Pushing the boundaries of SIESTA: Accelerated and massively parallel solvers

Alberto García (ICMAB-CSIC, Barcelona)

MAX CoE/ENCCS Workshop on Efficient Materials Modelling on HPC with Quantum ESPRESSO, Siesta and Yambo
March 11-15, 2024

The basic core of SIESTA

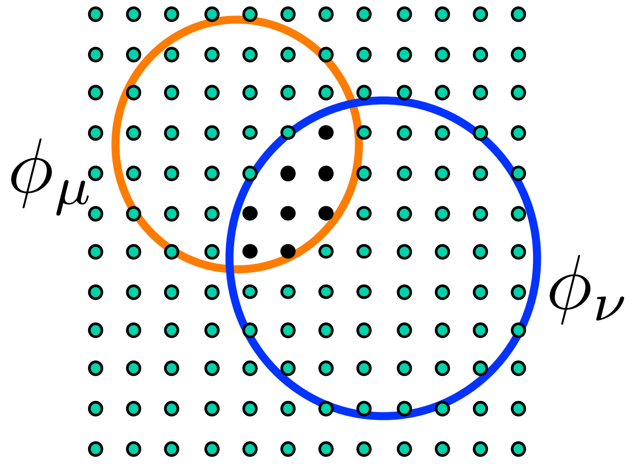
$$\psi_i(\boldsymbol{r}) = \sum_\mu \phi_\mu(\boldsymbol{r}) c_{\mu i},$$

$$\sum_{\nu\beta} (H_{\mu\nu}^{\alpha\beta} - E_i S_{\mu\nu} \delta^{\alpha\beta}) c_{\nu i}^{\beta} = 0$$

Generalized eigenvalue problem

$$S_{\mu\nu} = \langle \phi_\mu | \phi_\nu \rangle$$

$$\rho(\boldsymbol{r}) = \sum_{\mu\nu} \rho_{\mu\nu} \phi_\nu^*(\boldsymbol{r}) \phi_\mu(\boldsymbol{r})$$



$$\boxed{\rho_{\mu\nu} = \sum_i c_{\mu i} n_i c_{i\nu}}$$

Density matrix

The SOLVER step takes most of the CPU time

MAX DRIVING THE EXASCALE TRANSITION

Diagonalization-based solvers

Need to use DIRECT solvers, as the number of desired eigenvectors is a substantial fraction of the matrix size

$$\sum_{\nu\beta} (H^{\alpha\beta}_{\mu\nu} - E_i S_{\mu\nu} \delta^{\alpha\beta}) c^{\beta}_{\nu i} = 0$$

SIESTA uses pre-packaged libraries for this pure math problem:

- ScaLaPACK
  - pdsyev, pzheev and related drivers
  - MRRR
- ELPA: Alternative transformation sequence + optimizations
  https://elpa.mpcdf.mpg.de/

- Conversion of H and S to dense form
- Cholesky decomposition to reduce to standard eigenproblem
- Transformation to tri-diagonal form
- Solution of tri-diagonal problem
- Back-transformation

Cubic scaling with matrix size — Quadratic scaling for memory

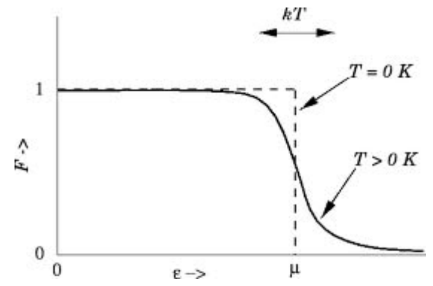
Still competitive for low-cardinality basis sets

Direct solution for the density matrix

$$\hat{\rho} = f_\beta(\hat{H} - \mu)$$

$$f_\beta(\epsilon_i - \mu) = \frac{2}{1 + \mathrm{e}^{\beta(\epsilon_i - \mu)}}$$

Fermi-Dirac function



Fermi Operator Expansion (FOE)

$$p(H) = \frac{c_0}{2} I + \sum_{j=1}^{n_{pl}} c_j T_j(H)$$

Calculation of the DM involves only (sparse) matrix-vector multiplications

CheSS library
(originally in BigDFT)

Linear-scaling

Stephan Mohr (BSC)

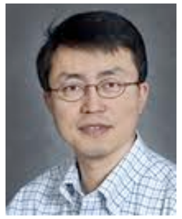
- Number of terms in the expansion can be large
- Efficiency increases for contracted basis sets.
- Exploring on-the-fly contraction

MAX DRIVING THE EXASCALE TRANSITION

Direct solution for the density matrix

PEXSI: Pole Expansion plus Selected Inversion

(Lin Lin, Chao Yang, et al., Berkeley)



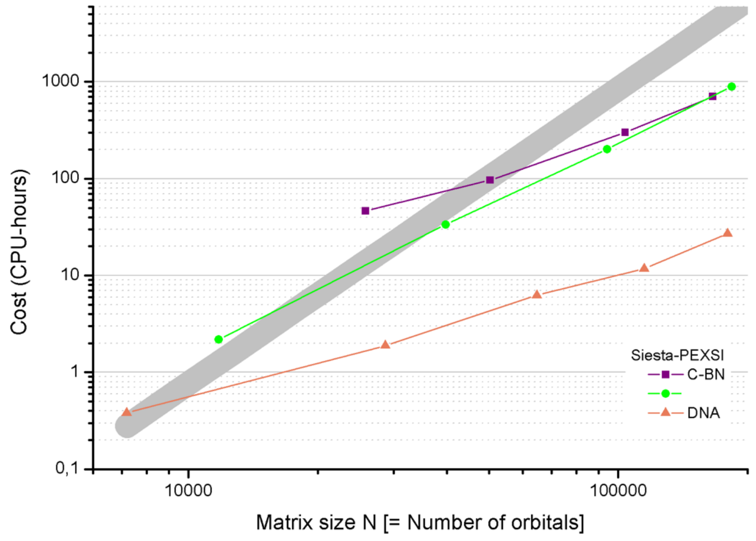
$$\hat{\rho} = Im\left(\sum_{l=1}^{P} \frac{\omega_l}{H - (z_l + \mu)S}\right)$$

