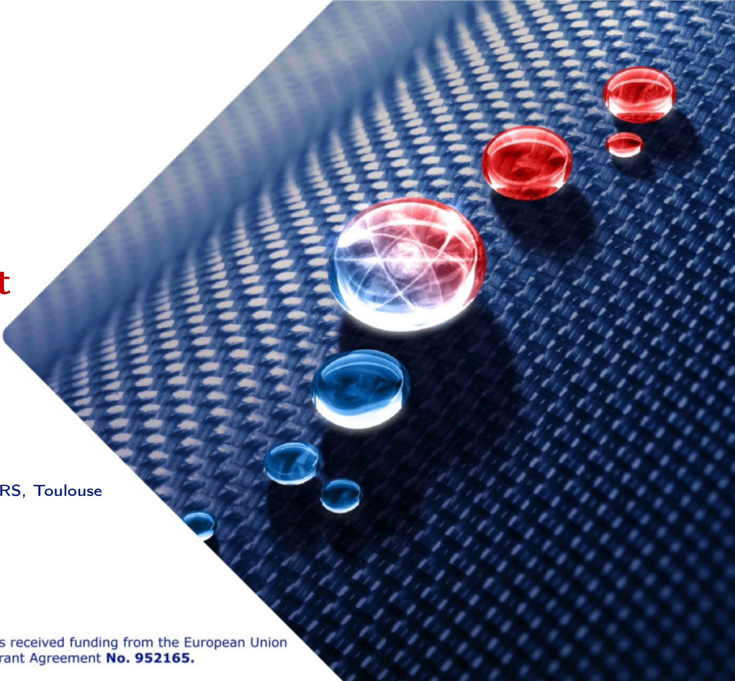
Targeting Real chemical accuracy at the EXascale

The TREXIO file format and library

Anthony Scemama, Evgeny Posenitskiy

8/06/2022

Lab. Chimie et Physique Quantiques, FERMI, UPS/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.**

TREX

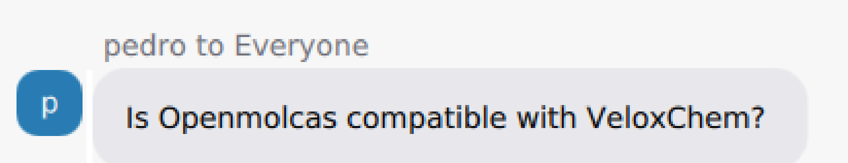
European Center of Excellence for exascale computing

- Objective: Help quantum chemistry benefit from Exascale systems
- Quantum chemists, Computer scientists, Supercomputing centers, SMEs

TREX Software

- Quantum Monte Carlo (QMC): QMC=Chem (AS), TurboRVB (S. Sorella), CHAMP (C. Filippi), QMC kernel library (TREX)
- FCIQMC: NECI (A. Alavi)
- Selected CI (CIPSI): Quantum Package (AS)
- SAPT: GammCor (K. Pernal)

This morning in the chat



pedro to Everyone

Is Openmolcas compatible with VeloxChem?

This talk

How to make more science with multiple codes with little effort

Composition

- A program is a function $p : \text{input} \longrightarrow \text{output}$
- If the output of a program p_1 is of the same type as the input of a program p_2, we can define a new program $p_3 = p_2 \circ p_1$:

$$\begin{aligned} p_1 &: t_1 \to t_2 \\ p_2 &: t_2 \to t_3 \\ p_2 \circ p_1 &: t_1 \to t_3 \end{aligned}$$

Unix philosophy

Douglas McIlroy (1978)

1. Make each program do one thing well. To do a new job, build afresh rather than complicate old programs by adding new "features".
2. Expect the output of every program to become the input to another, as yet unknown, program. Don't clutter output with extraneous information. Avoid stringently columnar or binary input formats. Don't insist on interactive input.

