Coulomb-Sturmians and molsturm

Michael F. Herbst
michael.herbst@iwr.uni-heidelberg.de
http://michael-herbst.com

Interdisziplinäres Zentrum für wissenschaftliches Rechnen
Ruprecht-Karls-Universität Heidelberg

18 September 2017

Contents

1. Hartree-Fock
   - Discretisation of the HF equations
2. Coulomb-Sturmian basis sets
   - Coulomb Sturmians (CS)
   - Need for a `contraction`-based scheme
3. Lazy matrices in quantum chemistry
   - `contraction`-based algorithms
   - The `lazyten` lazy matrix library
   - `molsturm` program package
4. Future work
   - Outlook

Contents

1. Hartree-Fock
   - Discretisation of the HF equations
2. Coulomb-Sturmian basis sets
   - Coulomb Sturmians (CS)
   - Need for a `contraction`-based scheme
3. Lazy matrices in quantum chemistry
   - `contraction`-based algorithms
   - The `lazyten` lazy matrix library
   - `molsturm` program package
4. Future work
   - Outlook

Hartree-Fock equations

- Hartree-Fock equations

$$\left(-\frac{1}{2}\Delta + \hat{\mathcal{V}}_{\mathrm{Nuc}} + \hat{\mathcal{V}}_{\mathrm{H}}\left[\{\psi_f\}_{f\in I}\right] + \hat{\mathcal{V}}_{\mathrm{x}}\left[\{\psi_f\}_{f\in I}\right] - \varepsilon_f\right)\psi_f(\underline{\boldsymbol{r}}) = 0$$

with

$-\frac{1}{2}\Delta$	Kinetic energy of electrons
$\hat{\mathcal{V}}_{\mathrm{Nuc}}$	Electron-nuclear interaction
$\hat{\mathcal{V}}_{\mathrm{H}}\left[\{\psi_f\}_{f\in I}\right]$	Hartree potential
$\hat{\mathcal{V}}_{\mathrm{x}}\left[\{\psi_f\}_{f\in I}\right]$	Exchange potential

- Non-linear system of partial differential equations
- Non-linear eigenproblem for eigenpairs $\{(\varepsilon_f, \psi_f)\}_{f\in I}$

Discretisation of Hartree-Fock

- Galerkin projection
- Discretise using finite basis $\{\varphi_b\}_{b\in B}$:

$$\psi_f \to \tilde{\psi}_f = \sum_b c_b^{(f)} \varphi_b$$

$\Rightarrow$ Non-linear discretised Eigenproblem

$$\left(\mathbf{T} + \mathbf{V}_{\mathrm{Nuc}} + \mathbf{J}\left[\{c_b^{(f)}\}\right] + \mathbf{K}\left[\{c_b^{(f)}\}\right]\right) \underline{\boldsymbol{c}}^{(f)} = \varepsilon_f \underline{\boldsymbol{c}}^{(f)}$$

- Variational, but which choices of φ_b are best?

An ideal basis

- Represents physical system well
- Results reliable
  - Error margin known
  - Systematic improvement possible
- Prior knowledge
  - Little required
  - If available: Can be incorporated
- Integrals and eigenproblem are feasible

$\Rightarrow$ In reality need a good compromise

Properties of HF solution

Nuclear cusp

- Kato's[1] electron-nucleus cusp condition:

$$\left.\frac{\partial\langle\Psi(\underline{\boldsymbol{x}})\rangle}{\partial r_i}\right|_{\underline{\boldsymbol{r}}_i=\underline{\boldsymbol{r}}_A} = -Z_A\,\langle\Psi(\underline{\boldsymbol{x}})\rangle|_{\underline{\boldsymbol{r}}_i=\underline{\boldsymbol{r}}_A}$$

  for each nucleus A and electron i where
  - $\underline{\boldsymbol{x}}^\dagger \equiv \left(\underline{\boldsymbol{r}}_1^\dagger, \ldots, \underline{\boldsymbol{r}}_N^\dagger\right)$
  - $\Psi(\underline{\boldsymbol{x}})$: Exact solution of Schrödinger equation
  - $\langle\Psi(\underline{\boldsymbol{x}})\rangle|_{\underline{\boldsymbol{r}}_i=\underline{\boldsymbol{r}}_A}$: Average value of $\Psi(\underline{\boldsymbol{x}})$ in hypersphere with $\underline{\boldsymbol{r}}_i = \underline{\boldsymbol{r}}_A$ fixed
- Derivative at nuclear cusp is non-zero

$\Rightarrow$ Single determinant / HF solution should have e-n cusps

[1] Kato, *Com. Pure Appl. Math*, **1957**, *10* 151

Properties of HF solution

Exponential decay

- Consider the HF equations

$$0 = \left(-\frac{1}{2}\Delta + \hat{\mathcal{V}}_{\mathrm{Nuc}} + \hat{\mathcal{V}}_{\mathrm{H}}\left[\,\{\psi_f\}_{f\in I}\,\right] + \hat{\mathcal{V}}_{\mathrm{x}}\left[\,\{\psi_f\}_{f\in I}\,\right] - \varepsilon_f \right) \psi_f(\underline{\boldsymbol{r}})$$

- At $r \to \infty$:

$$0 = \left(-\frac{1}{2}\Delta - \frac{Z_{\mathrm{net}}}{r} - \varepsilon_f \right) \psi_f(\underline{\boldsymbol{r}})$$

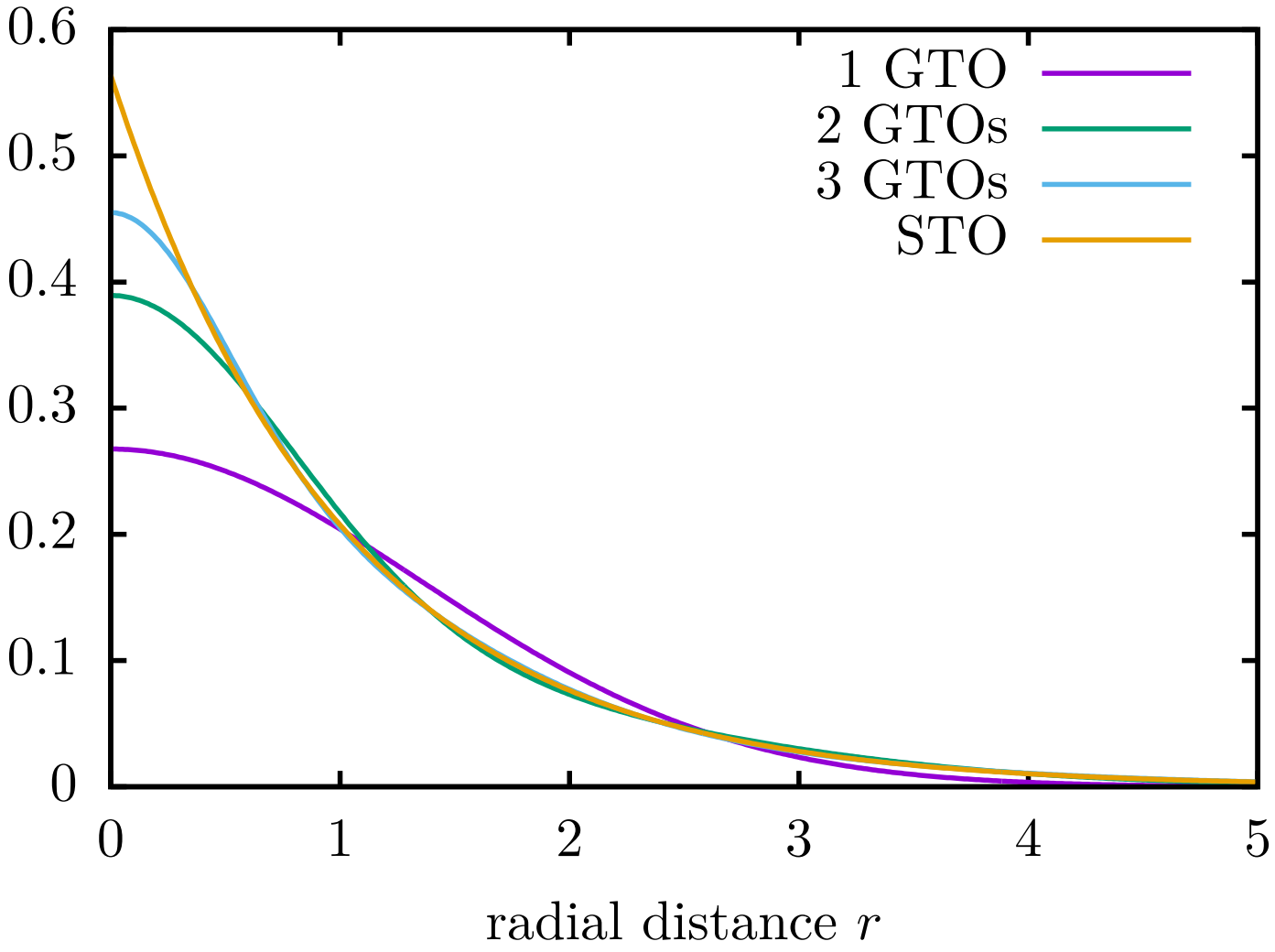
where $Z_{\mathrm{net}} = 1$ for neutral systems.