For sufficiently big problems

(quasi-)1D:	$\mathcal{O}(N)$
(quasi-)2D:	$\mathcal{O}(N^{3/2})$
3D:	$\mathcal{O}(N^2)$

(Due to sparsity of the target density matrix)

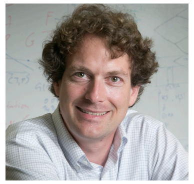
Relatively small number of poles (20-30)
Trivially parallelizable over them



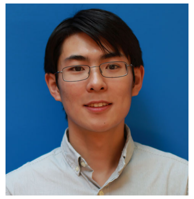
DRIVING THE EXASCALE TRANSITION

Solver strategies for performance and features: Use external libraries

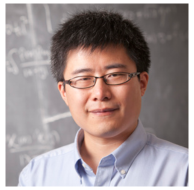
ELSI initiative to integrate solver libraries



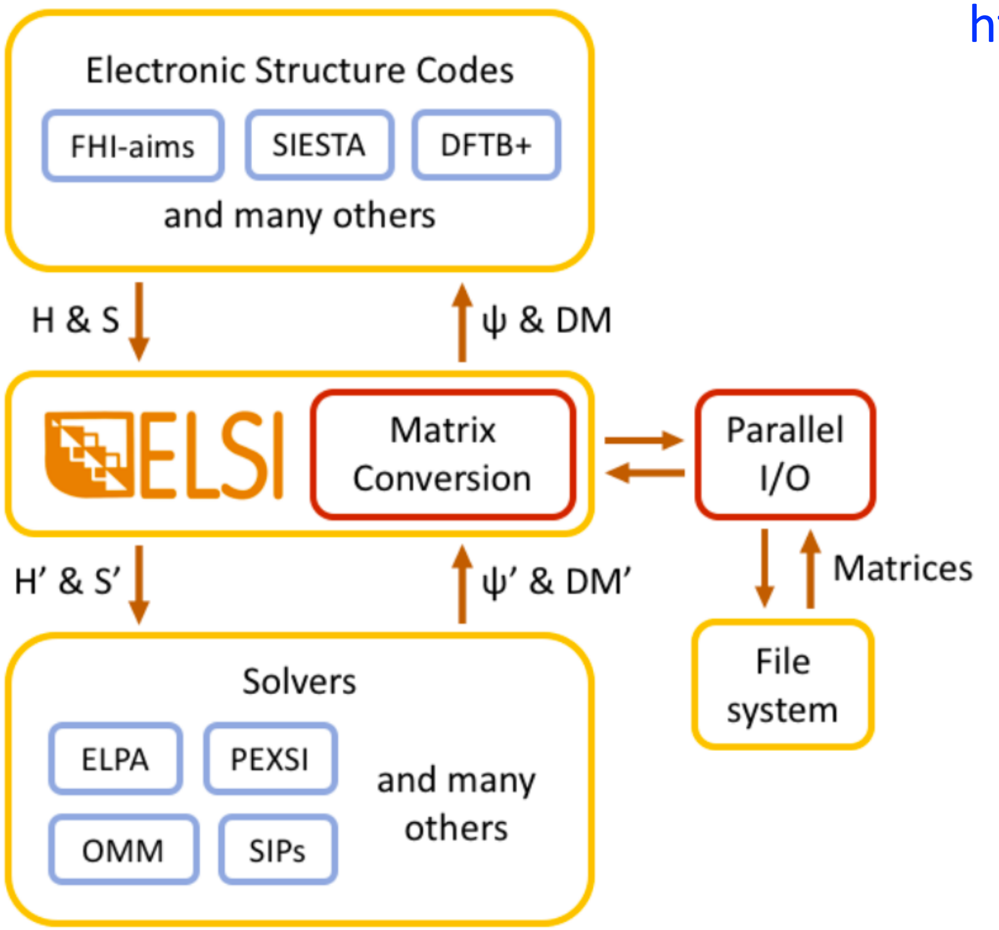
Volker Blum, Duke



Lin Lin, Berkeley



Jiangfen Lu, Duke



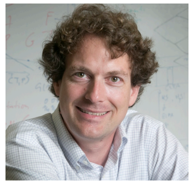
https://elsi-interchange.org

Interface in Siesta:

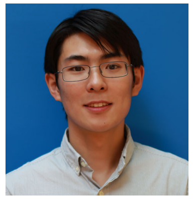
Collaboration with Victor Yu (Duke)

Solver strategies for performance and features: Use external libraries

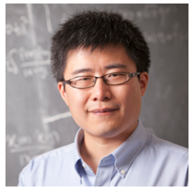
ELSI initiative to integrate solver libraries



