Fast Quantum Molecular Dynamics on Multi-GPU Architectures in LATTE

S. Mniszewski*, M. Cawkwell, A. Niklasson

GPU Technology Conference

San Jose, California

March 18-21, 2013

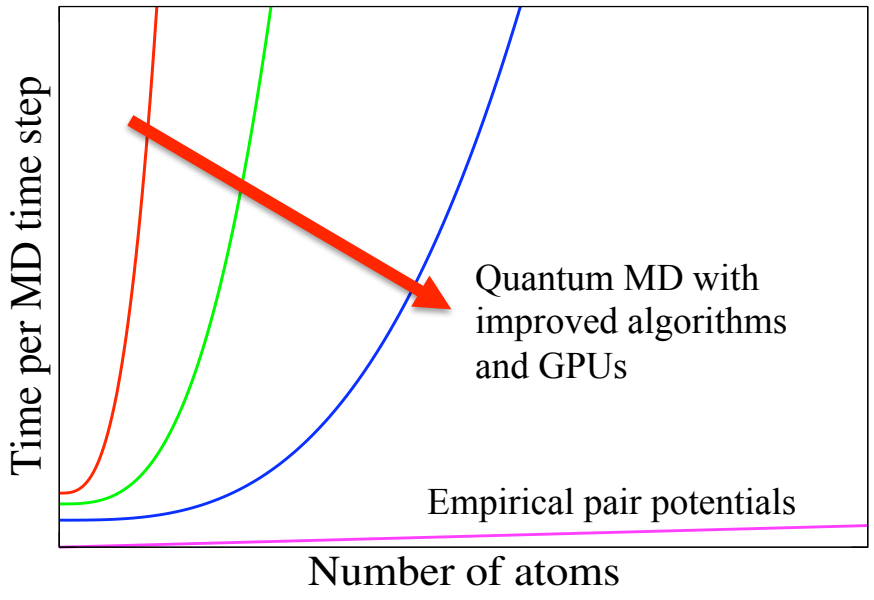
***smm@lanl.gov**

Background: Quantum Molecular Dynamics

- In molecular dynamics simulation, the relative positions of atoms evolve over a series of time steps according to the force acting on each atom
- Employed in materials science, chemistry, and biology to study structures, defects, and equilibrium and non-equilibrium phenomena
- Dependence on an interatomic potential to calculate forces and energy
- Quantum-based models capture the making and breaking of covalent bonds, charge transfer between species of differing electronegativities, and long-range electrostatic interactions

Quantum-based Interatomic Potentials

- **Electronic structure of atoms and molecules is modeled explicitly**
- **Most accurate and reliable descriptions of interatomic bonding**
- **Their prohibitive computational cost has prevented widespread use – better algorithms and GPU architectures are important paths forward**



- **Hamiltonian matrix** H
- **The density matrix, rho, is computed self-consistently from** H

Energy $E = 2\mathrm{Tr}\left[\rho H\right]$

Force $\mathbf{f}_i = -2\mathrm{Tr}\left[\rho \dfrac{\partial H}{\partial \mathbf{R}_i}\right]$