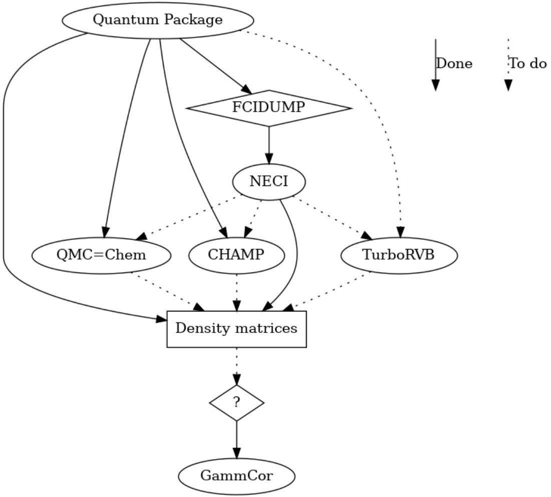
Examples

- `git log -1 | head -1 | cut -d ' ' -f 2`
- 45 years later, the unix pipe is still widely used!
- Monolythic codes are contrary to these principles

Input/Output

- Quantum Package (AS): CIPSI
  - Input: `xyz` coordinates, AOs, a wave function
  - Output: MOs, CI expansion, 1e/2e Integrals, 1e/2e Density matrices
- NECI (A. Alavi): FCIQMC
  - Input: 1e/2e Integrals in MO basis
  - Output: CI expansion, 1e/2e Density matrices
- QMC=Chem (AS), TurboRVB (S. Sorella), CHAMP (C. Filippi): QMC
  - Input: A wave function
  - Output: A wave function
- GammCor (K. Pernal): SAPT
  - Input: 1e/2e Integrals, 1e/2e Density matrices

Situation two years ago



Composition

Question

How can we build easily new programs by composition?

Answer

Making codes have the same signature with same type for input and output:

$$code : t \longrightarrow t$$

- By modifying the code
- Or by composition: $c_2 \circ \text{code} \circ c_1 : t \longrightarrow \text{input} \longrightarrow \text{output} \longrightarrow t$

- With Unix programs, the type t is a string.
- The string type is too primitive for Ψ, so we need to define a common file format.

File formats

- In Unix philosophy, text files are recommended:
  - portability (architecture independent)
  - can be read as a stream
  - readable in any language
  - no conversion required

If the common format is `text`, the programs can be composed with all unix tools (`grep`, `cut`, `head`, `tail`, text editors, . . .)

- Problems with text files
  - Large storage size (archiving), but can be compressed
  - Expensive conversion from ASCII to binary representation
  - $\Longrightarrow$ Poor I/O performance, bad for HPC

Proposed compromise

TREXIO: Domain-specific I/O Library

- Very permissive license (BSD-3-clause)
- Domain-specific: *Do one thing and do it well* ⟶ wave functions
- Portable (C API): usable with any language
- Single front-end, multiple back-ends
- Text back-end: multiple text files
- HDF5 back-end: single binary file

Advantages

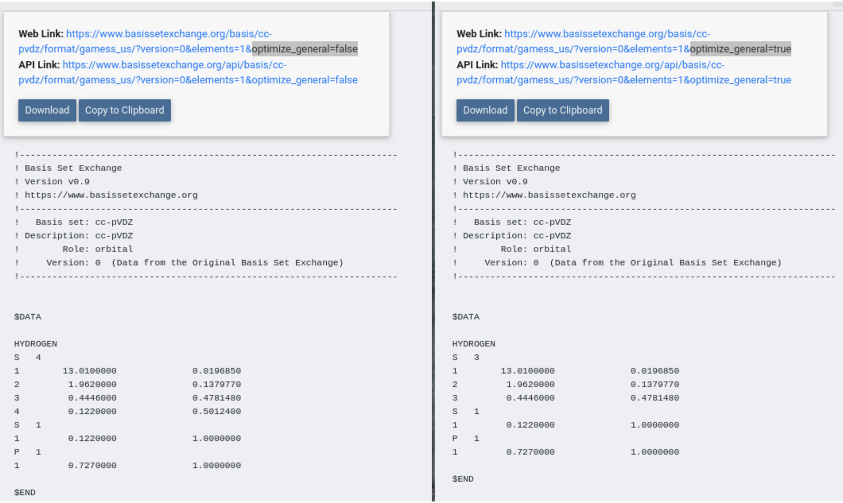
- Binary files if wanted (performance, small files)
- Text files if wanted (unix tools, git repositories, etc)
- HDF5 binary files are portable (endianness)
- If HDF5 is de-activated at compile time: zero dependency (pure C code)

Data stored with TREXIO

- No external knowledge is needed to compute $\Psi(r_1, \ldots, r_N)$:
    - Ψ is just a mathematical function defined by parameters.
    - We define a general form for Ψ, and we want to be able to read/write its parameters.
    - All the needed numbers are stored in the file: no external database or integral computation required.