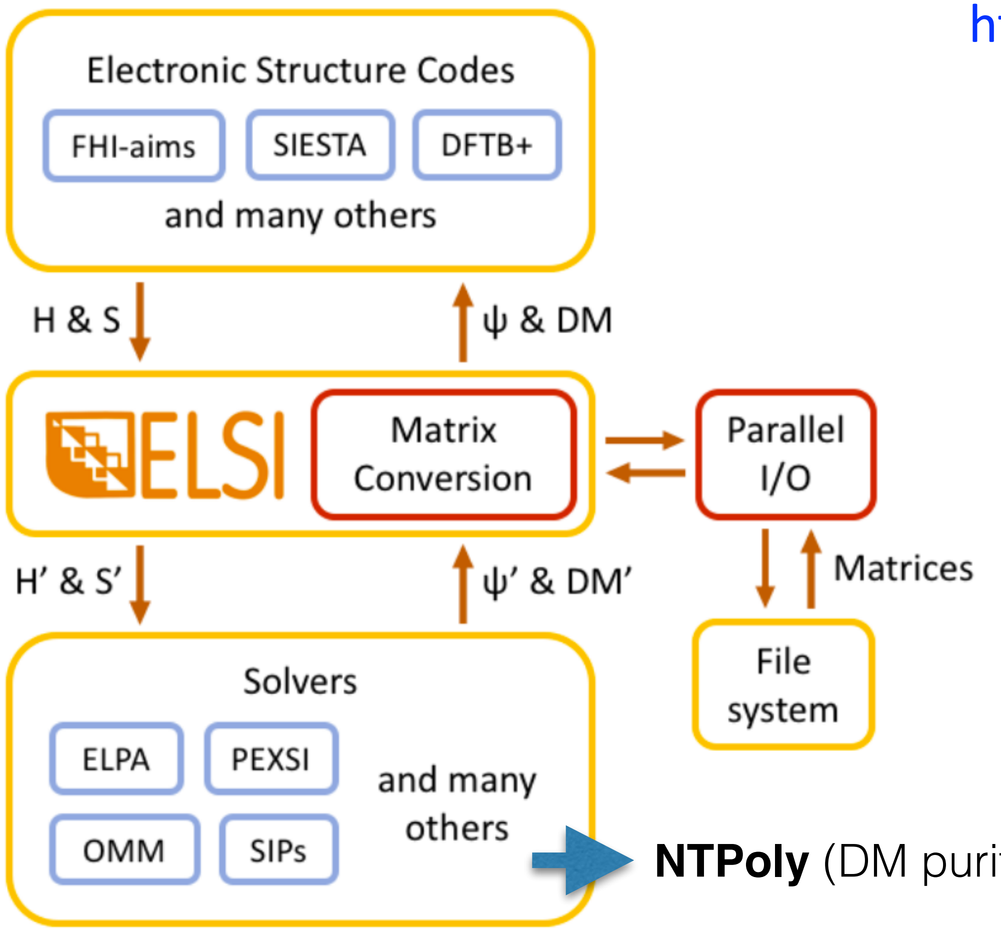
Volker Blum, Duke



Lin Lin, Berkeley



Jiangfen Lu, Duke



NTPoly (DM purification, O(N))

https://elsi-interchange.org

Interface in Siesta:

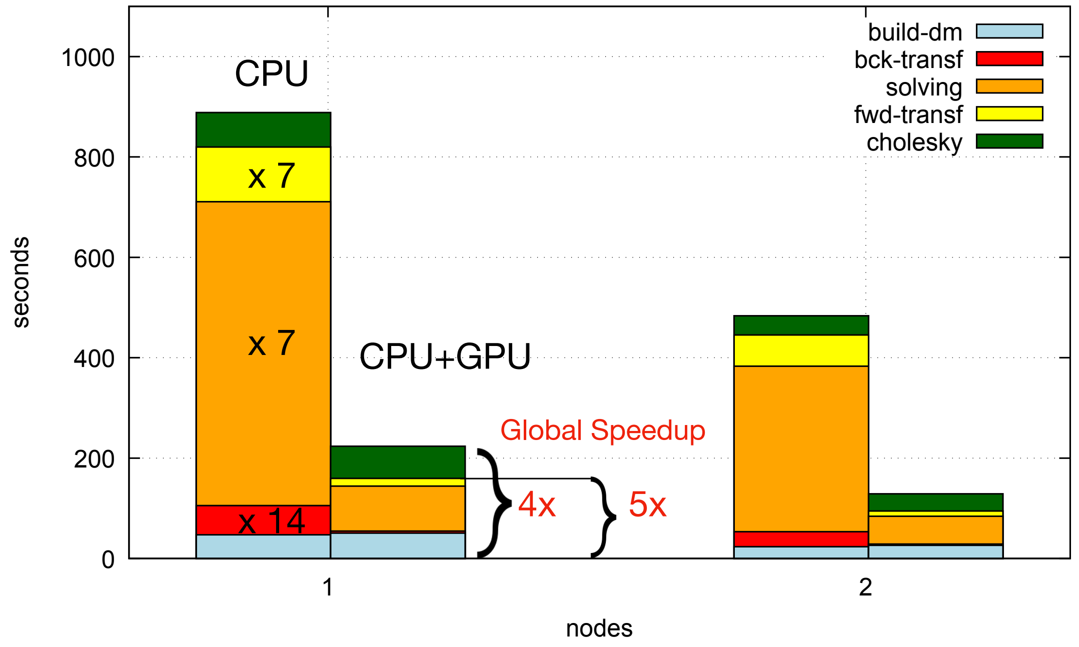
Collaboration with Victor Yu (Duke)