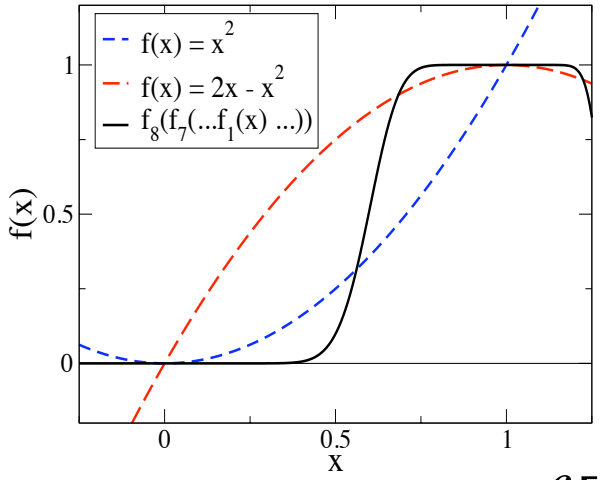
The Density Matrix Computation

- **Typically, algorithms used in quantum-based models, most notably matrix diagonalization, are not ideally suited to GPUs**
  - Due to their complexity
  - Difficulty in extracting thread-level parallelism
  - Difficulty of avoiding branching within warps
- **New approach in LATTE**
  - Computed directly from the Hamiltonian through a recursive expansion of the Fermi Operator with the second order spectral projection (SP2) algorithm
  - Based on a series of generalized matrix-matrix multiplications
  - Only one matrix-matrix multiplication is required per iteration
  - Maps very well to GPUs

The Second Order Spectral Projection Algorithm (SP2) – Reduced Complexity

Recursive Fermi Operator expansion

$$\rho = \theta\left[\mu\mathbf{I} - \mathbf{H}\right] = \lim_{i\to\infty} f_i[f_{i-1}[\ldots f_0[\mathbf{X}_0]\ldots]]$$



$$\mathbf{X}_0 = \frac{\varepsilon_{\max}\mathbf{I} - \mathbf{H}}{\varepsilon_{\max} - \varepsilon_{\min}}$$

$$f_i[\mathbf{X}_i] = \begin{cases} \mathbf{X}_i^2 & \text{if } 2\mathrm{Tr}[\mathbf{X}_i] \geq N \\ 2\mathbf{X}_i - \mathbf{X}_i^2 & \text{if } 2\mathrm{Tr}[\mathbf{X}_i] < N_e \end{cases}$$

The GPU Implementation

- **Part of the LATTE codebase**
  - Employs a semi-empirical tight-binding model of interatomic bonding that is based on the formalisms derived from density functional theory
  - Density matrix build is by far the slowest step in the calculation
  - CPU version in Fortran 90
- **Hardware/Software Architecture**
  - Keeneland* cluster at the National Institute for Computational Sciences
  - CPU - 2 Intel hex-core Xeon CPUs per node, Intel Fortran Compiler, MKL
  - 3 Nvidia M2090 GPUs (previously M2070 GPUs)
  - CUDA 4.2, CUBLAS, and a thread block size of 1024
- **Use of CUDA Features on GPUs**
  - Unified Virtual Addressing
  - Peer to peer memory access/copy
  - Streams –sequence of commands
  - Single thread access to all GPUs

*J.S. Vetter, R. Glassbrook, J. Dongarra, K. Schwan, B. Loftis, S. McNally, J. Meredith, J. Rogers, P. Roth, K. Spafford, and S. Yalamanchili, "Keeneland: Bringing heterogeneous GPU computing to the computational science community," IEEE Computing in Science and Engineering, 13(5):90-5, 2011, http://dx.doi.org/10.1109/MCSE.2011.83.

SP2 Algorithm Using the Hybrid CPU/GPU Approach

Estimate ε_{max} and ε_{min}

$\mathbf{X} = (\varepsilon_{max}\mathbf{I}\text{-}\mathbf{H})/(\varepsilon_{max}\text{-}\varepsilon_{min})$

TraceX = Tr[**X**] /* Trace kernel on GPU */

Until converged do

$\mathbf{X}_{tmp} = \mathbf{X}$

$\mathbf{X}_{tmp} = \mathbf{X}^2+\mathbf{X}_{tmp}$ /*CUBLAS xGEMM */

$TraceX_{tmp} = Tr[\mathbf{X}_{tmp}]$ /*Trace kernel on GPU */

if $|2TraceX - 2TraceX_{tmp} - N_e| > |2TraceX + 2TraceX_{tmp} - N_e|$

$\mathbf{X} = \mathbf{X} + \mathbf{X}tmp$

$TraceX = TraceX + TraceX_{tmp}$

else

$\mathbf{X} = \mathbf{X} - \mathbf{X}_{tmp}$

$TraceX = TraceX - TraceX_{tmp}$

end until

$\rho = \mathbf{X}$

SP2 Algorithm Using the Full GPU Approach

Estimate ε_{max} and ε_{min}

$\mathbf{X} = (\varepsilon_{max}\mathbf{I}-\mathbf{H})/(\varepsilon_{max}-\varepsilon_{min})$

