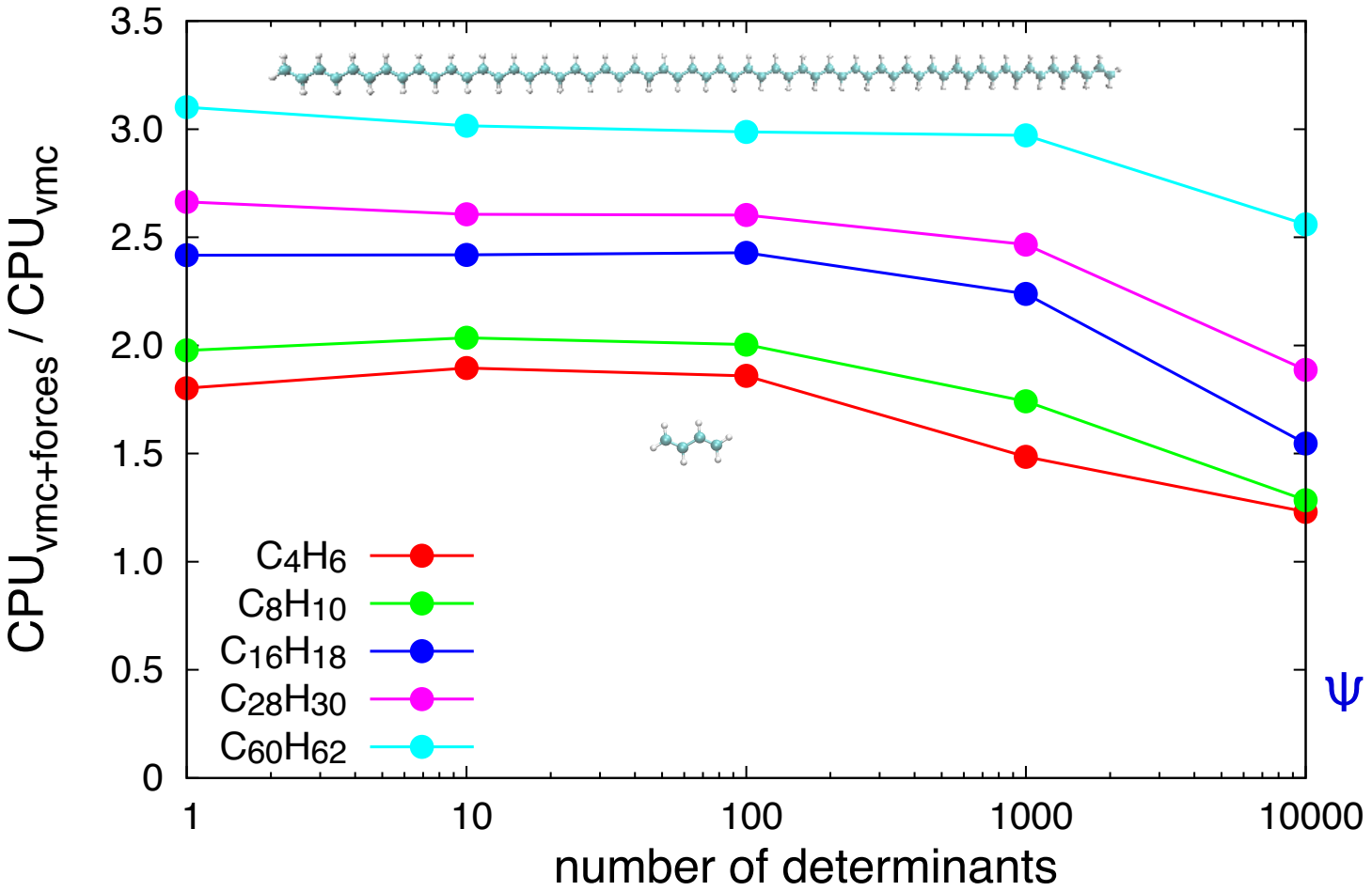
with geometries and wave functions determined in QMC

– Truly exploit freedom of choice of wave function $\Psi(\mathbf{r}_1, \ldots, \mathbf{r}_N)$

$\rightarrow$ development of new functional forms (geminals, FermiNet ..)