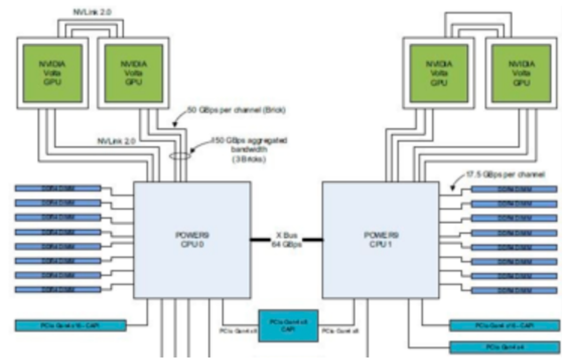
MAX DRIVING THE EXASCALE TRANSITION

GPU acceleration for diagonalization

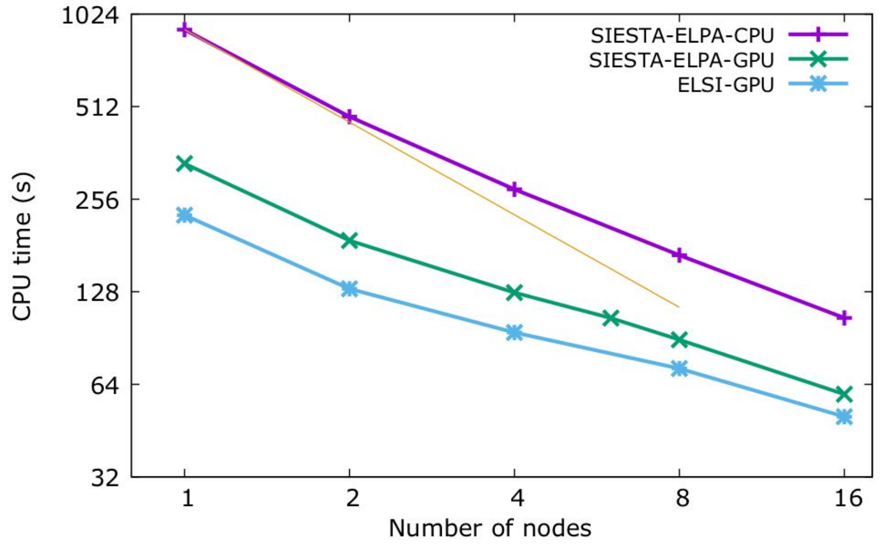
Marconi-100 (CINECA): 32 CPUs+ 4 GPUs /node

ELSI-ELPA GPU acceleration





Proper binding of GPUs to MPI ranks

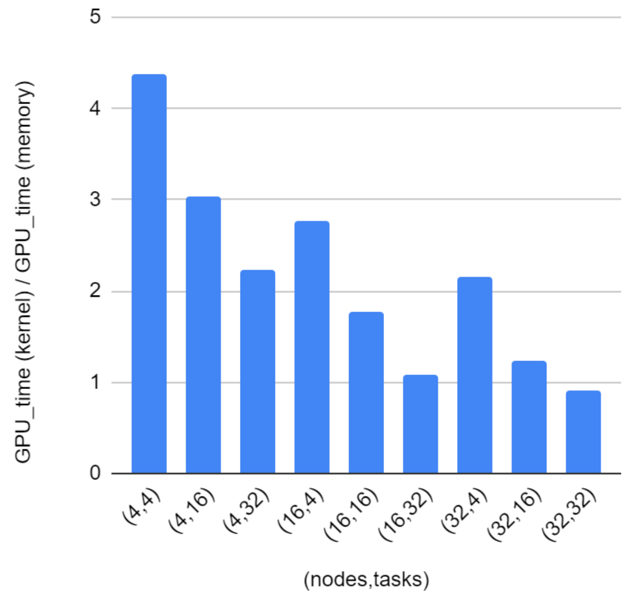


Future enhancements in ELPA (better kernels) and in ELSI (e.g. build-DM stage) are integrated in SIESTA automatically

System: Si quantum dot, with approx 35000 orbs

Further improvements in GPU offloading

- Exploration of MPI-ranks — GPU balance in Leonardo:
  - Use of MPS framework (**collaboration with CINECA**)
- Incorporation of the latest **ELPA developments**:
  - New GPU offloading of Cholesky step
  - GPU offloading of 1-stage operations
  - Improvements in porting to AMD and Intel GPUs
- Exploration of ChASE library