Disambiguation

"cc-pVDZ" is not enough information:



Web Link: https://www.basissetexchange.org/basis/cc-pvdz/format/gamess_us/?version=0&elements=1&optimize_general=false
API Link: https://www.basissetexchange.org/api/basis/cc-pvdz/format/gamess_us/?version=0&elements=1&optimize_general=false

```
!----------------------------------------------------------------------
! Basis Set Exchange
! Version v0.9
! https://www.basissetexchange.org
!----------------------------------------------------------------------
!   Basis set: cc-pVDZ
! Description: cc-pVDZ
!        Role: orbital
!     Version: 0  (Data from the Original Basis Set Exchange)
!----------------------------------------------------------------------


$DATA

HYDROGEN
S   4
1        13.0100000              0.0196850
2         1.9620000              0.1379770
3         0.4446000              0.4781480
4         0.1220000              0.5012400
S   1
1         0.1220000              1.0000000
P   1
1         0.7270000              1.0000000

$END
```

Web Link: https://www.basissetexchange.org/basis/cc-pvdz/format/gamess_us/?version=0&elements=1&optimize_general=true
API Link: https://www.basissetexchange.org/api/basis/cc-pvdz/format/gamess_us/?version=0&elements=1&optimize_general=true

```
!----------------------------------------------------------------------
! Basis Set Exchange
! Version v0.9
! https://www.basissetexchange.org
!----------------------------------------------------------------------
!   Basis set: cc-pVDZ
! Description: cc-pVDZ
!        Role: orbital
!     Version: 0  (Data from the Original Basis Set Exchange)
!----------------------------------------------------------------------


$DATA

HYDROGEN
S   3
1        13.0100000              0.0196850
2         1.9620000              0.1379770
3         0.4446000              0.4781480
S   1
1         0.1220000              1.0000000
P   1
1         0.7270000              1.0000000

$END
```

Disambiguation

Example: Different AO conventions

- Ordering of the AOs:
  - $d_{-2}, d_{-1}, d_0, \ldots$ or $d_0, d_{+1}, d_{-1}, \ldots$?
  - $d_{x^2}, d_{y^2}, d_{z^2}, \ldots$?

$\Longrightarrow$ The order is fixed and given in the documentation.

- Are the AOs assumed normalized?
- Should d_{xy} have the same normalization coefficient as d_{z^2}?

$$\iiint \left(x\,y\,G(x,y,z)\right)^2 dx\,dy\,dz \neq \iiint \left(z^2\,G(x,y,z)\right)^2 dx\,dy\,dz$$

$\Longrightarrow$ All the normalization coefficients are stored in the file.

Data stored with TREXIO

- Hierarchical data layout:
  - Groups

`Metadata`	`Electron`	`Nucleus`	`ECP`	`Basis`
`AO`	`MO`	`Determinant`	`State`	`Cell`
`AO_1e_int`	`MO_1e_int`	`RDM`	`PBC`	`QMC`
`AO_2e_int`	`MO_2e_int`			

  - Inside each group, multiple values

Example: `Nucleus` group

Variable	Type	Dimensions	Description
`num`	`dim`		Number of nuclei
`charge`	`float`	`(nucleus.num)`	Charges of the nuclei
`coord`	`float`	`(3,nucleus.num)`	Coordinates of the atoms
`label`	`str`	`(nucleus.num)`	Atom labels
`point_group`	`str`		Symmetry point group
`repulsion`	`float`		Nuclear repulsion energy

API

- Computable function names (32- or 64-bit variants):
  `trexio_<read|write|has>[_safe]_<group>_<data>[_32|_64]`
  `(trexio_file, data[, size])`
- Safe API: Takes max dimension as arguments for memory safety
- Return code for error handling