TraceX = Tr[**X**] /* Trace kernel on GPU */

Until converged do

$\mathbf{X}_{tmp} = \mathbf{X}$

$\mathbf{X}_{tmp} = \mathbf{X}^2+\mathbf{X}_{tmp}$ /*CUBLAS xGEMM */

$TraceX_{tmp} = Tr[\mathbf{X}_{tmp}]$ /*Trace kernel on GPU */

if $|2TraceX - 2TraceX_{tmp} - N_e| > |2TraceX + 2TraceX_{tmp} - N_e|$

$\mathbf{X} = \mathbf{X} + \mathbf{X}_{tmp}$ /* CUBLAS xAXPY */

$TraceX = TraceX + TraceX_{tmp}$ /* CUBLAS xAXPY */

else

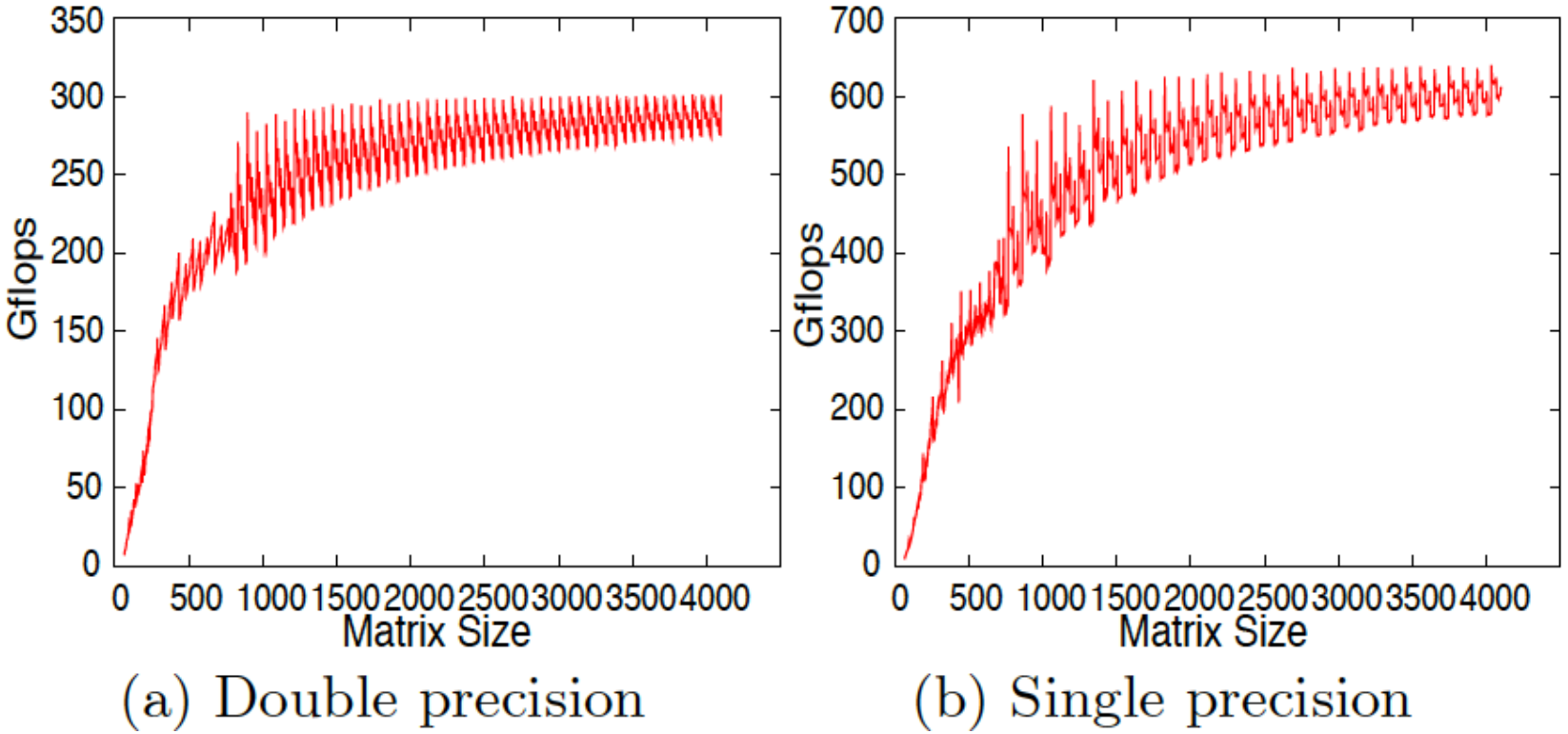
$\mathbf{X} = \mathbf{X} - \mathbf{X}_{tmp}$ /* CUBLAS xAXPY */