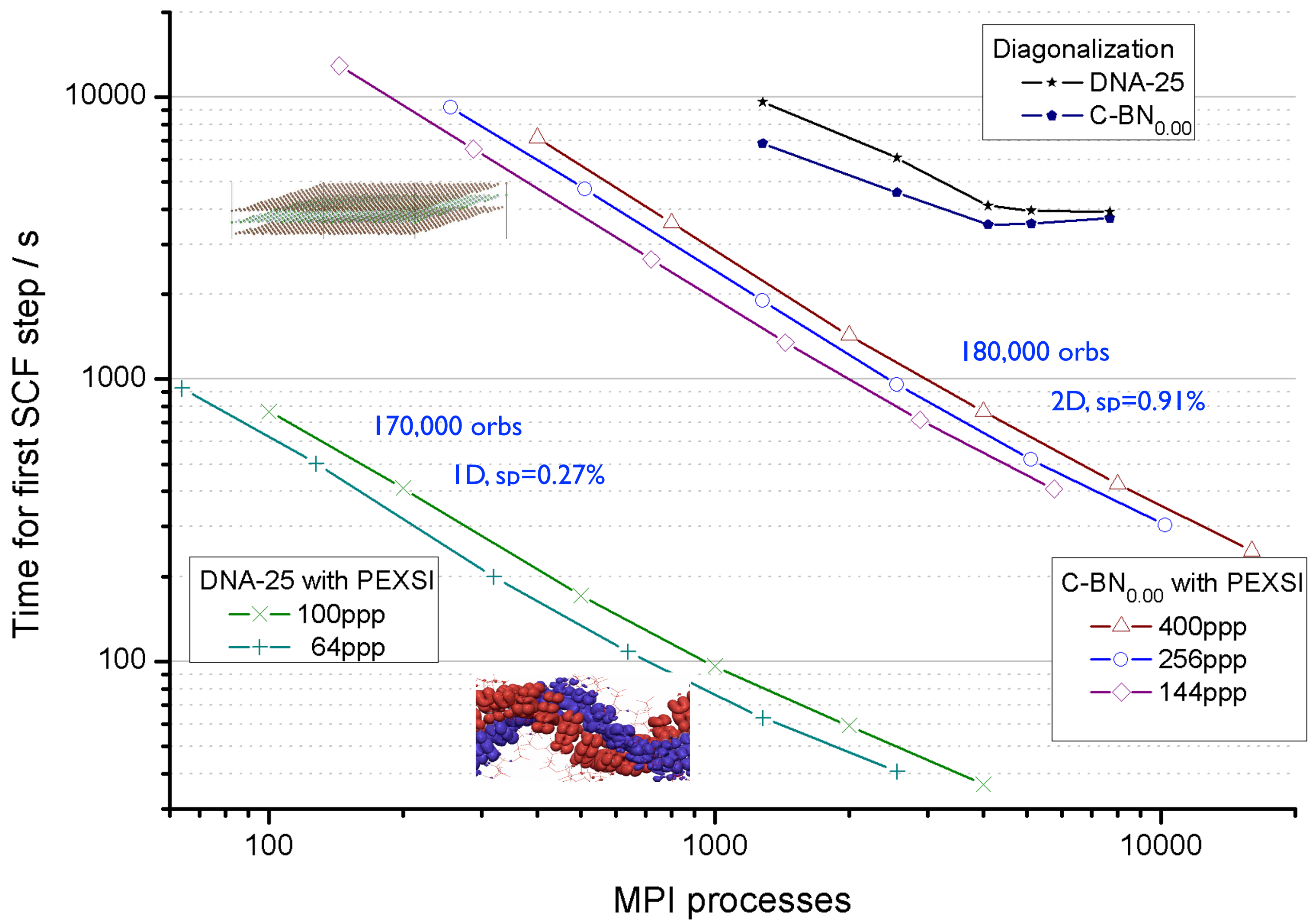
DRIVING THE EXASCALE TRANSITION

Massive scalability: PEXSI solver

$$\hat{\rho} = Im\left(\sum_{l=1}^{P} \frac{\omega_l}{H - (z_l + \mu)S}\right)$$

PEXSI offers:

- Three levels of parallelization (over orbitals, poles, and chemical potential values)
- A reduced memory footprint (only sparse matrices are stored)
- Reduced complexity (maximum O(N^2) size scaling)

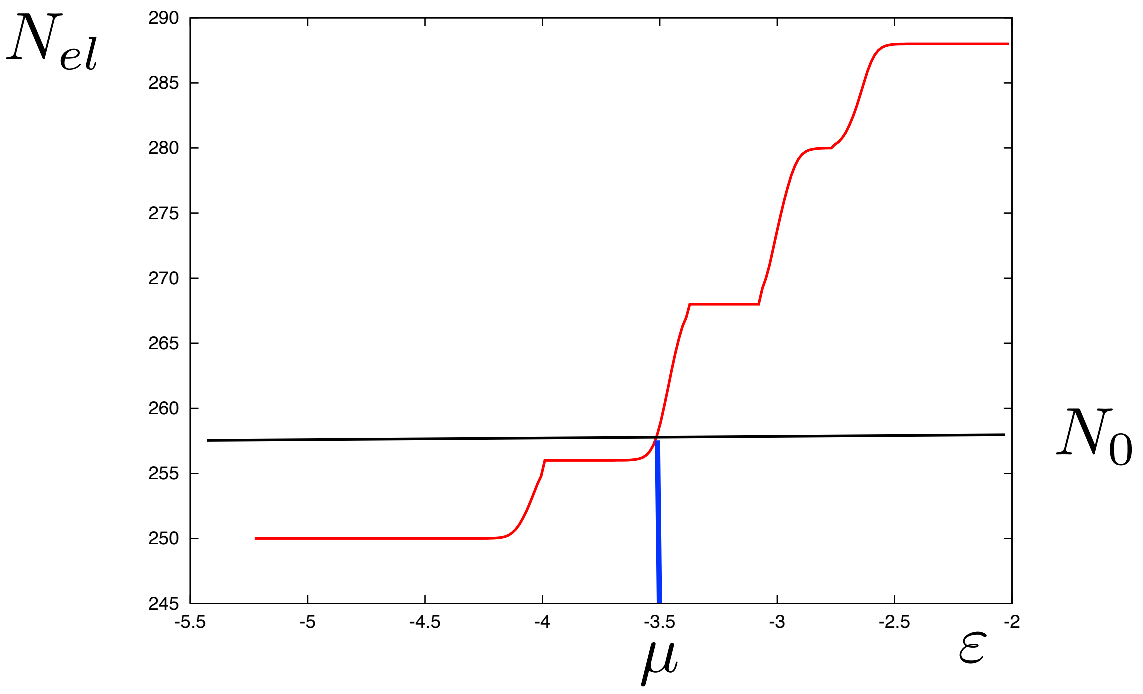


Determination of the chemical potential in PEXSI

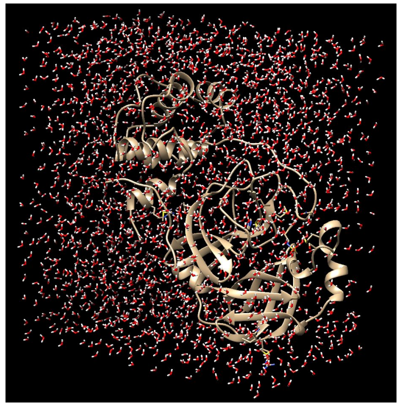
The shifted hamiltonian $H - \sigma S = LDL^T$

has the same number of negative eigenvalues as the diagonal matrix D in its factorization ("law of preservation of inertia").

One can thus get the "integrated DOS" by computing the inertia counts for various shifts, and μ by interpolation.