Some examples: Efficient derivatives of the energy

Example: Polyenes C_nH_{n+2} → from C_4H_6 to $C_{60}H_{62}$

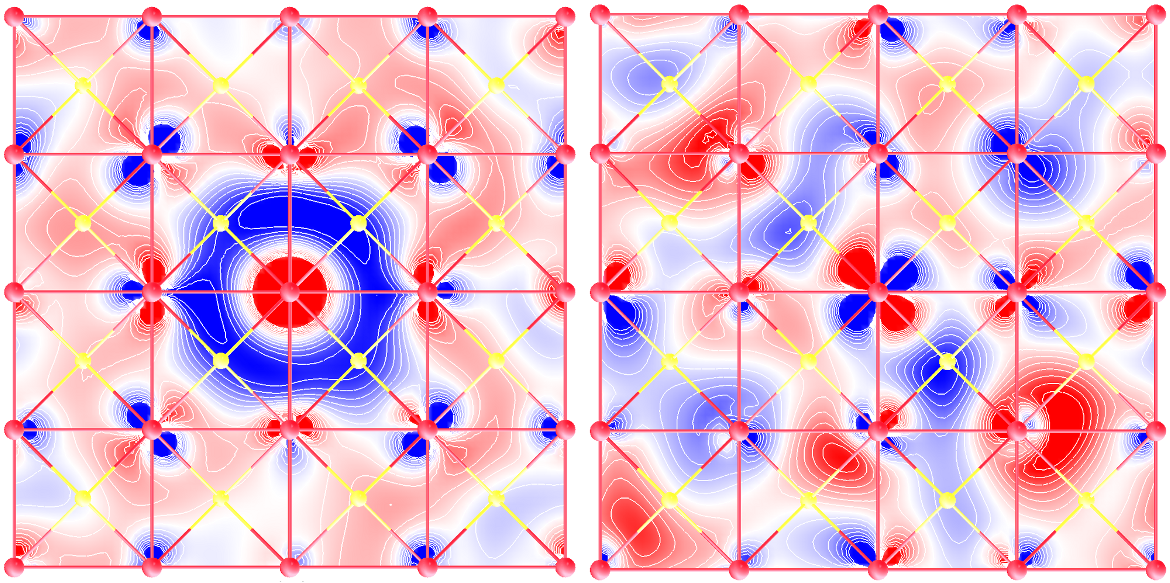


$$\Psi = \mathcal{J} \boxed{\sum c_i D_i}$$

Assaraf, Moroni, Filippi, JCTC **13**, 5273 (2017)

Some examples: Strong correlation

Predicting shape of electronic pairing in FeSe



Fully-optimized QMC wave function projected on symmetry

$\rightarrow$ s-wave or d-wave character is output NOT input!

Casula and Sorella, PRB **88**, 155125 (2013)

TREX software model for quantum Monte Carlo

- Software model → not a monolitic code!
- HPC platform of interoperable codes/libraries

Key steps to build TREX platform:

1) Create QMCkl library of QMC kernels
   → Easy integration in TREX/external (non)stochastic codes
2) Develop common TREXIO library
   → Easy use of TREX and other codes in a pipeline
3) Refactor and modularize TREX codes to use libraries
4) Integration in AiiDA for workflow managment/HTC
   → Easily exploit integrated machine learning (QML) tools

TREX codes

- Real-space quantum Monte Carlo

  CHAMP
  QMC=Chem
  TurboRVB

- Full configuration interaction QMC

  NECI

- Deterministic quantum chemical codes

  Quantum Package
  GammCor

- Machine learning

  Quantum Machine Learning (QML) package

QMCkl library (1)

Our concern: Ensure that future progress in QMC continues

- This requires codes with new algorithms
- New algorithms implemented by physicists/chemists
  with different programming language knowledge/preference

. . . but programming for the exascale horribly complex

How do we reconcile these two aspects?