$TraceX = TraceX - TraceX_{tmp}$

end until

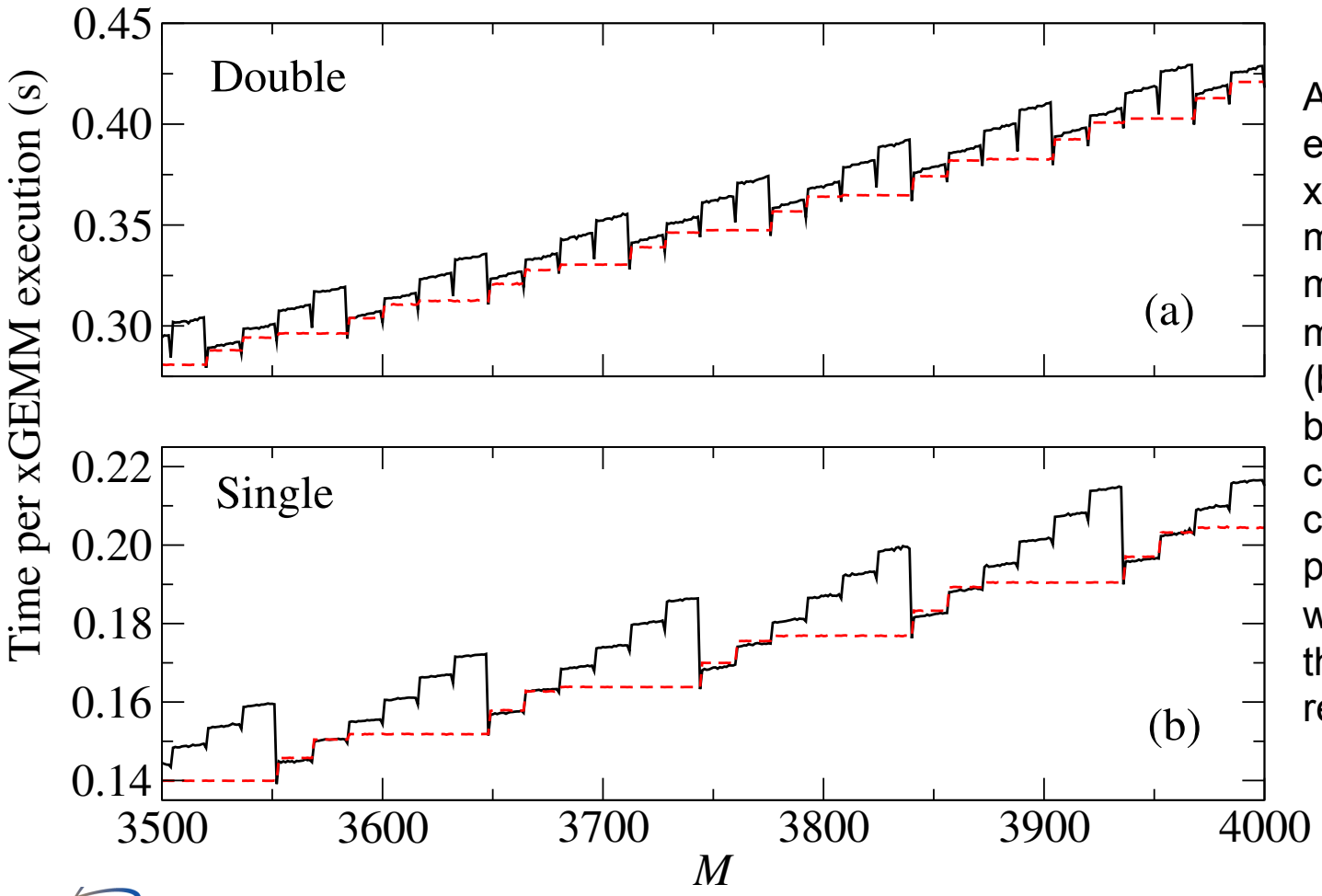
$\rho = \mathbf{X}$

CUBLAS Matrix Multiplication Performance (Nvidia M2070)



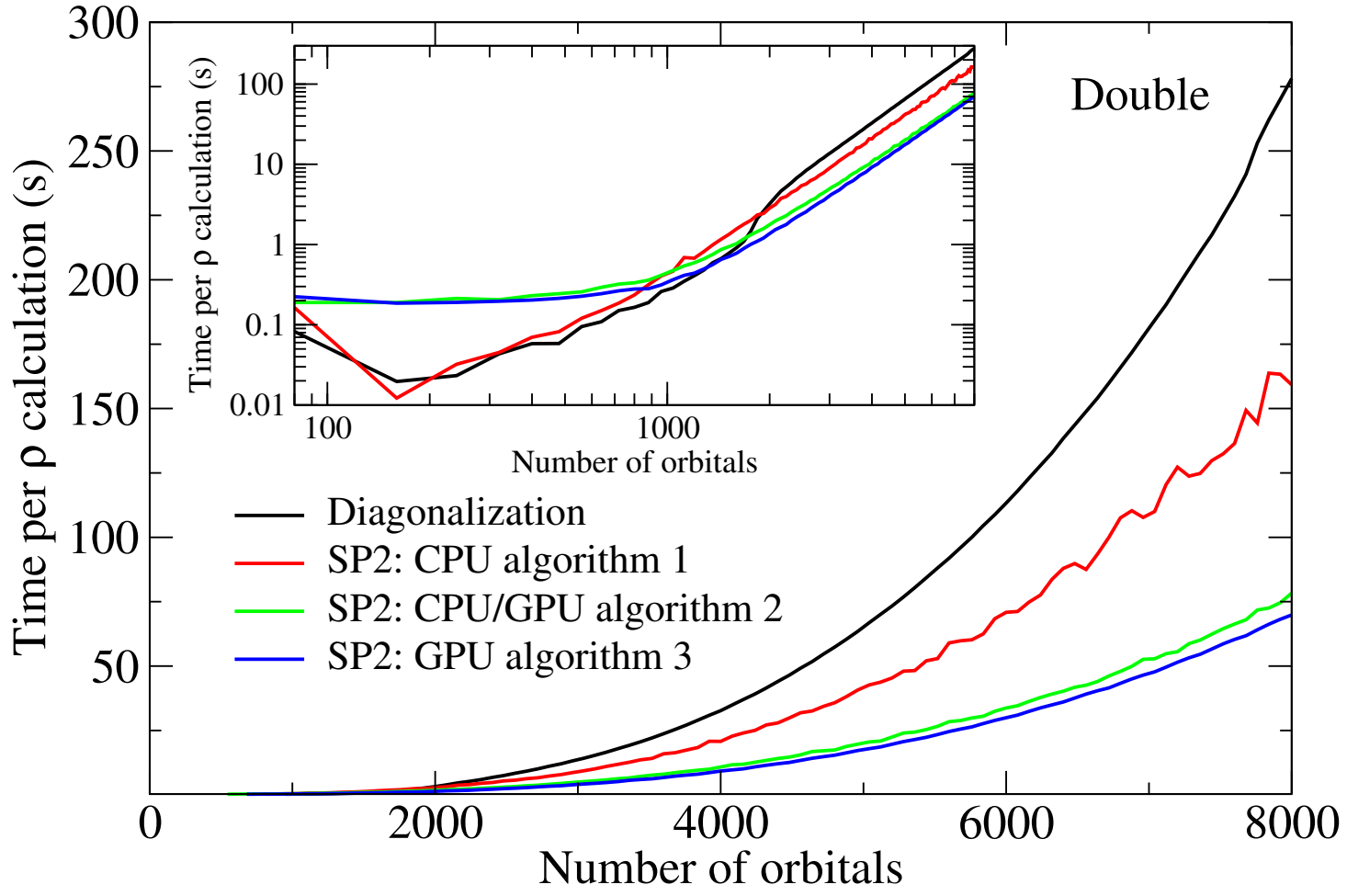
Song, F., Tomov, S., Dongarra, J. "Enabling and Scaling Matrix Computations on Heterogeneous Multi-Core and Multi-GPU Systems," 26th ACM International Conference on Supercomputing (ICS 2012), ACM, San Servolo Island, Venice, Italy, June, 2012.