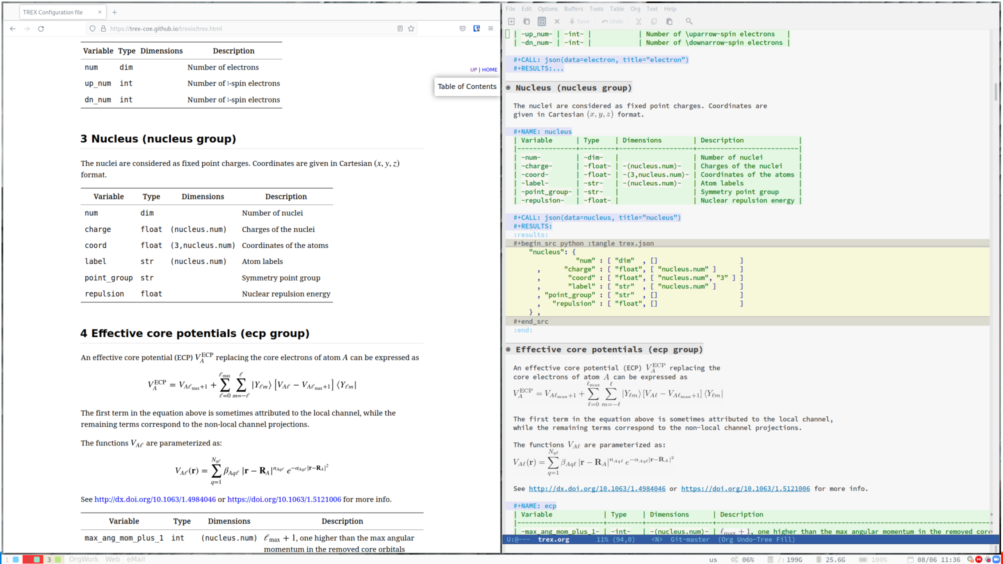
```c
#define MAX_SIZE 10
double charge_array[MAX_SIZE];
trexio_exit_code rc;

rc = trexio_read_nucleus_charge(trexio_file, charge_array);

rc = trexio_read_safe_nucleus_charge_64(trexio_file, charge_array, MAX_SIZE);
```

- Usable in C, C++, Fortran, Python, [Bash, OCaml]
- Auto-generated
- Literate programming with Org-mode: easy to extend

org-mode


TREX Configuration file
https://trex-coe.github.io/trexio/trex.html
UP | HOME
Table of Contents
File Edit Options Buffers Tools Table Org Text Help
Save
Undo
1 3 OrgWork Web eMail
us 06% /: 199G 25.6G 100% 08/06 11:36

Variable	Type	Dimensions	Description
num	dim		Number of electrons
up_num	int		Number of $\uparrow$-spin electrons
dn_num	int		Number of $\downarrow$-spin electrons

3 Nucleus (nucleus group)

The nuclei are considered as fixed point charges. Coordinates are given in Cartesian (x, y, z) format.

Variable	Type	Dimensions	Description
num	dim		Number of nuclei
charge	float	(nucleus.num)	Charges of the nuclei
coord	float	(3,nucleus.num)	Coordinates of the atoms
label	str	(nucleus.num)	Atom labels
point_group	str		Symmetry point group
repulsion	float		Nuclear repulsion energy

4 Effective core potentials (ecp group)

An effective core potential (ECP) V_A^{ECP} replacing the core electrons of atom A can be expressed as

$$V_A^{\text{ECP}} = V_{A\ell_{\max}+1} + \sum_{\ell=0}^{\ell_{\max}} \sum_{m=-\ell}^{\ell} |Y_{\ell m}\rangle \left[V_{A\ell} - V_{A\ell_{\max}+1} \right] \langle Y_{\ell m}|$$

The first term in the equation above is sometimes attributed to the local channel, while the remaining terms correspond to the non-local channel projections.

The functions $V_{A\ell}$ are parameterized as:

$$V_{A\ell}(\mathbf{r}) = \sum_{q=1}^{N_{q\ell}} \beta_{Aq\ell} \, |\mathbf{r} - \mathbf{R}_A|^{n_{Aq\ell}} \, e^{-\alpha_{Aq\ell} |\mathbf{r}-\mathbf{R}_A|^2}$$

See http://dx.doi.org/10.1063/1.4984046 or https://doi.org/10.1063/1.5121006 for more info.