- This gives an exponential decay:

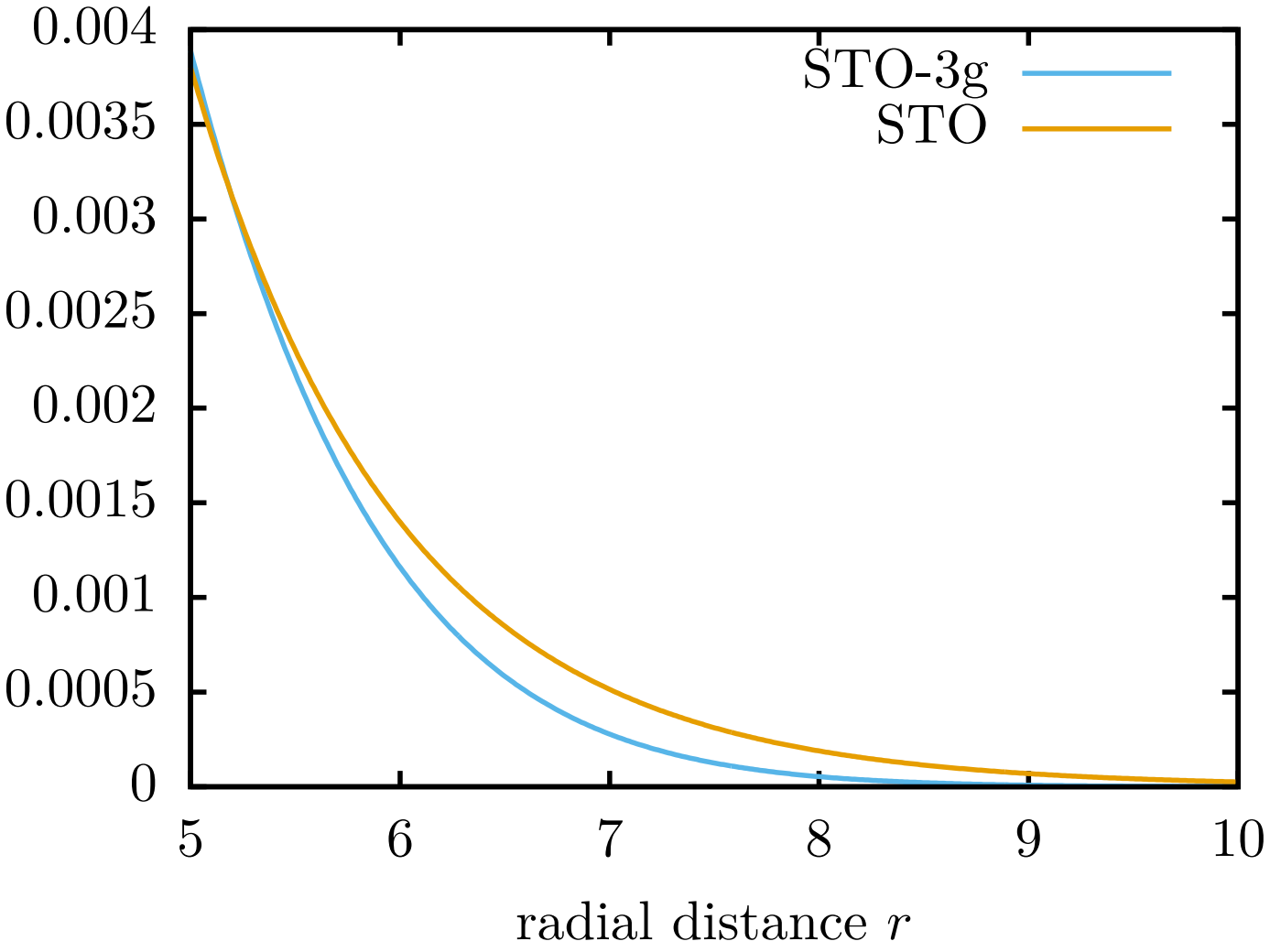
$$\psi_f(\underline{\boldsymbol{r}}) \to \exp(-\sqrt{-2\varepsilon_f}\, r) \quad \text{as } r \to \infty$$

$\Rightarrow$ Decay energy-dependent

Hydrogen atom radial part



Hydrogen atom radial part



Local energy

- Assume a one-electron system
- Discretised solution $\tilde{\psi}_f$ not *exact*:

$$0 \neq \left(-\frac{1}{2}\Delta + \hat{\mathcal{V}}_{\text{Nuc}} - \varepsilon_f\right)\tilde{\psi}_f(\underline{\boldsymbol{r}}) \quad \Rightarrow \quad \varepsilon_f \neq \frac{-\frac{1}{2}\Delta\tilde{\psi}_f(\underline{\boldsymbol{r}})}{\tilde{\psi}_f(\underline{\boldsymbol{r}})} + \hat{\mathcal{V}}_{\text{Nuc}}$$

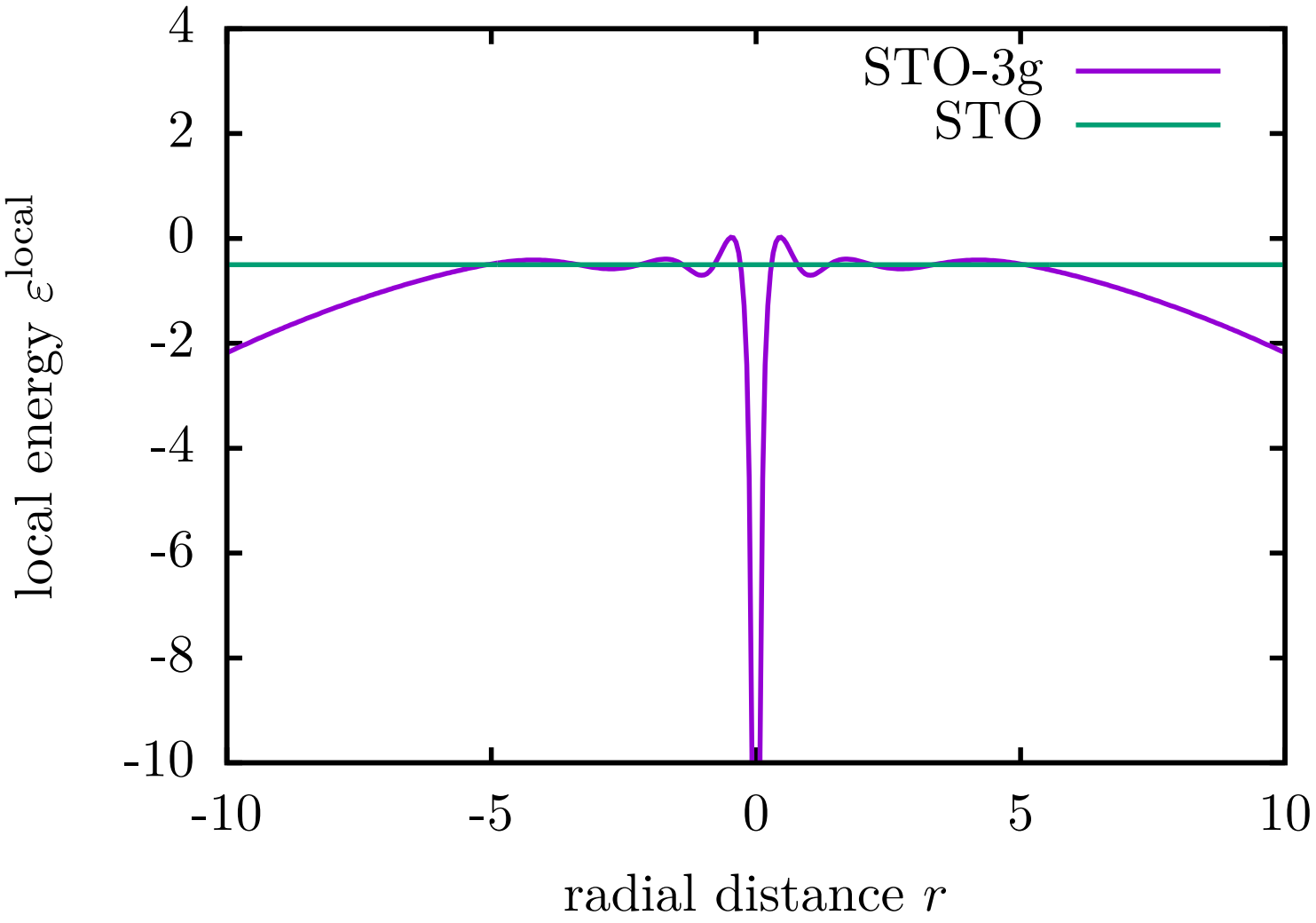
- Local orbital energy:

$$\varepsilon_f^{\text{local}}(\underline{\boldsymbol{r}}) = \frac{-\frac{1}{2}\Delta\tilde{\psi}_f(\underline{\boldsymbol{r}})}{\tilde{\psi}_f(\underline{\boldsymbol{r}})} + \hat{\mathcal{V}}_{\text{Nuc}}$$

- Measure for exactness of approximation[1]
- Similar for many-electron HF / Schrödinger equation

[1] P.-F. Loos et.al. CMMSE 2017

Hydrogen atom local energy



Gaussian basis sets

- Gaussian-product theorem:
  - ERI tensor $\langle \varphi_a \varphi_b | \varphi_c \varphi_d \rangle$ feasible
  - $\mathbf{J}$ and $\mathbf{K}$ feasible
- Need contracted GTOs (cGTOs)
- Nuclear cusp and tail still problematic

$\Rightarrow$ Some tasks require special basis sets (e.g. diffuse functions)

- cGTOs can become strongly linearly dependant
- Small to moderate basis sizes
- Basis set families with known convergence properties

Contents

1. Hartree-Fock
   - Discretisation of the HF equations
2. Coulomb-Sturmian basis sets
   - Coulomb Sturmians (CS)
   - Need for a `contraction`-based scheme
3. Lazy matrices in quantum chemistry
   - `contraction`-based algorithms
   - The `lazyten` lazy matrix library
   - `molsturm` program package
4. Future work
   - Outlook