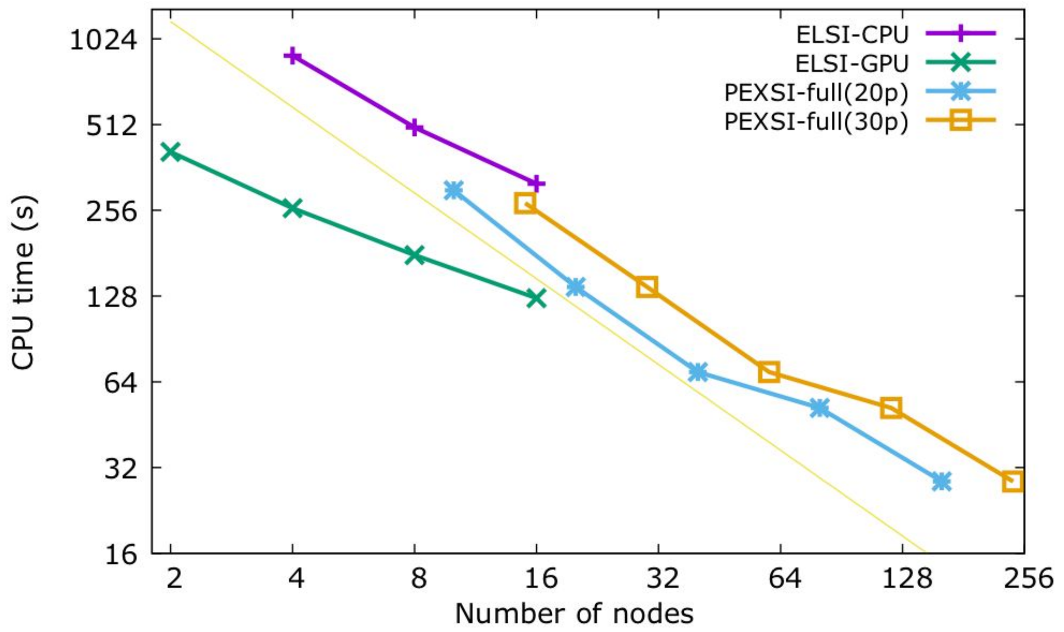


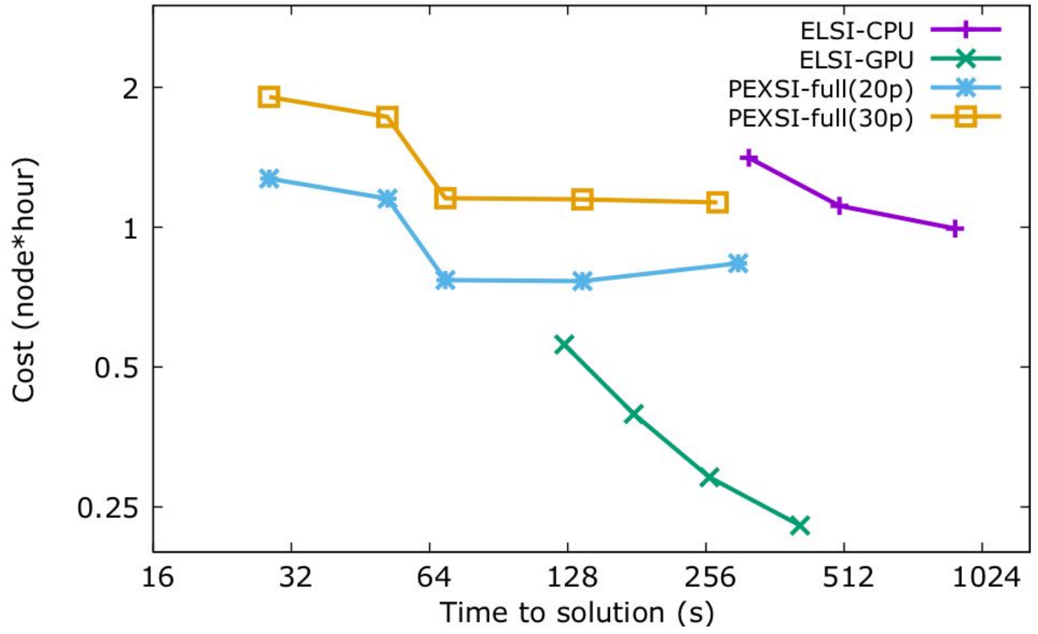
Comparison of global efficiency of solvers for a very large problem



SARS CoV-2 M^{pro} with solvation water molecules

Approx 8800 atoms; 58000 orbitals





There is as yet no GPU acceleration of the PEXSI library

DRIVING THE EXASCALE TRANSITION

Other Solvers in SIESTA

- Original linear-scaling solver based on the (localized) Orbital Minimization Method: Re-designed to endow it with a modular sparse-matrix algebra backend.
- MAGMA, NTPoly, etc: Part of the ELSI library of solvers.
- Native interface to PEXSI: Uses a Newton method to search for the appropriate chemical potential. Uses a legacy pole-generation method.
- Native interface to ELPA: It needs to be updated to exploit the latest library enhancements.

PRACTICAL EXERCISES

/leonardo_work/EUHPC_TD02_030/siesta-tutorials/day4-Thu/03-SiestaSolvers

SPECIAL NOTES FOR THIS SESSION

- A ‘master branch’ version of SIESTA is used (It includes the ELSI interface)
- This version is compiled with a Spack environment, and the setup is different (but automatic)
- The exercises involve running already prepared slurm scripts.