Variable	Type	Dimensions	Description
max_ang_mom_plus_1	int	(nucleus.num)	$\ell_{\max}+1$, one higher than the max angular momentum in the removed core orbitals

```
| ~up_num~ | ~int~ |              | Number of \uparrow-spin electrons   |
| ~dn_num~ | ~int~ |              | Number of \downarrow-spin electrons |

#+CALL: json(data=electron, title="electron")
#+RESULTS:...
```

⊙ Nucleus (nucleus group)

The nuclei are considered as fixed point charges. Coordinates are given in Cartesian (x, y, z) format.

```
#+NAME: nucleus
| Variable      | Type    | Dimensions        | Description              |
|---------------+---------+-------------------+--------------------------|
| ~num~         | ~dim~   |                   | Number of nuclei         |
| ~charge~      | ~float~ | ~(nucleus.num)~   | Charges of the nuclei    |
| ~coord~       | ~float~ | ~(3,nucleus.num)~ | Coordinates of the atoms |
| ~label~       | ~str~   | ~(nucleus.num)~   | Atom labels              |
| ~point_group~ | ~str~   |                   | Symmetry point group     |
| ~repulsion~   | ~float~ |                   | Nuclear repulsion energy |

#+CALL: json(data=nucleus, title="nucleus")
#+RESULTS:
:results:
#+begin_src python :tangle trex.json
    "nucleus": {
                "num" : [ "dim"  , []                      ]
      ,      "charge" : [ "float", [ "nucleus.num" ]       ]
      ,       "coord" : [ "float", [ "nucleus.num", "3" ] ]
      ,       "label" : [ "str"  , [ "nucleus.num" ]       ]
      , "point_group" : [ "str"  , []                      ]
      ,   "repulsion" : [ "float", []                      ]
    } ,
#+end_src
:end:
```

⊙ Effective core potentials (ecp group)

An effective core potential (ECP) V_A^{ECP} replacing the core electrons of atom A can be expressed as

$$V_A^{\text{ECP}} = V_{A\ell_{\max}+1} + \sum_{\ell=0}^{\ell_{\max}} \sum_{m=-\ell}^{\ell} |Y_{\ell m}\rangle \left[V_{A\ell} - V_{A\ell_{\max}+1} \right] \langle Y_{\ell m}|$$

The first term in the equation above is sometimes attributed to the local channel, while the remaining terms correspond to the non-local channel projections.

The functions $V_{A\ell}$ are parameterized as:

$$V_{A\ell}(\mathbf{r}) = \sum_{q=1}^{N_{q\ell}} \beta_{Aq\ell} \, |\mathbf{r} - \mathbf{R}_A|^{n_{Aq\ell}} \, e^{-\alpha_{Aq\ell} |\mathbf{r}-\mathbf{R}_A|^2}$$

See http://dx.doi.org/10.1063/1.4984046 or https://doi.org/10.1063/1.5121006 for more info.

```
#+NAME: ecp
| Variable             | Type    | Dimensions        | Description
|----------------------+---------+-------------------+----------------------------------------------------------
| ~max_ang_mom_plus_1~ | ~int~   | ~(nucleus.num)~   | $\ell_{\max}+1$, one higher than the max angular momentum in the removed core
```

```
U:@---  trex.org      11% (94,0)     <N>  Git-master  (Org Undo-Tree Fill)
```

Example (Fortran)

```fortran
use trexio                              ! ISO-C-binding module to be included with your code
double precision          :: charge(3)
integer                   :: n
integer(trexio_t)         :: f          ! File handle
integer(trexio_exit_code) :: rc         ! Return code

!  Write
f = trexio_open('water.h5', 'w', TREXIO_HDF5, rc)   ! rc -> TREXIO_SUCCESS
charge = (/ 8., 1., 1. /)
rc = trexio_write_nucleus_num(f, 3)
rc = trexio_write_nucleus_charge(f, charge)
rc = trexio_close(f)

!  Read
f  = trexio_open('water.h5', 'r', TREXIO_HDF5, rc)   ! rc -> TREXIO_SUCCESS
rc = trexio_read_nucleus_num(f, n)                  ! n = 3
charge(:) = 0.
rc = trexio_read_nucleus_charge(f, charge)          ! charge = (/ 1., 2., 3./)
rc = trexio_close(f)
```

Example (Python)

```python
import trexio

# Write
with trexio.File('water.h5', mode='w', back_end=trexio.TREXIO_HDF5) as f:
    charge = [ 8., 1., 1. ]
    trexio.write_nucleus_num(f,3)
    trexio.write_nucleus_charge(f,charge)

# Read
with trexio.File('water.h5', mode='r', back_end=trexio.TREXIO_HDF5) as f:
    n = trexio.read_nucleus_num(f)             # n = 3
    charge = trexio.read_nucleus_charge(f)     # charge = [1., 2., 3.]
```