Functional form

- Iso-energetic solutions φ_{nlm} to hydrogen-like equation

$$\left(-\frac{1}{2}\Delta - \beta_n \frac{Z}{r}\right)\varphi_{nlm}(\underline{\boldsymbol{r}}) = E\varphi_{nlm}(\underline{\boldsymbol{r}})$$

- Scaling factor β_n chosen to uniform energy:

$$\beta_n = \frac{kn}{Z} \quad \Rightarrow \quad E = -\frac{k^2}{2}$$

- φ_{nlm} look like hydrogenic orbitals with $\frac{Z}{n}$ replaced by k:

$$\varphi_{100}(\underline{\boldsymbol{r}}) = \sqrt{\frac{k^3}{\pi}}e^{-kr} \qquad \text{(1s)}$$

$$\varphi_{200}(\underline{\boldsymbol{r}}) = 2\sqrt{k^3}(1-kr)e^{-kr} \qquad \text{(2s)}$$

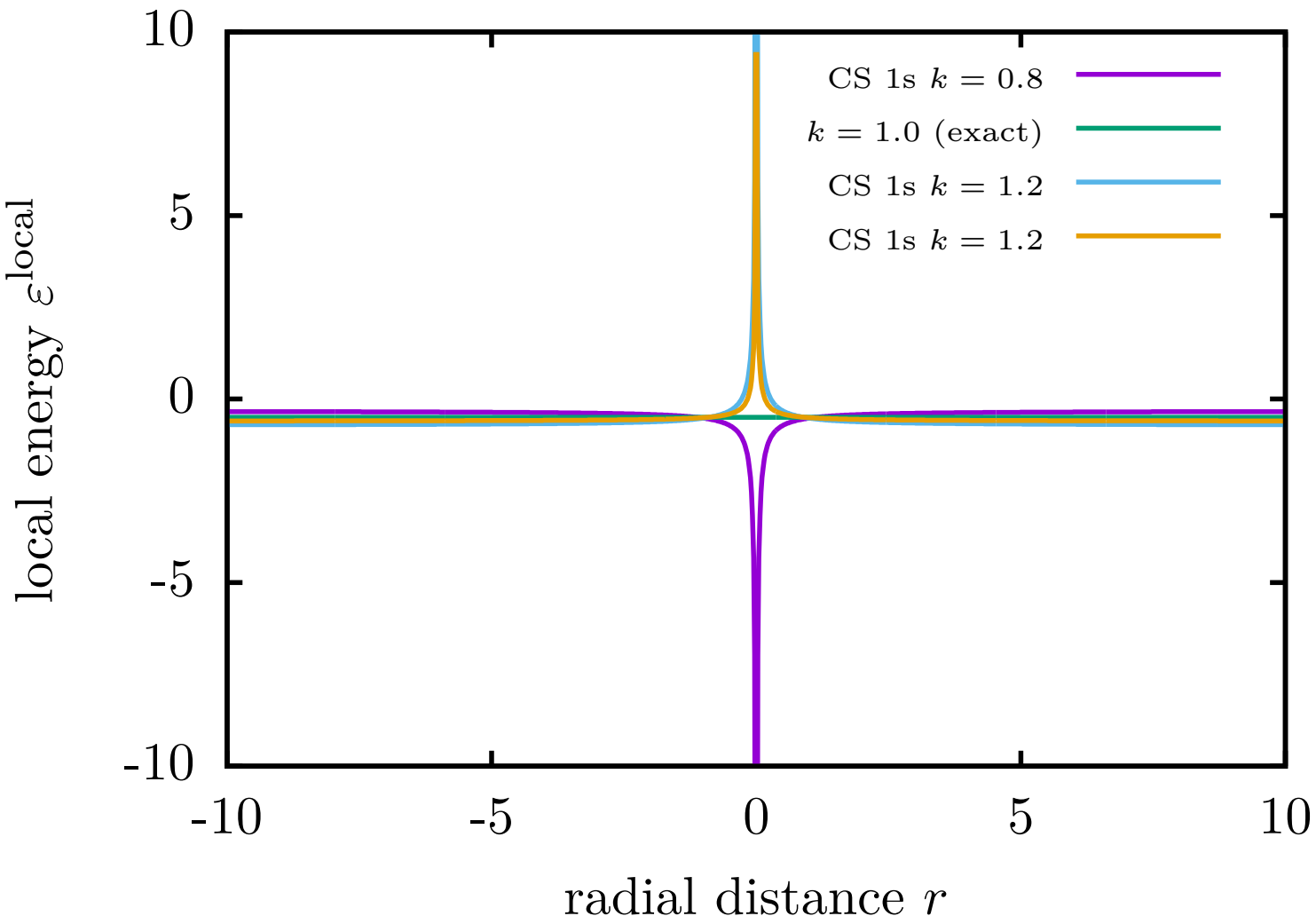
$$\varphi_{211}(\underline{\boldsymbol{r}}) = 2\sqrt{\frac{k^3}{3}}kze^{-kr} \qquad \text{(2pz)}$$

Properties

- Only atoms: Different Sturmians for molecules required
- Complete basis for $H^1(\mathbb{R}^3)$
- Correctly represent nuclear cusp
- Proper exponential decay for large r
- Basis has free tuning parameter k