Example slurm script

```
#!/bin/bash
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=16
#SBATCH --cpus-per-task=2
#SBATCH --gres=gpu:4
#SBATCH --time=00:30:00
#SBATCH –exclusive
#SBATCH --account=EUHPC_TD02_030
#SBATCH --partition=boost_usr_prod
#SBATCH --qos=normal

# This example uses 2 nodes and 16 tasks per node, that is, 4 MPI tasks per GPU

. /leonardo/prod/opt/environment/module/current/init/profile.sh

module purge
module load /leonardo_work/EUHPC_TD02_030/softwares/siesta-master/siesta-module
export OMP_NUM_THREADS=1

export SIESTA_PS_PATH=/leonardo_work/EUHPC_TD02_030/siesta-solvers/Pseudos

mpirun -np 32 --map-by socket:PE=2 --rank-by core  \
                       siesta covid-12k-diag.fdf > covid-12k-diag.out
```

Execution

```
Leonardo> cd ELSI-ELPA/GPU
Leonardo> cd gpu-4mpi
Leonardo> sbatch submit.job
Leonardo> squeue -u $USER

…
```

Compilation of timing results

```
Leonardo> cd ELSI-ELPA                    # … must be in this directory, or in ELSI-PEXSI
Leonardo> sh analyze.sh GPU               # argument is a directory name

#  Nodes Ntasks     H0        DM    cholesky     to_std  solve_std       back    dm_calc     redist     nranks
 2      16       4.620    81.897      16.252      6.472     41.847      2.656      2.516     11.391         32
 1      16       7.425   139.761      32.334      9.034     64.907      4.390      4.302     23.460         16
 2      32       3.372    78.896      15.858      7.243     43.393      2.615      2.307      6.784         64
 2      4       11.988   141.475      31.657      7.317     51.225      3.212      4.302     42.401          8

Leonardo> sh analyze.sh CPU
#  Nodes Ntasks     H0        DM    cholesky     to_std  solve_std       back    dm_calc     redist     nranks
 2      32       4.919   214.400      16.052     12.581    152.226     11.341     14.530      6.776         64
```

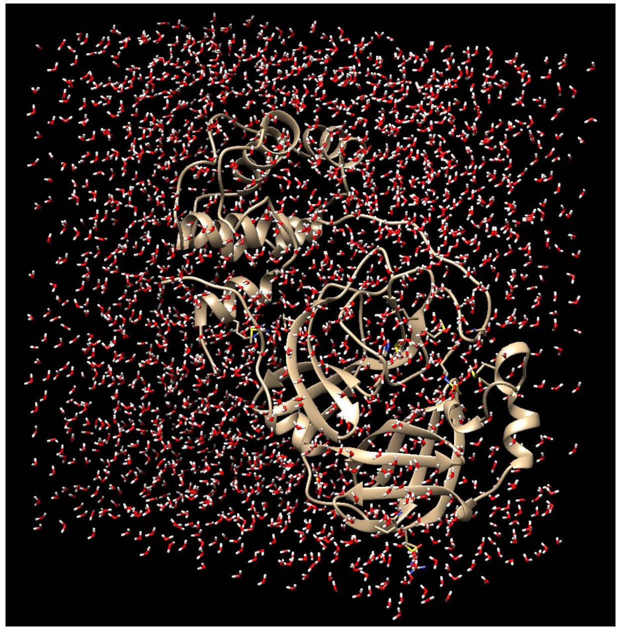
SYSTEM:

Piece of a virus spike protein in water

(Approximately 12000 atoms)

Minimal basis set (sz)

(approx 26400 orbitals)



FDF flags for ELSI-ELPA operation

```
solution-method elsi
elsi-solver elpa

elsi-elpa-flavor 2                # 1: One-pass conversion to tri-diagonal matrix
                                  # 2: Two-pass: first banded, then tri-diagonal

elsi-elpa-gpu 1                   # Whether to use GPUs (1) or not (0)

elsi-output-level 3               # log level
use-tree-timer T                  # hierarchical timer (SIESTA general feature)


number-of-eigenstates -500        # Compute up to 500 states above the Fermi level
                                  # (not just the occupied states – useful for
                                  #  smearing when needed)

### number-of-eigenstates 3456    # Example of request of specific number of states
```

ELSI-ELPA hands-on practice

```
.
├── CPU
│   ├── covid-12k-diag.fdf
│   ├── covid+H_clean+h2o.coor
│   └── submit.job
├── GPU
│   ├── gpu-1mpi
│   │   ├── covid-12k-diag.fdf
│   │   ├── covid+H_clean+h2o.coor
│   │   └── submit.job
│   ├── gpu-1node
│   │   ├── covid-12k-diag.fdf
│   │   ├── covid+H_clean+h2o.coor
│   │   ├── README
│   │   └── submit.job
│   ├── gpu-4mpi
│   │   ├── covid-12k-diag.fdf
│   │   ├── covid+H_clean+h2o.coor
│   │   └── submit.job
│   ├── gpu-8mpi
│   │   ├── covid-12k-diag.fdf
│   │   ├── covid+H_clean+h2o.coor
│   │   └── submit.job
│   ├── gpu-bs32
│   │   ├── covid-12k-diag.fdf
│   │   ├── covid+H_clean+h2o.coor
│   │   └── submit.job
│   ├── gpu-bs40
│   │   ├── covid-12k-diag.fdf
│   │   ├── covid+H_clean+h2o.coor
│   │   └── submit.job
│   └── README
└── README
```

Minimal suggested practice:
- Run CPU-only example
- Run GPU/gpu-32mpi and GPU/gpu-4mpi

to compare the ‘solve_std’ columns

. . .

FDF flags for ELSI-PEXSI operation

```
solution-method elsi
elsi-solver pexsi

elsi-pexsi-method 3                 # Method to generate the poles (AAA)
elsi-pexsi-tasks-per-pole 4         # Number of MPI ranks per pole (orb dist)
elsi-pexsi-number-of-poles 20
elsi-pexsi-number-of-mu-points 2    # Number of independent calculations
                                    # to find the fermi level by interpolation

elsi-pexsi-initial-mu-min -10.0 eV  # Initial bracketing of fermi level
elsi-pexsi-initial-mu-max 0.0 eV

elsi-output-level 3

DM.NormalizationTolerance 1.0e-3    # Tolerance level for number of electrons
```

ELSI-PEXSI hands-on practice

```
.
├── README
├── tpp-16
│   ├── covid-12k-pexsi.fdf
│   ├── covid+H_clean+h2o.coor
│   └── submit.job
├── tpp-32
│   ├── covid-12k-pexsi.fdf
│   ├── covid+H_clean+h2o.coor
│   └── submit.job
├── tpp-4
│   ├── covid-12k-pexsi.fdf
│   ├── covid+H_clean+h2o.coor
│   ├── submit.job
└── tpp-8
    ├── covid-12k-pexsi.fdf
    ├── covid+H_clean+h2o.coor
    └── submit.job
```