Array Padding for Performance (M x M)

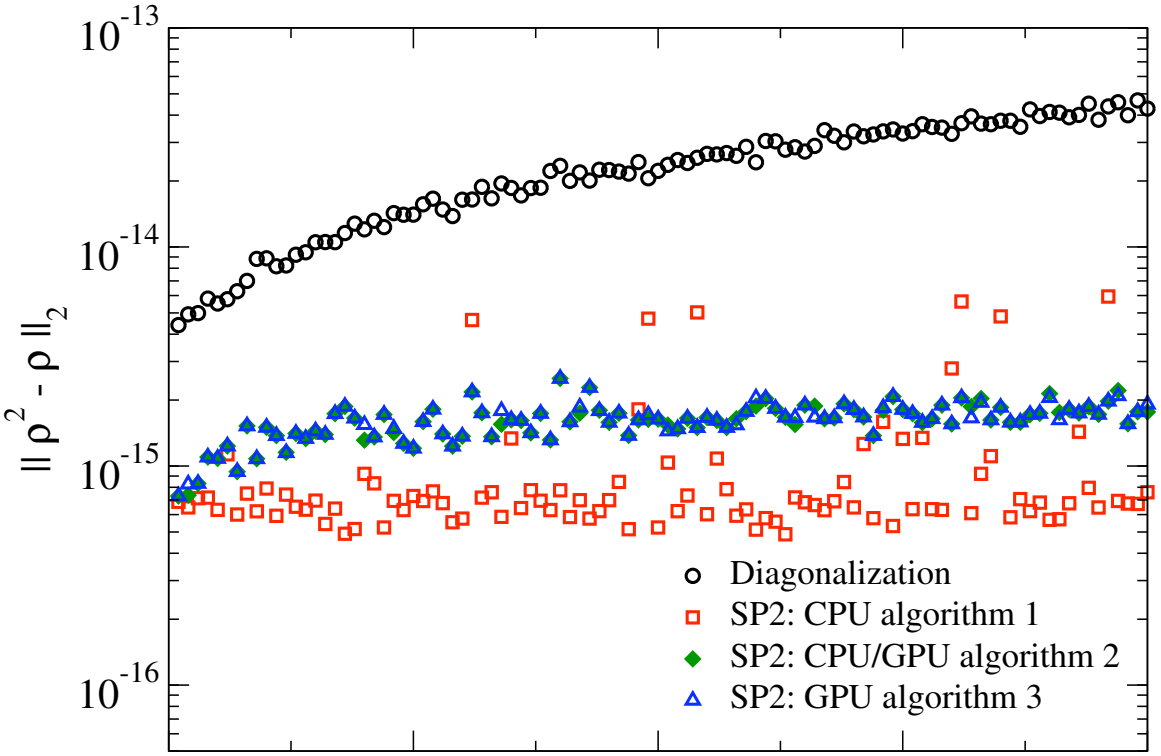


Average time to execute the CUBLAS xGEMM generalized matrix-matrix multiplication for M × M matrices. (a) DGEMM, (b) SGEMM. The broken and solid lines correspond computations performed with and without the padding of the arrays, respectively.