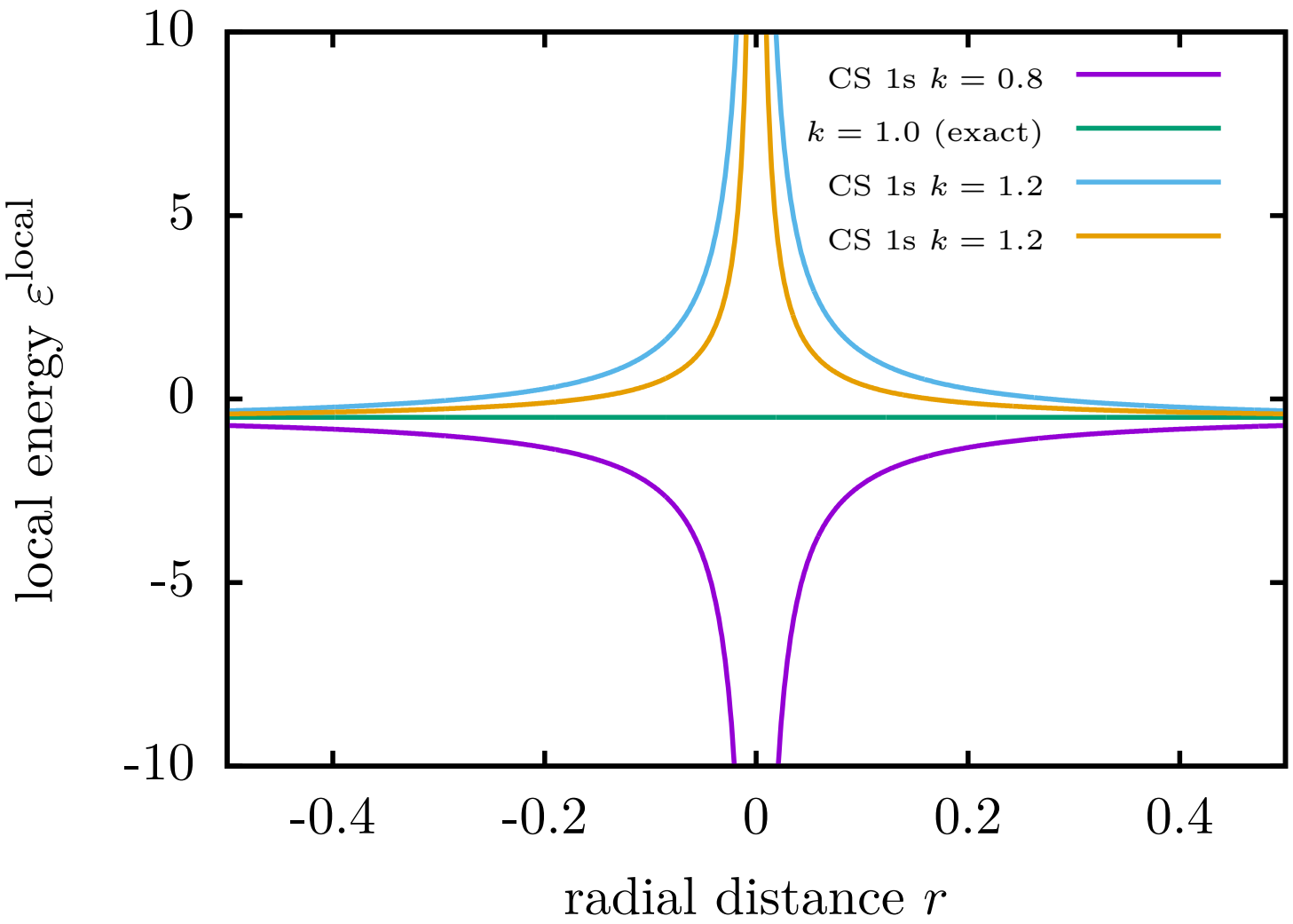
k-dependence

local orbital energy



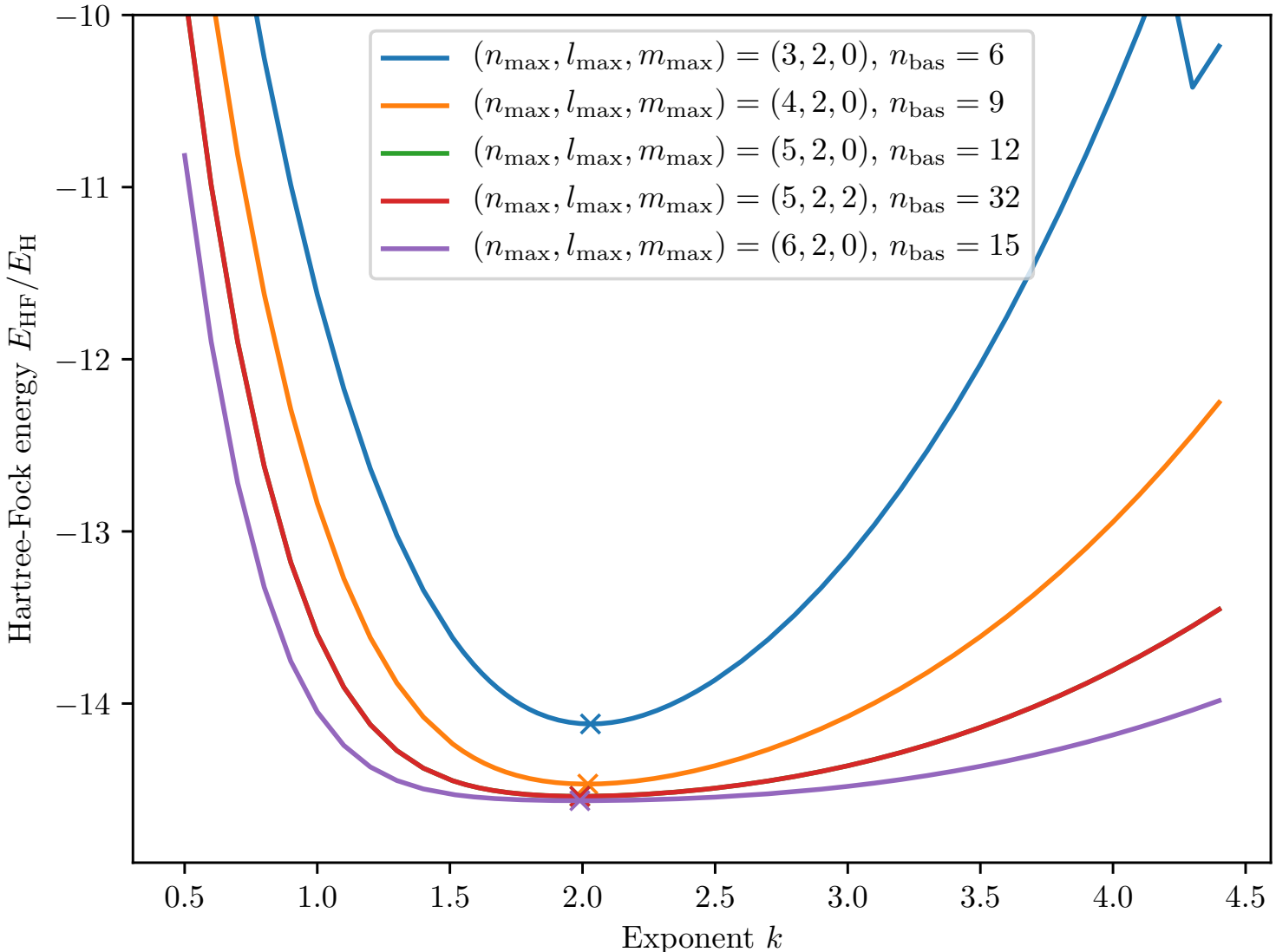
k-dependence

local orbital energy



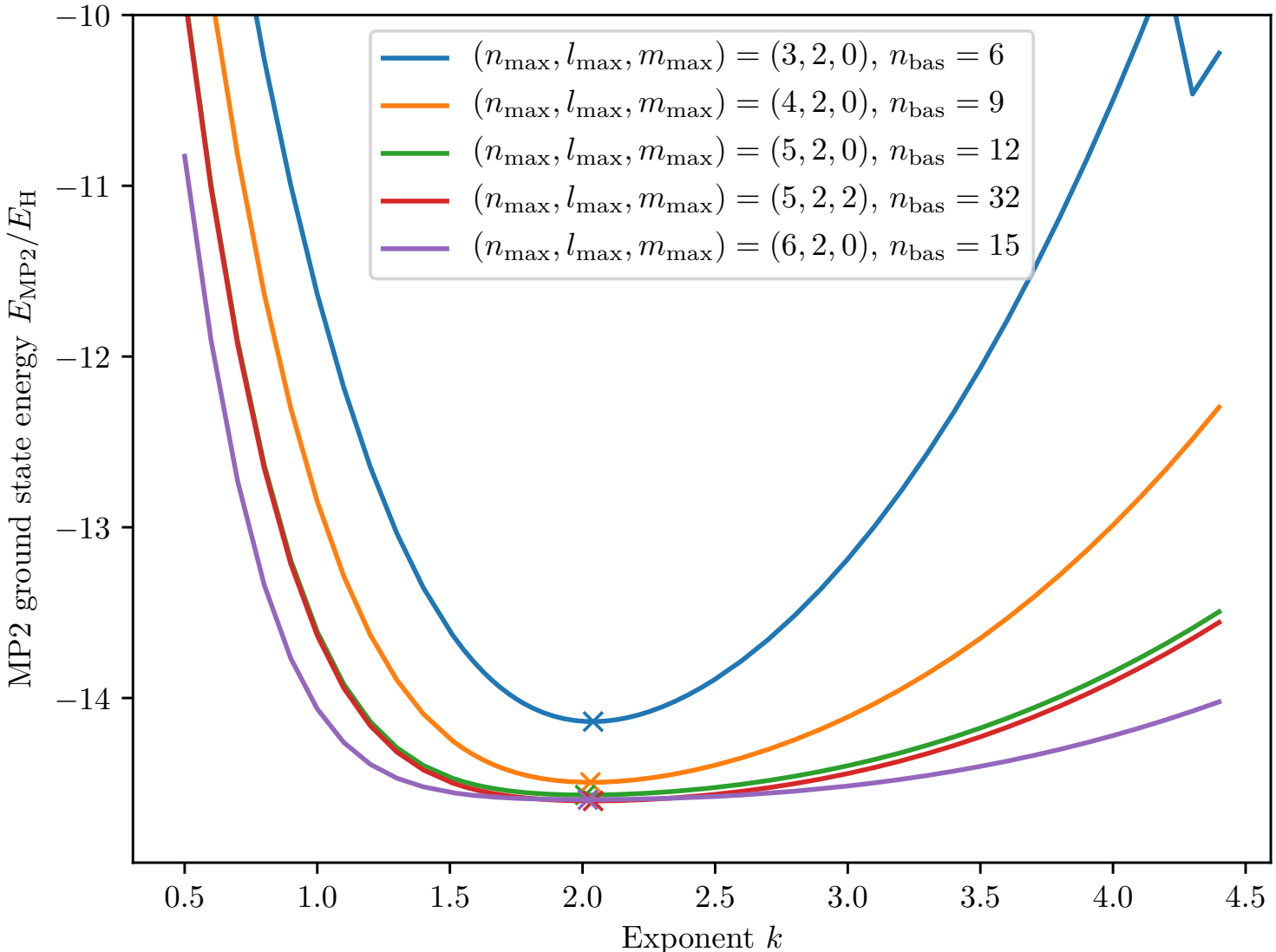
k-dependence

HF energy



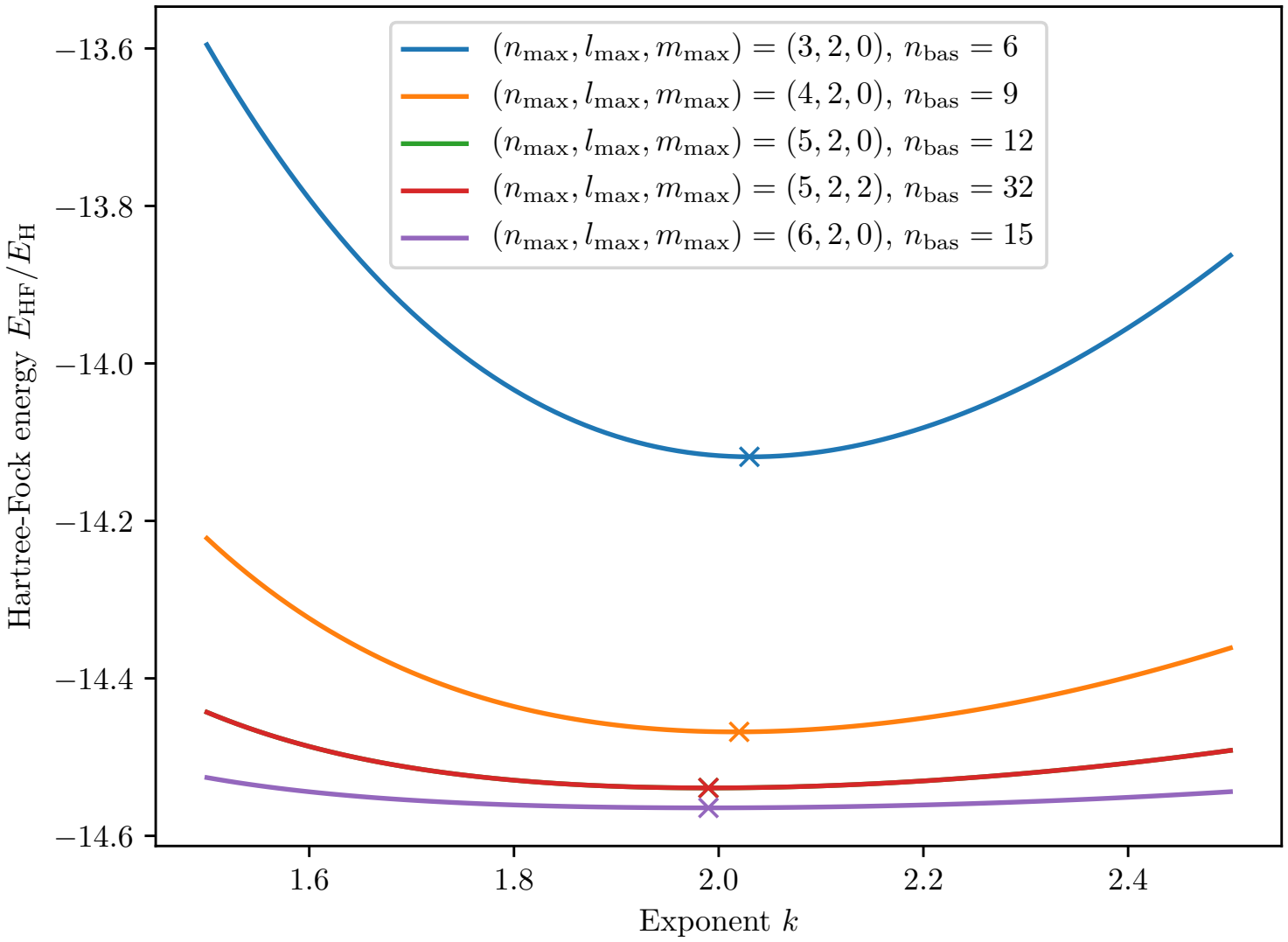
k-dependence

MP2 energy



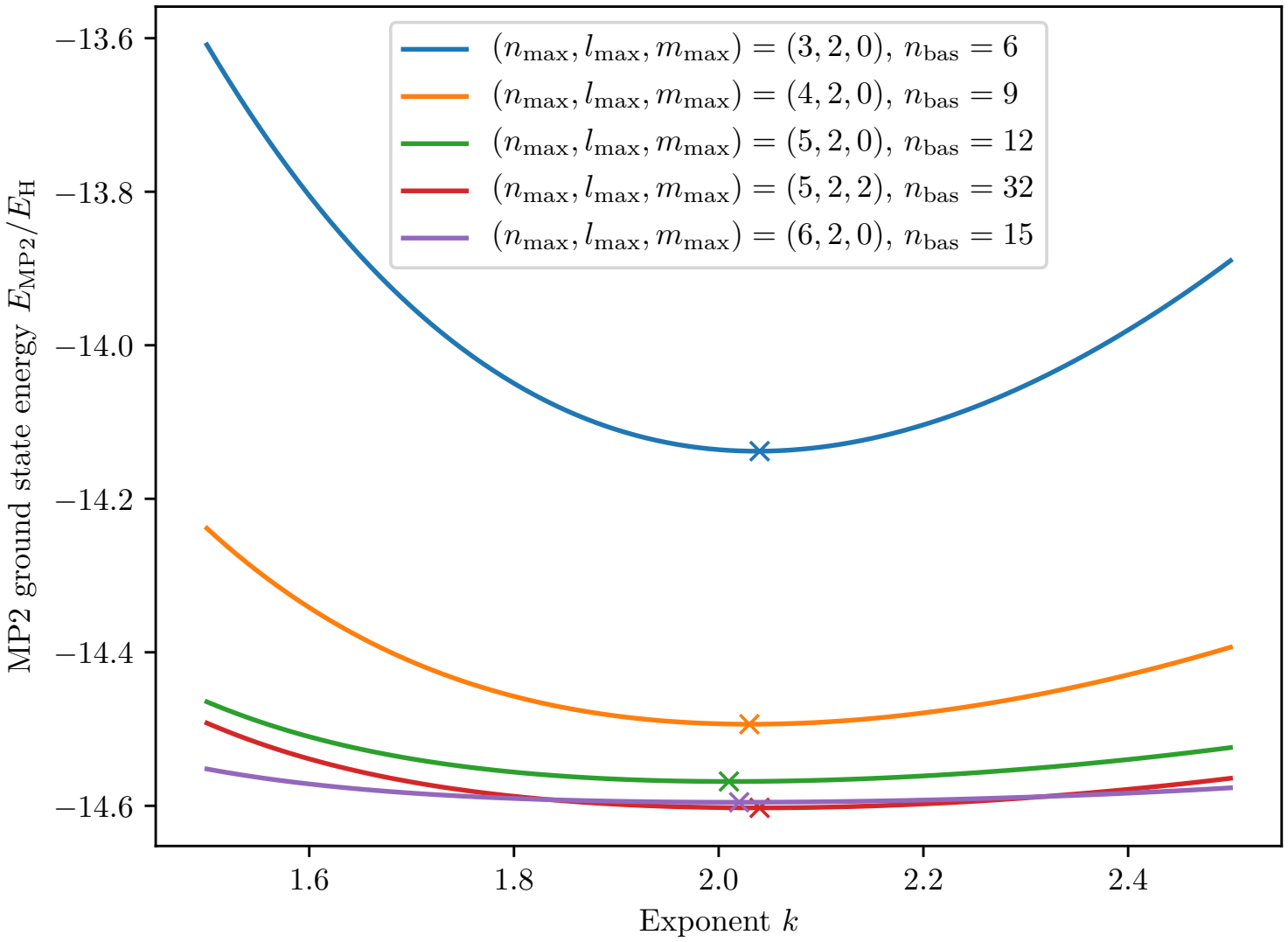
k-dependence

HF energy



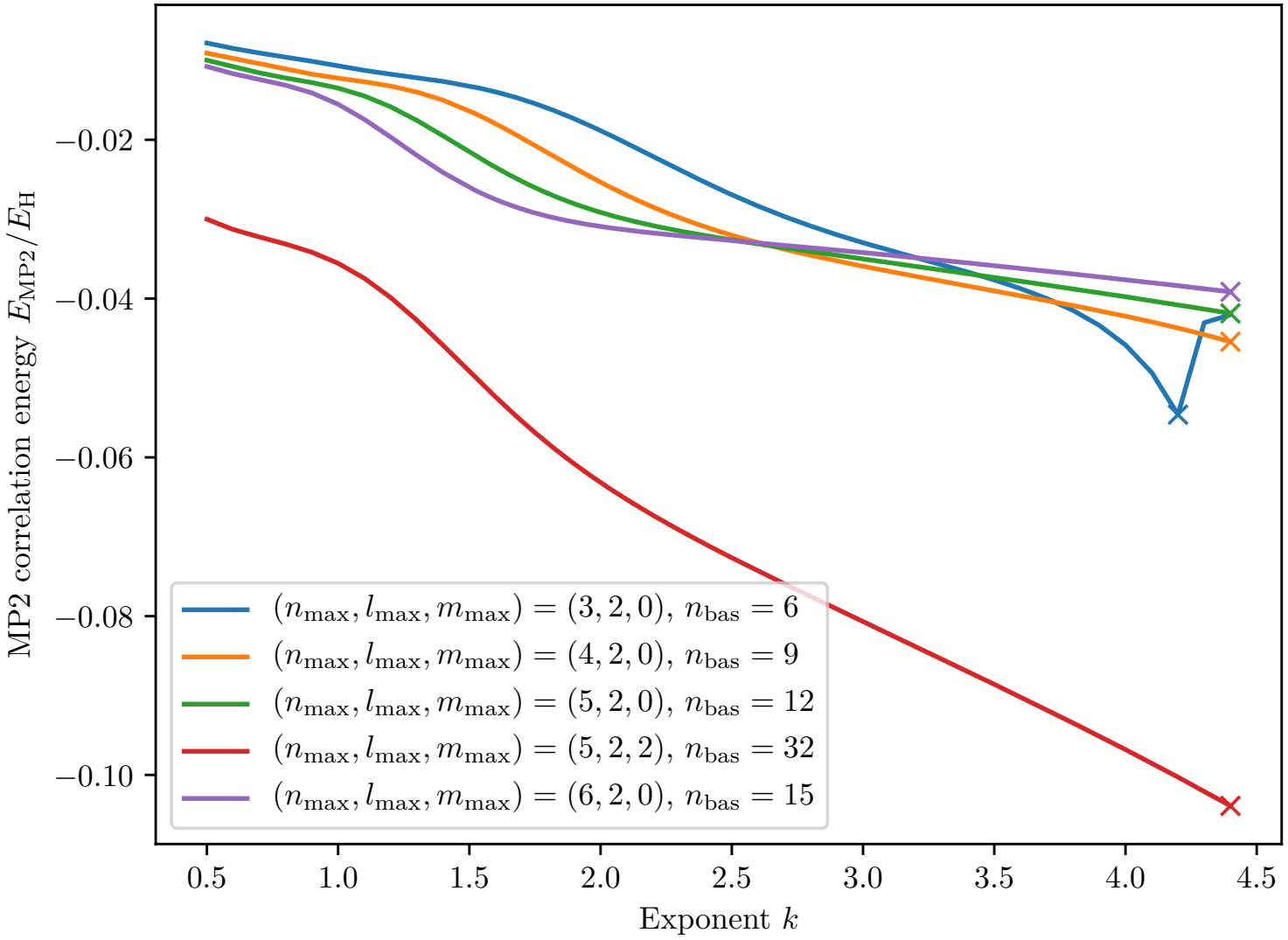
k-dependence

MP2 energy



k-dependence

MP2 energy



Preliminary results

- Helium HF energies:

22 Sturmians	$-2.861\,679\,995\,612\,\mathrm{E_H}$
CBS value[1]	$-2.861\,679\,995\,615\,\mathrm{E_H}$

Coulomb Sturmians with $n_{\max} = 22$, $l_{\max} = m_{\max} = 0$, $k = 2$

- Beryllium comparison:

	E_{HF}	E_{MP2}
21 Sturmians	$-14.5634\,\mathrm{E_H}$	$-14.6248\,\mathrm{E_H}$
21 cGTOs	$-14.5730\,\mathrm{E_H}$	$-14.6217\,\mathrm{E_H}$