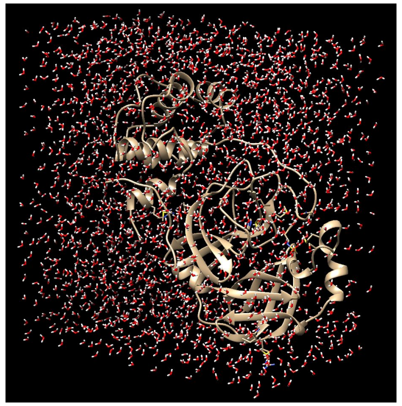
Minimal suggested practice:

- Run tpp-4 and tpp-8 (moderate number of nodes requested)

Optional: Change the number of nodes in tpp-8/submit.job to just 1. This will process batches of poles sequentially, running 4 pole-teams at a given time, until the 40 poles-points are done. The calculation will fit in a single node (but will take 10 times longer)

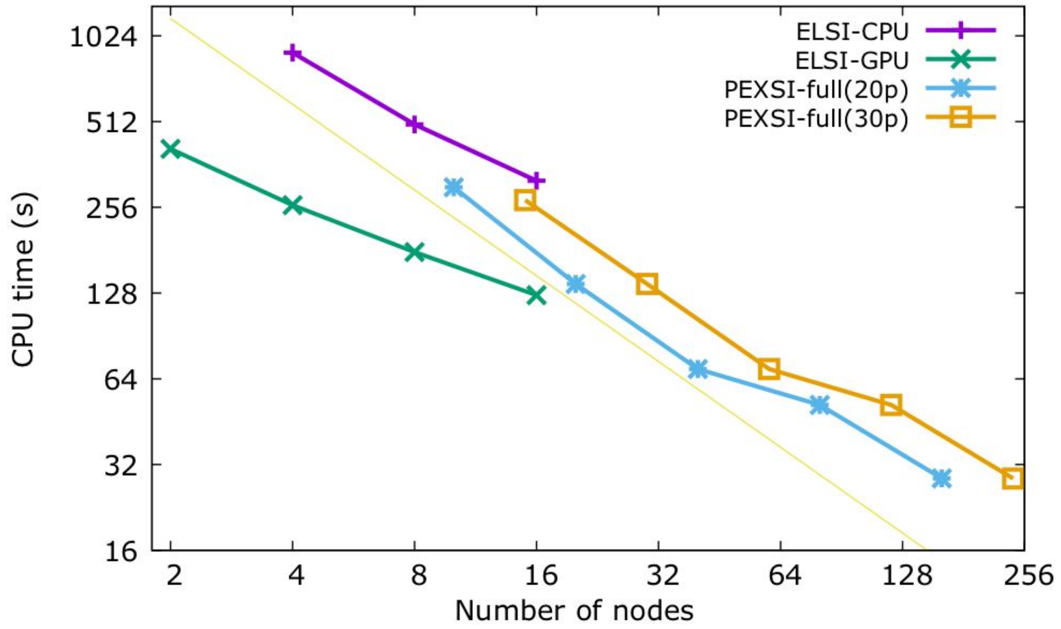
. . .

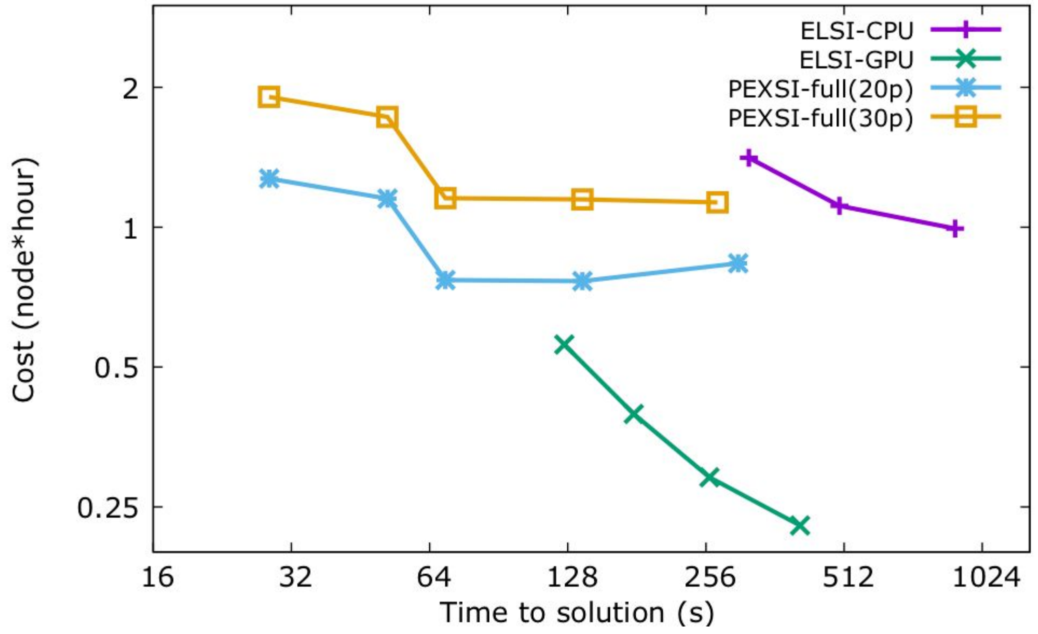
Recap on solver capabilities for a very large problem



SARS CoV-2 M^{pro} with solvation water molecules

Approx 8800 atoms; 58000 orbitals (szp)





There is as yet no GPU acceleration of the PEXSI library

THANKS