Example (C)

```c
#include <trexio.h>
#include <assert.h>

trexio_exit_code rc;

/*  Write */
trexio_t* f = trexio_open("water.h5", 'w', TREXIO_HDF5, &rc);
double charge[] = { 8., 1., 1. };
rc = trexio_write_nucleus_num(f, 3);          assert (rc == TREXIO_SUCCESS);
rc = trexio_write_nucleus_charge(f, charge);  assert (rc == TREXIO_SUCCESS);
rc = trexio_close(f);

/* Read */
charge = { 0., 0., 0. };
f  = trexio_open("water.h5", 'r', TREXIO_HDF5, &rc);
rc = trexio_read_nucleus_num(f, n);           assert (rc == TREXIO_SUCCESS);
rc = trexio_read_nucleus_charge(f, charge);   assert (rc == TREXIO_SUCCESS);
rc = trexio_close(f);
```

TREXIO tools

`https://github.com/TREX-CoE/trexio_tools`

Repository of tools to manipulate TREXIO files:

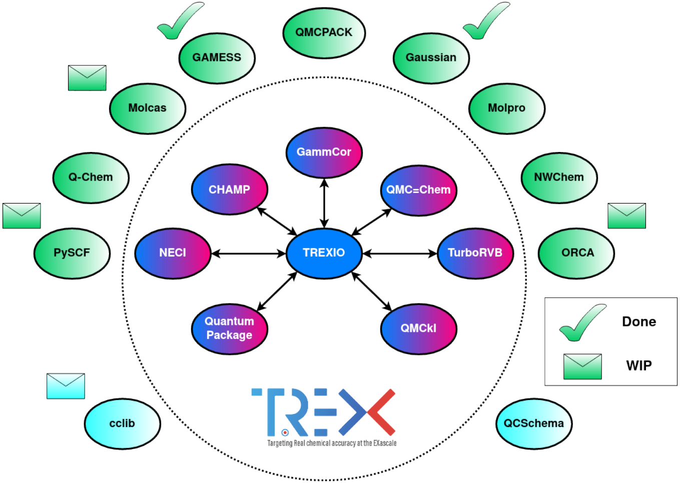
- Cartesian ⟷ Spherical AO conversion
- Numerical computation of AO overlap matrix compared with the one stored in the file: ⟹ debugging
- Compare numerical computation of MO overlap matrix with identity: ⟹ debugging when no integrals are available
- Converters for GAMESS, Gaussian, PySCF, FCIDUMP

TREXIO tools

Ideas for other tools:

- Decontract basis set
- Re-order atoms
- Compute AOs or MOs on a grid
- Convert from TEXT to HDF5 back-end
- CSF ⟷ Determinant conversion
- Compute the energy using integrals and density matrices
- ...
- Contributions welcome! :-)

Current situation



GAMESS ✓
QMCPACK
Gaussian ✓
Molcas ✉
Molpro
Q-Chem
NWChem
PySCF ✉
ORCA ✉
cclib ✉
QCSchema

GammCor
CHAMP
QMC=Chem
NECI
TREXIO
TurboRVB
Quantum Package
QMCkl

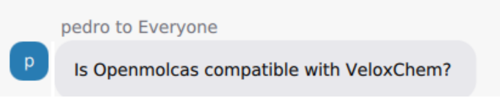
Targeting Real chemical accuracy at the EXascale

✓ Done
✉ WIP

Enabled science

- QP ⟶ TREXIO ⟶ GammCor: SAPT with CIPSI density matrices
- QP ⟶ TREXIO ⟶ CHAMP: geometry optimization of CIPSI wave functions
- Checking PySCF to TurboRVB interface:
    - PySCF ⟶ TREXIO ⟶ QP: check energy
    - TREXIO ⟶ TurboRVB: OK
- QP ⟶ TREXIO ⟶ NECI: Compare CIPSI and FCIQMC estimates of FCI energy
- QP ⟶ TREXIO ⟶ NECI ⟶ TREXIO ⟶ QMC=Chem: QMC with FCI wave function
- . . .

Conclusion



pedro to Everyone

p

Is Openmolcas compatible with VeloxChem?

If OpenMolcas can import/export data with TREXIO, new science will be easily made by composing OpenMolcas with any other code (existing or future) using TREXIO.

Quotation

"TREXIO will become the JPEG of quantum chemistry" (Anonymous)

- TREX: `https://trex-coe.eu`
- TREXIO documentation: `https://trex-coe.github.io/trexio/`
- TREXIO source code: `https://github.com/trex-coe/trexio/`