Coulomb Sturmians with $n_{\max} = 6, l_{\max} = m_{\max} = 1$, $k = 2.1$,
cGTOs: cc-pV5Z, but only $l = 0$ and $l = 1$

[1] N. H. Morgon et. al., *Comp. Theor. Chem.*, **1997**, *394*, 95-100

Preliminary results

- Helium HF energies:

22 Sturmians	$-2.861\,679\,995\,612\,\mathrm{E_H}$
CBS value[1]	$-2.861\,679\,995\,615\,\mathrm{E_H}$

Coulomb Sturmians with $n_{\max} = 22$, $l_{\max} = m_{\max} = 0$, $k = 2$

- Beryllium comparison:

	E_{HF}	E_{MP2}
21 Sturmians	$-14.5634\,\mathrm{E_H}$	$-14.6248\,\mathrm{E_H}$
21 cGTOs	$-14.5730\,\mathrm{E_H}$	$-14.6217\,\mathrm{E_H}$

Coulomb Sturmians with $n_{\max} = 6, l_{\max} = m_{\max} = 1$, $k = 2.1$,
cGTOs: cc-pV5Z, but only $l = 0$ and $l = 1$

[1] N. H. Morgon et. al., *Comp. Theor. Chem.*, **1997**, *394*, 95-100

One-electron integrals

- Set $\mu \equiv nlm$
- One-electron integrals are sparse:

$$S_{\mu',\mu} = \delta_{m',m}\delta_{l',l} \begin{cases} n = n' & 1 \\ |n-n'| = 1 & s^{(l)}_{n,n+1} \in \mathbb{R} \\ \text{else} & 0 \end{cases}$$

$$(V_{\text{Nuc}})_{\mu',\mu} = \delta_{\mu',\mu}\frac{kZ}{n}$$

$$T_{\mu',\mu} = k^2(\delta_{\mu',\mu} - S_{\mu',\mu})$$

where

$$s^{(l)}_{n,n+1} = -\frac{1}{2}\sqrt{\frac{(n-l)(n+l+1)}{n(n+1)}}$$

Two-electron integrals

- ERI tensor from pre-computed sparse tensor $I_{\mu\nu}$:

$$\langle \varphi_{\mu_1}\varphi_{\mu_2} | \varphi_{\mu_3}\varphi_{\mu_4} \rangle = \sum_{\mu\nu} \mathcal{C}^{\mu}_{\mu_1\mu_2} I_{\mu\nu} \mathcal{C}^{\nu}_{\mu_3\mu_4}$$

- The $\mathcal{C}^{\mu}_{\mu_1\mu_2}$ are the coefficients in

$$\varphi^*_{\mu_1}(\underline{\boldsymbol{r}})\varphi_{\mu_2}(\underline{\boldsymbol{r}}) = \sum_{\mu} \mathcal{C}^{\mu}_{\mu_1\mu_2} \varphi_{\mu}(2k, \underline{\boldsymbol{r}})$$

- $\mathcal{C}^{\mu}_{\mu_1\mu_2}$ obtained by simple operation from pre-computed data

$\Rightarrow$ Like *exact* density fitting approximation

Exploiting the properties of Coulomb Sturmians

- One-electron integrals simple, closed-form expressions

⇒ Storing values is wasted memory

- Building $\mathbf{K}$ is $\mathcal{O}(N^2)$:

$$K_{bd} = \sum_{aco\mu\nu} c_a^{(o)} \; \mathcal{C}_{ab}^{\mu} I_{\mu\nu} \mathcal{C}_{cd}^{\nu} \; c_c^{(o)}$$

- Applying $\mathbf{K}$ is $\mathcal{O}(N)$:

$$\left(K\tilde{c}^{(f)}\right)_b = \sum_{acdo\mu\nu} c_a^{(o)} \; \mathcal{C}_{ab}^{\mu} I_{\mu\nu} \mathcal{C}_{cd}^{\nu} \; c_c^{(o)} \tilde{c}_d^{(f)}$$

- Similar for $\mathbf{J}$.

contraction-based scheme

- Iterative solvers only need matrix-vector products

⇒ Contraction-based or matrix-free algorithm:

  - Never build fock matrix in storage
  - Use matrix-vector `contraction` expressions
  - Employ SCF based on orbital coefficients $\underline{\boldsymbol{c}}^{(f)}$

- In the end we need

$$\mathbf{F} = \mathbf{T} + \mathbf{V}_{\mathrm{Nuc}} + \mathbf{J} + \mathbf{K} + \cdots$$

⇒ Expressions for `contraction` complicated to code

Contents

1. Hartree-Fock
   - Discretisation of the HF equations
2. Coulomb-Sturmian basis sets
   - Coulomb Sturmians (CS)
   - Need for a `contraction`-based scheme
3. Lazy matrices in quantum chemistry
   - `contraction`-based algorithms
   - The `lazyten` lazy matrix library
   - `molsturm` program package
4. Future work
   - Outlook

Characteristics of `contraction`-based algorithms

Advantages

- Scaling (storage and time) reduced — in examples to $\mathcal{O}(N)$
- Parallelisation easier
  - $\Rightarrow$ Less data management
- Hardware trends are in favour

Disadvantages

- Matrices more intuitive than `contraction`-functions
- More computations
  - $\Rightarrow$ Need efficient contraction schemes for the `contraction`
  - $\Rightarrow$ Algorithms more complex

Characteristics of `contraction`-based algorithms

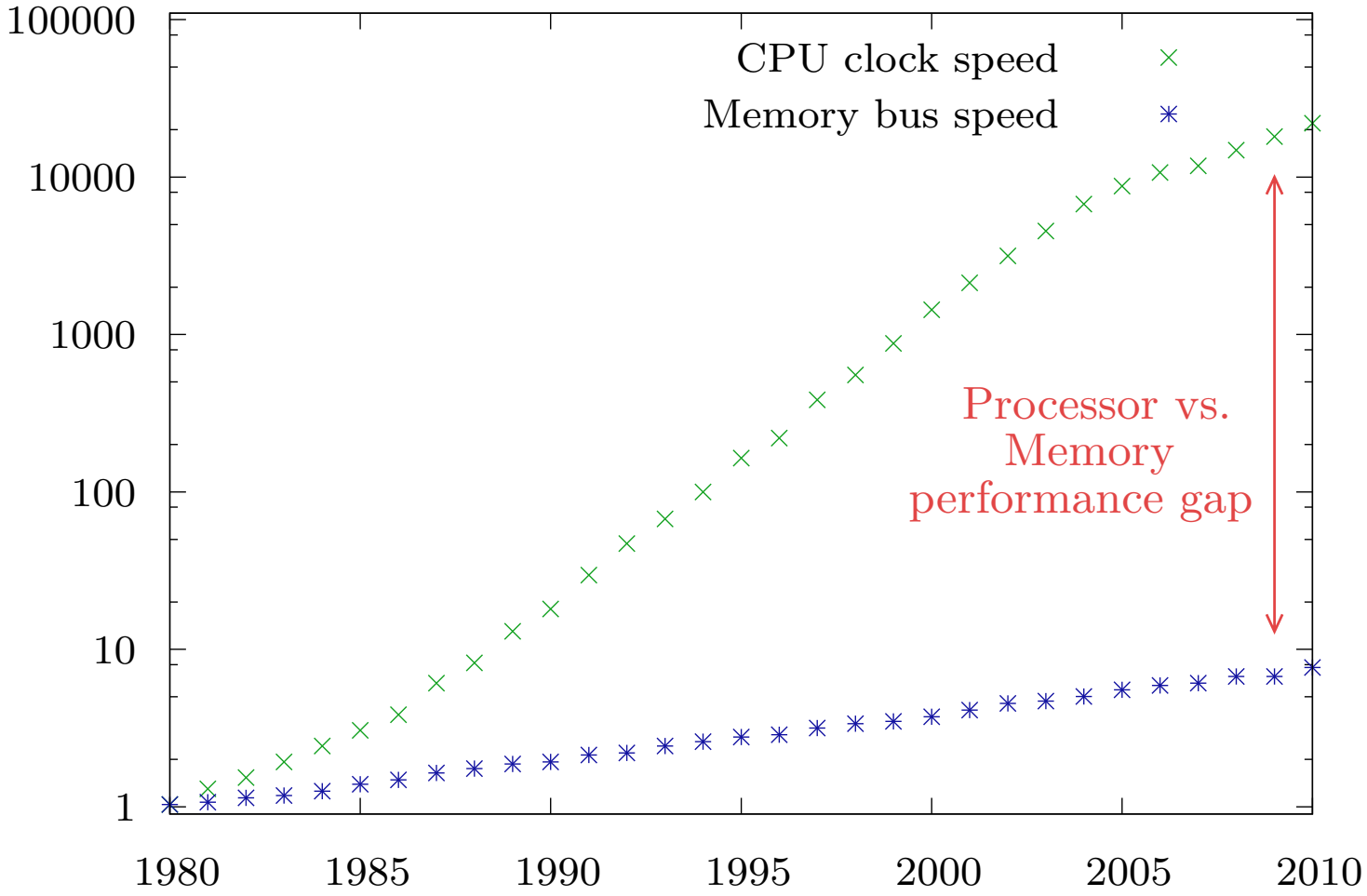
Advantages

- Scaling (storage and time) reduced — in examples to $\mathcal{O}(N)$
- Parallelisation easier
  - $\Rightarrow$ Less data management
- Hardware trends are in favour

Disadvantages

- Matrices more intuitive than `contraction`-functions
- More computations
  - $\Rightarrow$ Need efficient contraction schemes for the `contraction`
  - $\Rightarrow$ Algorithms more complex