QMCkl library: two implementations (2)

Our solution:

- Implement kernels in a Fortran **human-readable library**

  $\rightarrow$ task of QMC experts
- Translate the human-readable library in a C **HPC-library**

  $\rightarrow$ task of HPC experts
- Scientists can link either library with their codes

Benefits:

- Codes remain understandable/controllable
- Do not die with change of architecture
- Separation of concerns

Human-readable QMCkl

Some more words:

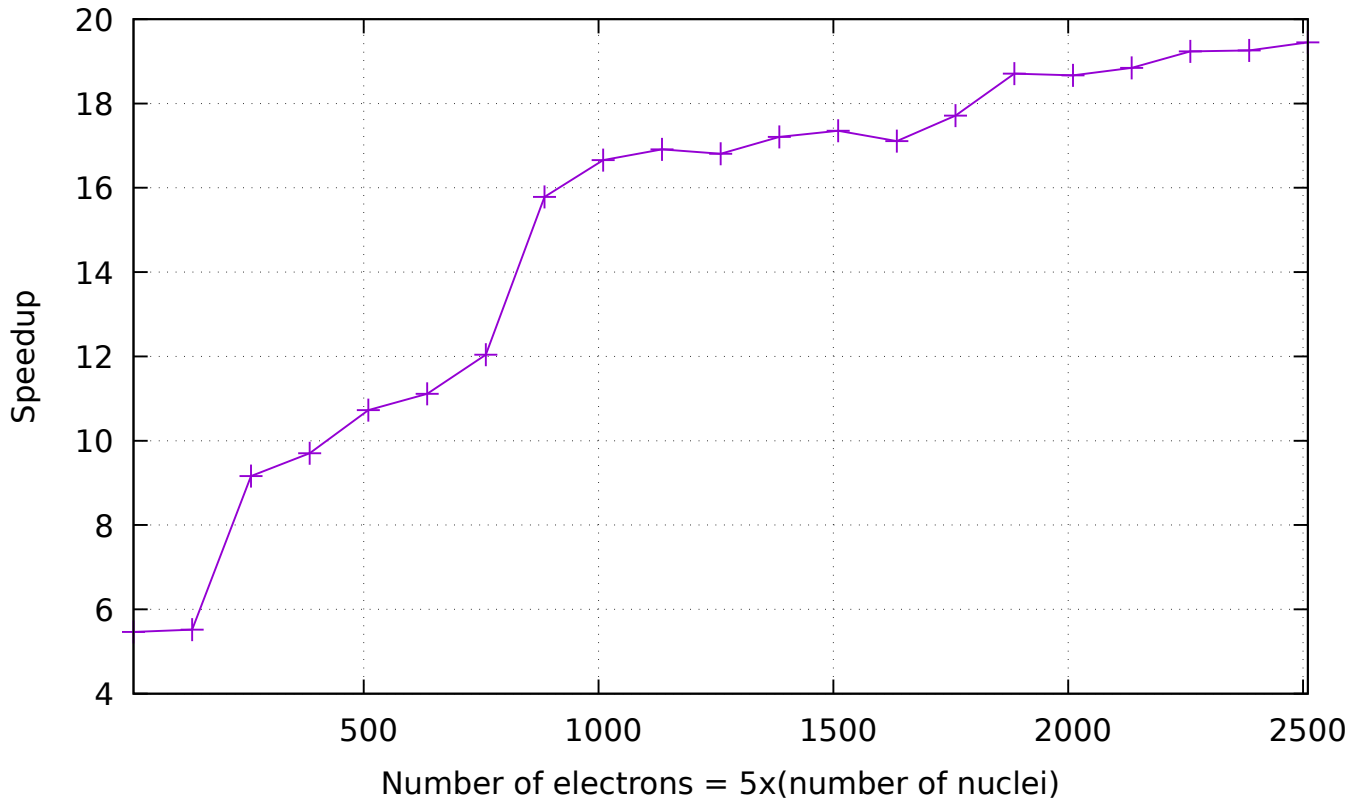
- Computational kernels in Fortran for readability
- The API is C-compatible: QMCkl appears like a C library

  $\rightarrow$ can be used in all other languages

  $\rightarrow$ same API as HPC-library
- System functions in C (memory allocation, *etc*)
- A lot of documentation

Our first application: Jastrow factor

Construction of kernel for key element in $\Psi = \boxed{\mathcal{J}} \sum_i D_i$