Characteristics of contraction-based algorithms



Data from https://dave.cheney.net/2014/06/07/five-things-that-make-go-fast

Characteristics of `contraction`-based algorithms

Advantages

- Scaling (storage and time) reduced — in examples to $\mathcal{O}(N)$
- Parallelisation easier
  - $\Rightarrow$ Less data management
- Hardware trends are in favour

Disadvantages

- Matrices more intuitive than `contraction`-functions
- More computations
  - $\Rightarrow$ Need efficient contraction schemes for the `contraction`
  - $\Rightarrow$ Algorithms more complex

Lazy matrices

- Stored matrix: All elements reside in memory
- Lazy matrix:
  - Generalisation of matrices
    - State
    - Non-linear
    - Elements may be expressions

  ⇒ Obtaining elements expensive
  - Evaluation of internal expression: Delayed until `contraction`
  - For convenience: Offer matrix-like interface

Using lazy matrices

- Program as usual

$$\mathbf{D} = \mathbf{A} + \mathbf{B}$$

- Build expression tree internally

$$\boxed{\mathbf{D}} = \boxed{\mathbf{A}} + \boxed{\mathbf{B}}$$

- On application:

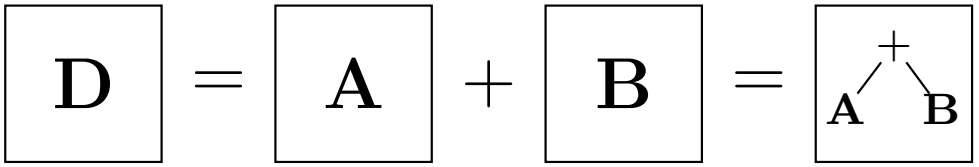
$$\mathbf{D}\underline{x} = (\mathbf{A}\underline{x}) + (\mathbf{B}\underline{x})$$

Using lazy matrices

- Program as usual

$$\mathbf{D} = \mathbf{A} + \mathbf{B}$$

- Build expression tree internally



- On application:

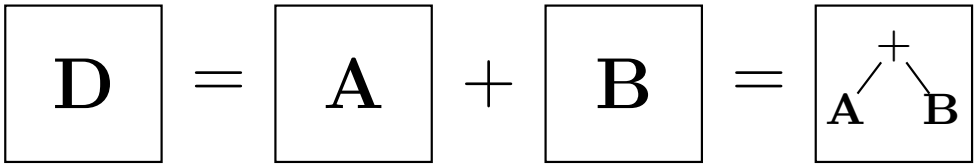
$$\mathbf{D}\underline{x} = (\mathbf{A}\underline{x}) + (\mathbf{B}\underline{x})$$

Using lazy matrices

- Program as usual

$$\mathbf{D} = \mathbf{A} + \mathbf{B}$$

- Build expression tree internally



- On application:

$$\mathbf{D}\underline{\boldsymbol{x}} = (\mathbf{A}\underline{\boldsymbol{x}}) + (\mathbf{B}\underline{\boldsymbol{x}})$$

Notes and observations

- `lazyten`: Bookkeeping for `contraction`-functions
- Programmer still sees matrices

⇒ Language for writing `contraction`-based algorithms

- Lazy matrices allow layered responsibility for computation, e.g. $(\mathbf{A} + \mathbf{B})\underline{\boldsymbol{x}}$
  - $\mathbf{A}\underline{\boldsymbol{x}}$ and $\mathbf{B}\underline{\boldsymbol{x}}$ decided by implementation of $\mathbf{A}$ and $\mathbf{B}$
  - $(\mathbf{A}\underline{\boldsymbol{x}}) + (\mathbf{B}\underline{\boldsymbol{x}})$ done in linear algebra backend

⇒ Proper modularisation between

  - Higher-level algorithms
  - Lazy matrix implementations
  - LA backends

Interface and example code

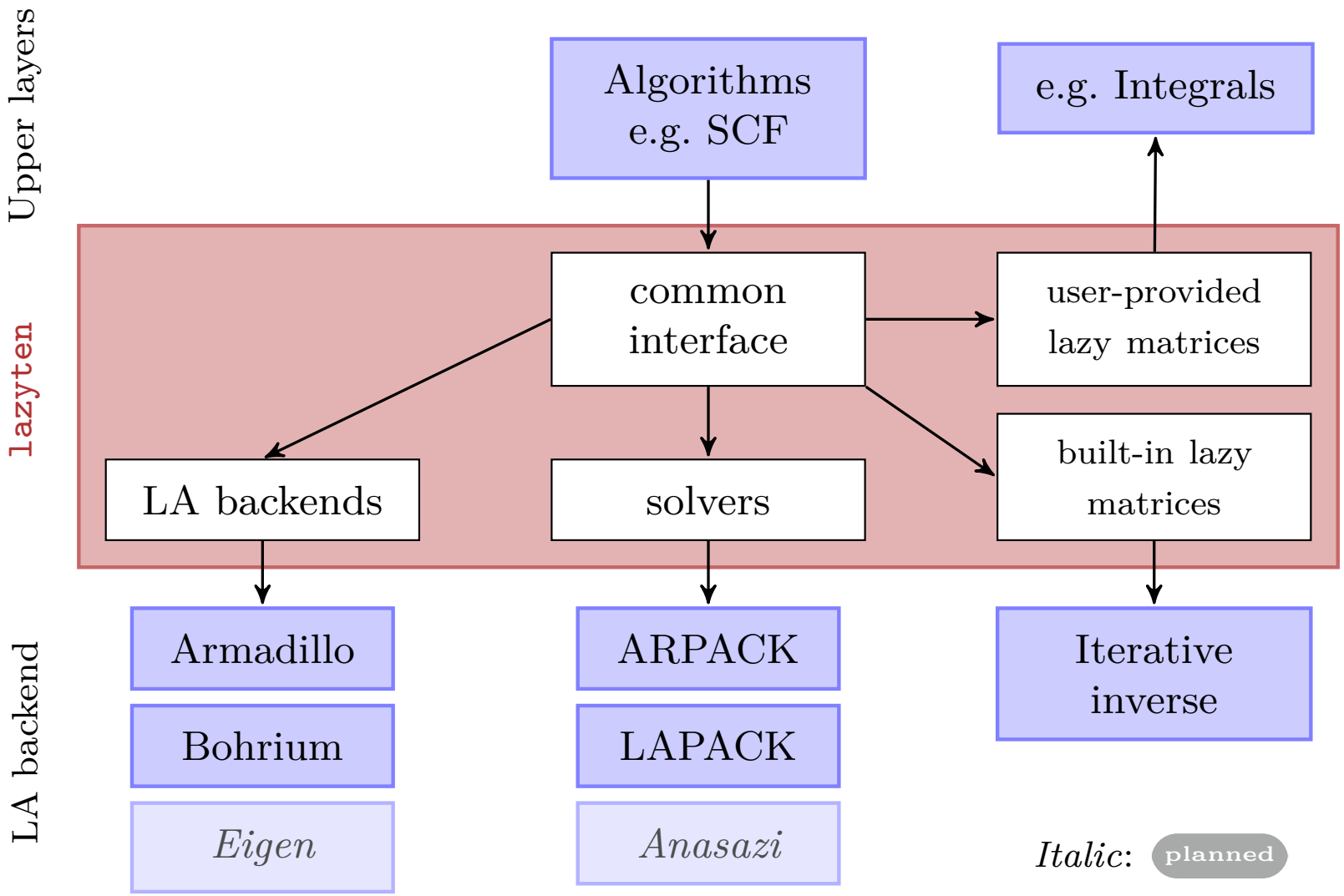
- lazyten[1]: Prototype C++ implementation

```
typedef SmallVector<double> vector_type;
typedef SmallMatrix<double> matrix_type;
auto v = random<vector_type>(100);
DiagonalMatrix<matrix_type> diag(v);
auto a = random<matrix_type>(100,100);
auto b = random<matrix_type>(100,100);

// No computation: Just build expression tree
auto sum = diag + a;
auto projector = sum * inverse(sum);
auto tree = b - projector * b;

// Evaluate tree on application:
SmallVector<double> res = tree * v;
```

[1] https://lazyten.org

lazyten

Lazy linear algebra wrapper library



What's in it for me?

- `contraction`-based algorithms

⇒ Lower memory footprint, scaling improvements

- Readable code

⇒ Great for teaching or to play around

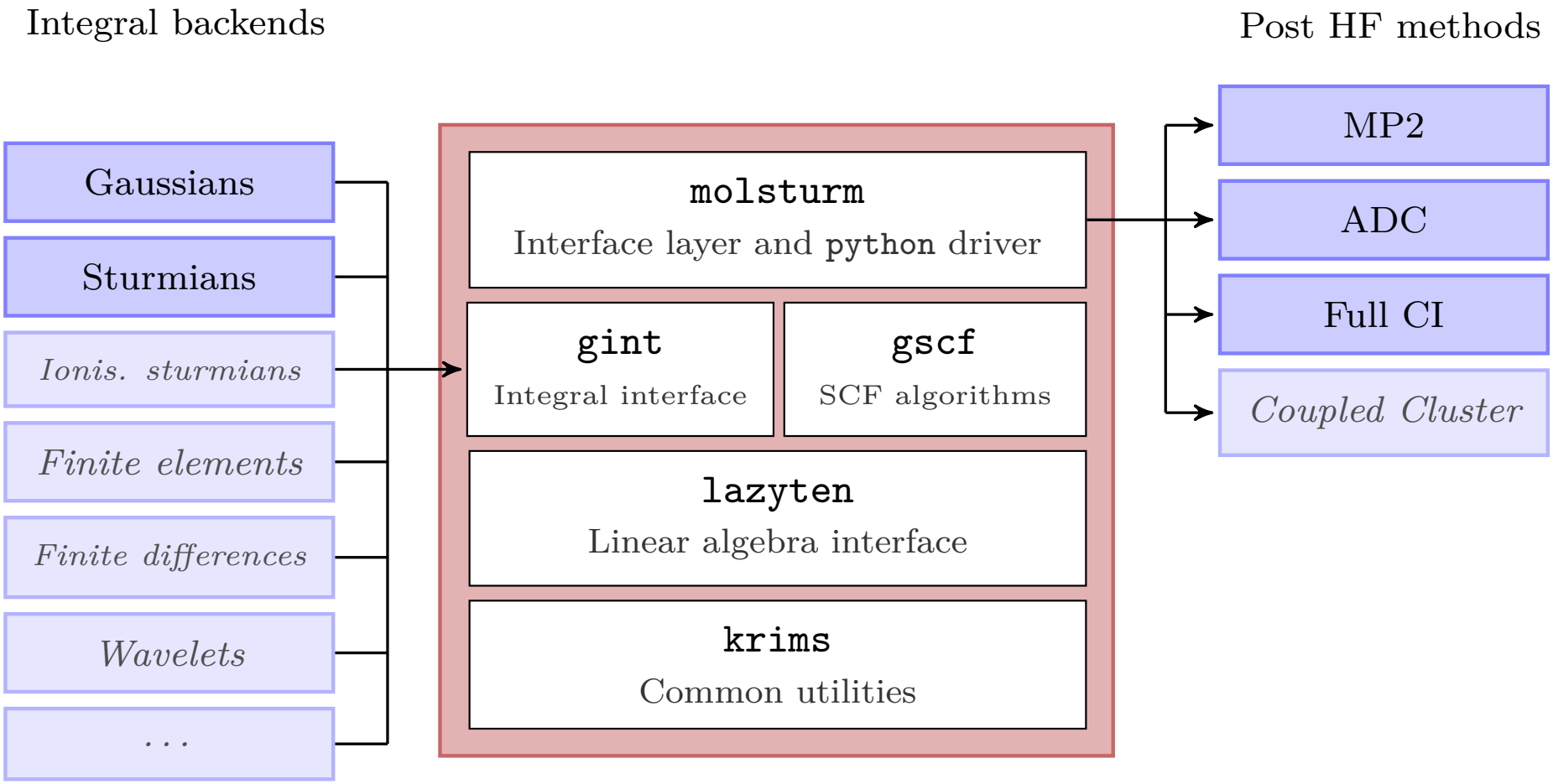
- Abstraction between integrals and SCF algorithms

⇒ Plug and play integral libraries

⇒ Swap LA backends

⇒ Basis-type independent SCF / quantum chemistry

molsturm structure



- *Italic*: planned

molsturm design

- Enables `contraction`-based SCF routines
- Flexiblity as primary goal
- Behaviour controlled via `python`
  - Keywords to change basis type or solver
  - All computed data available in `numpy` format
  - No input file, just a `python` script
- `python` utilities
  - Import / export results
  - Post-HF calculations

Demo

DEMO

of using molsturm

Contents

1. Hartree-Fock
   - Discretisation of the HF equations
2. Coulomb-Sturmian basis sets
   - Coulomb Sturmians (CS)
   - Need for a `contraction`-based scheme
3. Lazy matrices in quantum chemistry
   - `contraction`-based algorithms
   - The `lazyten` lazy matrix library
   - `molsturm` program package
4. Future work
   - Outlook

Investigate the use of Coulomb-Sturmians

- CS show correct long-range behaviour

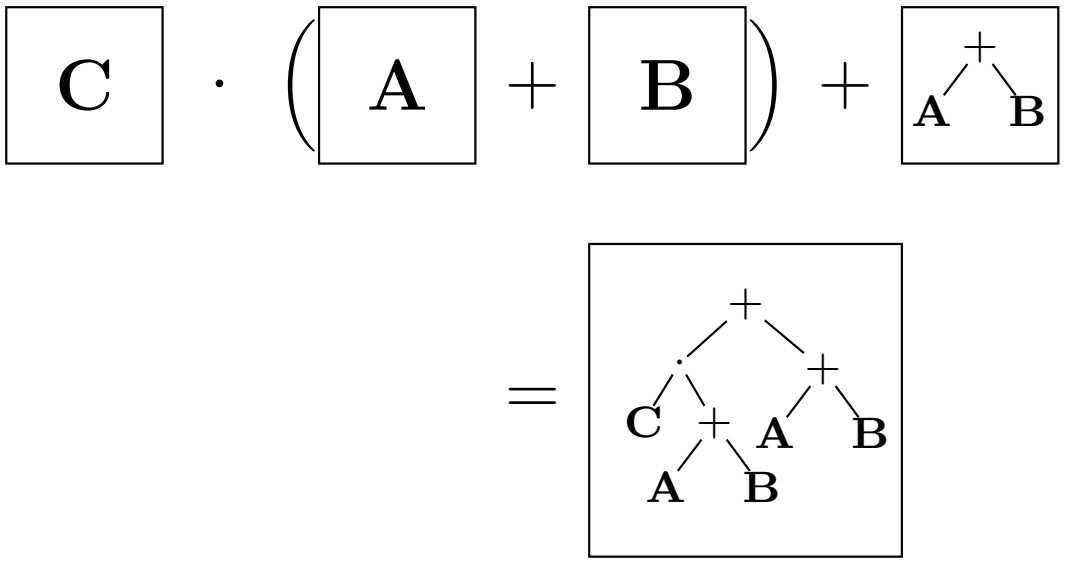
⇒ Smaller basis than GTOs needed?

⇒ Advantageous for describing density tail:

  - Rydberg states
  - Ionisation

- Improved description of correlation?

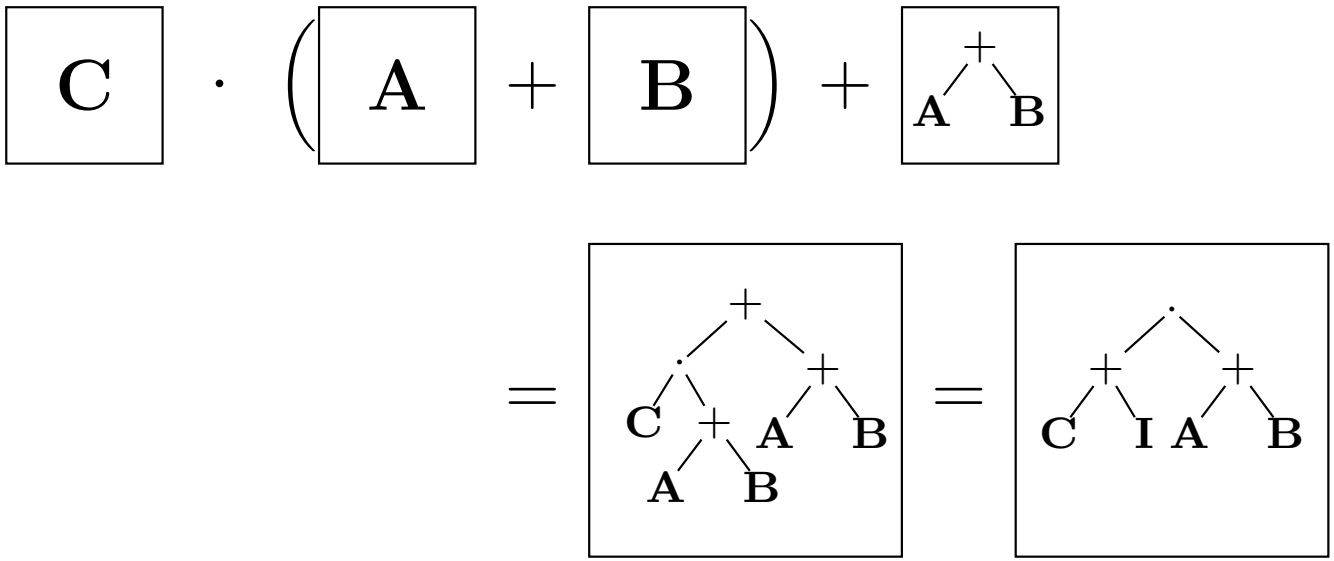
Lazy matrix expression optimisation



- Matrix expression tree ≡ abstract syntax tree

⇒ May be optimised by standard methods

Lazy matrix expression optimisation



- Matrix expression tree ≡ abstract syntax tree

⇒ May be optimised by standard methods

Selection of LA backend

- Right now: LA backend is compiled in
- e.g. Bohrium backend
  - Uses just-in-time (JIT) compilation
  - Very specific for hardware
  - Compilation takes time
- Better: Dynamic selection
  - Load on the expression tree
  - Availablility of backends
  - Hardware specs

Extension to lazy tensors

- `contraction` is a special tensor contraction

⇒ Lazy tensors:

- Delay all tensor contractions as long as possible

$$\text{e.g.} \qquad \tilde{k}_{bf} = \sum_{acdo\mu\nu} \mathcal{C}_{ao} \; c^{\mu}_{ab} I_{\mu\nu} c^{\nu}_{cd} \; \mathcal{C}_{co}\mathcal{C}_{df}$$

- Compare possible contraction schemes by complexity
- Execute cheapest evaluation scheme
- May incorporate and exploit symmetry

⇒ Determine optimal contraction sequence automatically

Acknowledgements

- Dr. James Avery
- Prof. Andreas Dreuw and the Dreuw group

- Prof. Guido Kanschat
- HGS Mathcomp

Questions?

- EMail: `michael.herbst@iwr.uni-heidelberg.de`
- Website/blog: `https://michael-herbst.com`
- Projects: `https://lazyten.org` and `https://molsturm.org`