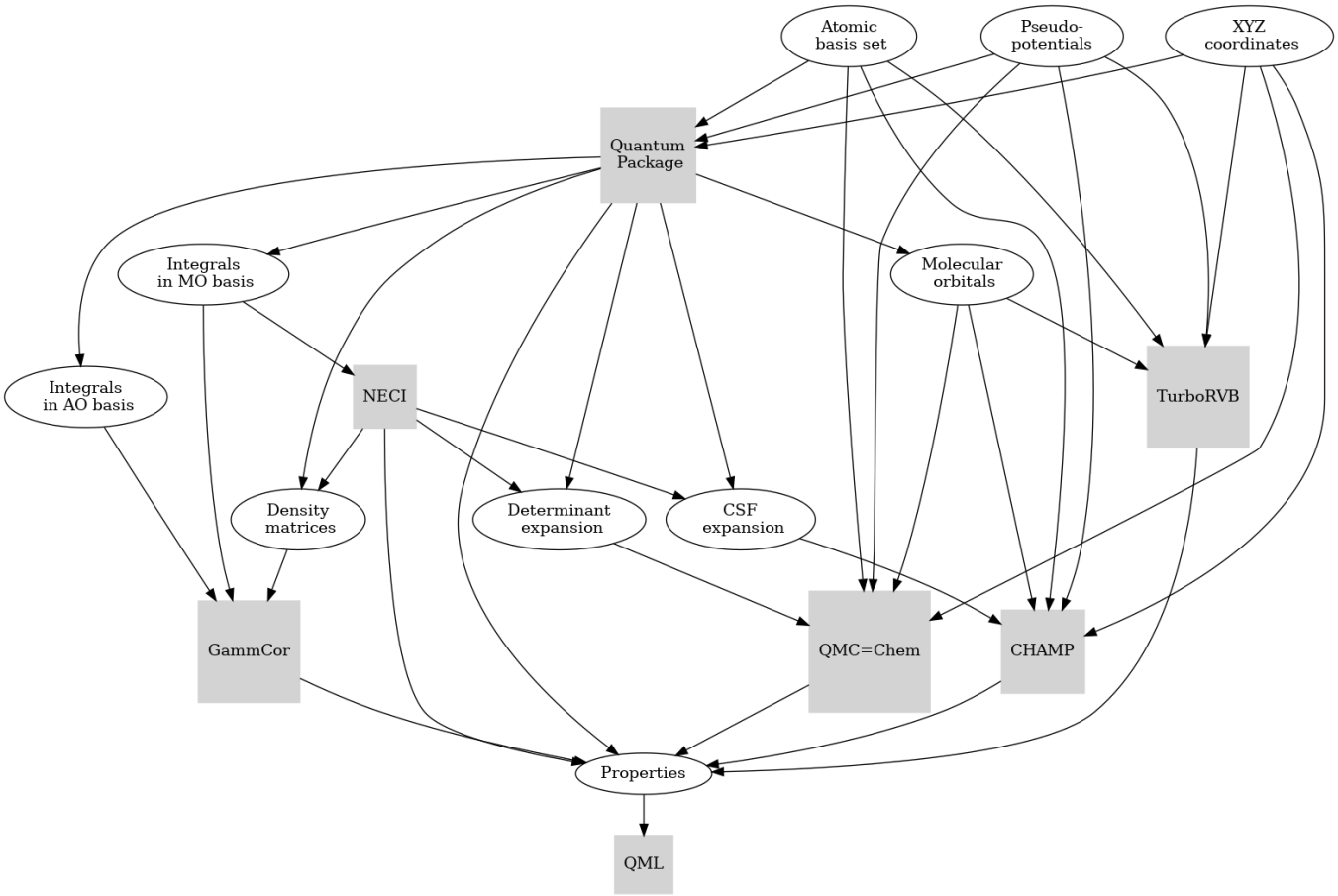


About 80% of the AVX-512 peak is reached on a Skylake CPU

$\rightarrow$ currently working on GPU kernel

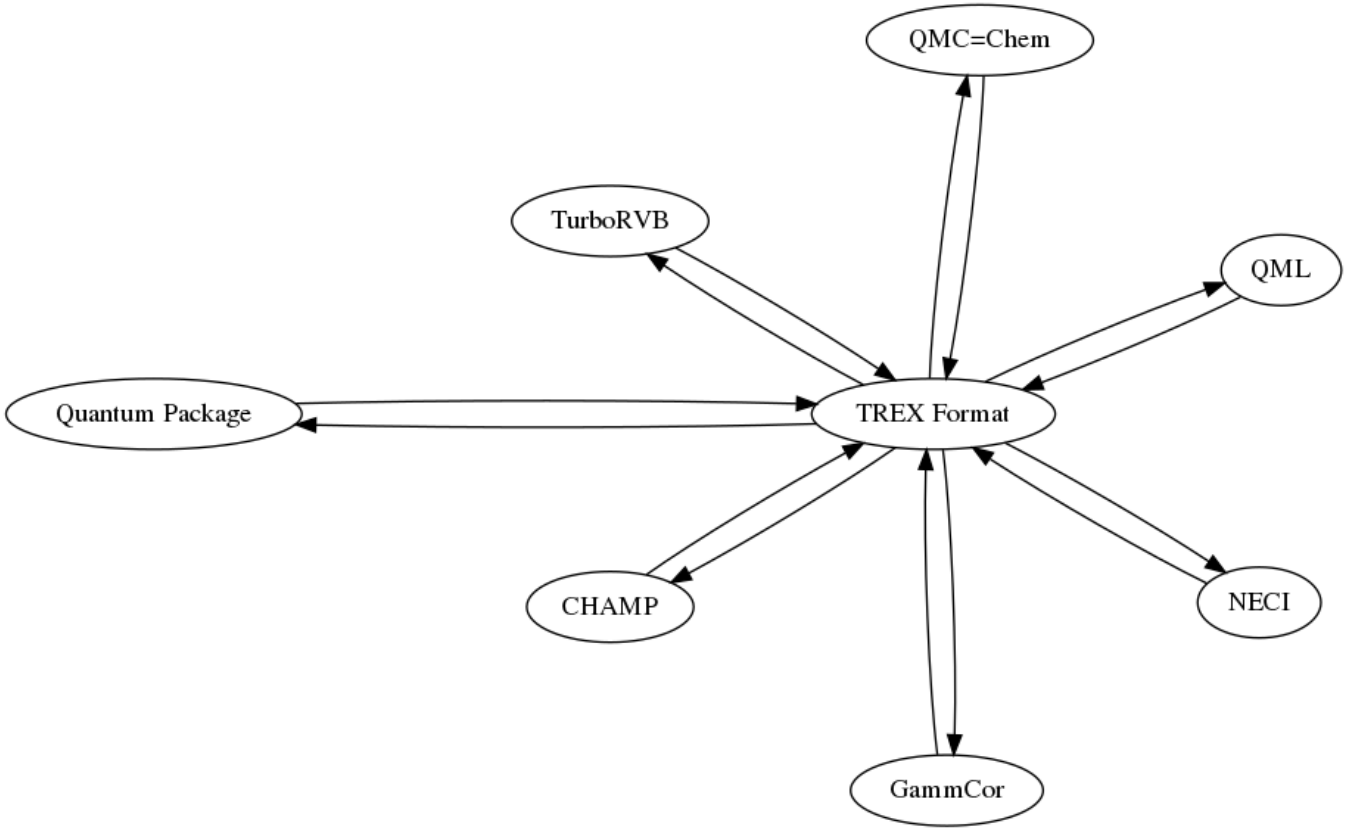
The TREX I/O library

Current situation



The TREX I/O library

Final goal



TREXIO: main features

Front-end

- Definition of an API to read/write wave functions
- C-compatible API: Easy bindings in other languages

Back-end

- HDF5: Efficient I/O
- Text: debugging, fallback when HDF5 cannot be installed

Links

- TREX web site : `https://trex-coe.eu`
- QMCkl documentation : `https://trex-coe.github.io/qmckl`
- QMCkl repository : `https://github.com/trex-coe/qmckl`
- TREXIO repository : `https://github.com/trex-coe/trexio`

In conclusion . . . stay tuned on TREX developments

Thank you for your attention!