Performance Analysis: Density Matrix Calculation (Nvidia M2070) – Liquid Methane (10-1000 molecules)



Error Analysis in Density Matrices – Idempotency Measure



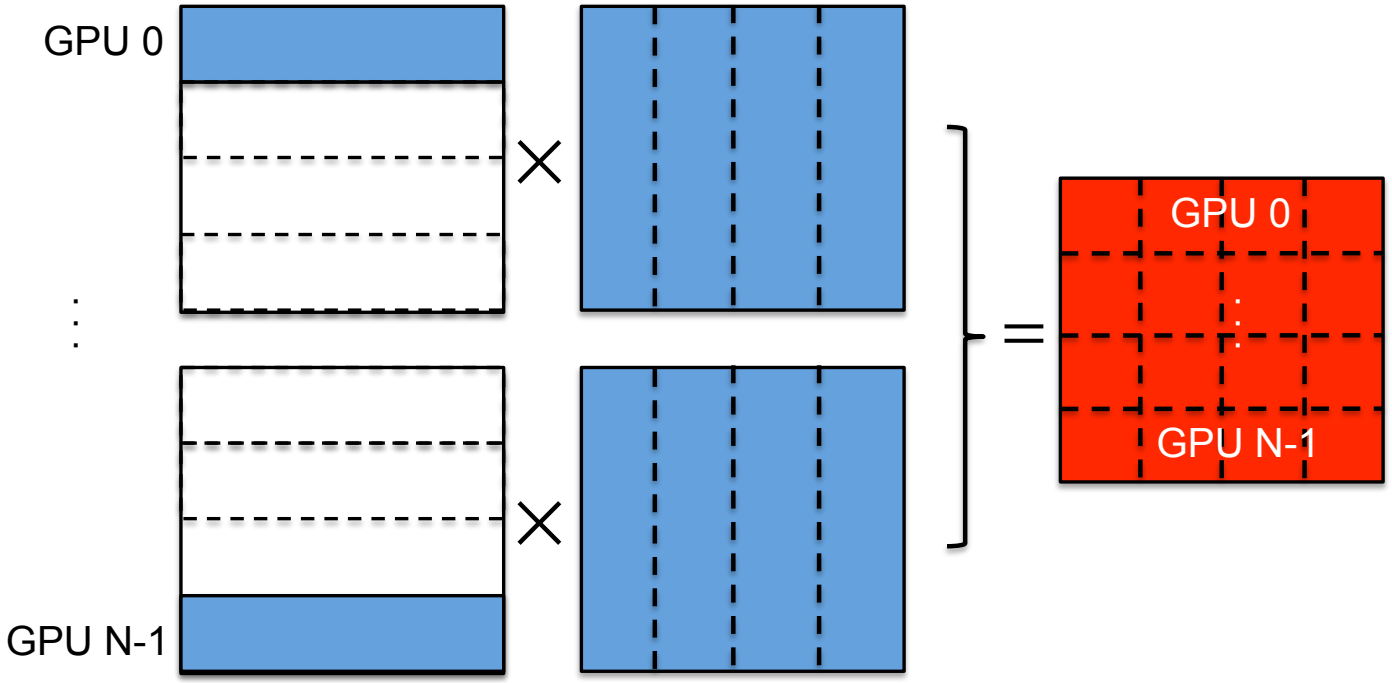
$$\rho^2 = \rho$$

$$\text{Error} = \left\| \rho^2 - \rho \right\|_2$$

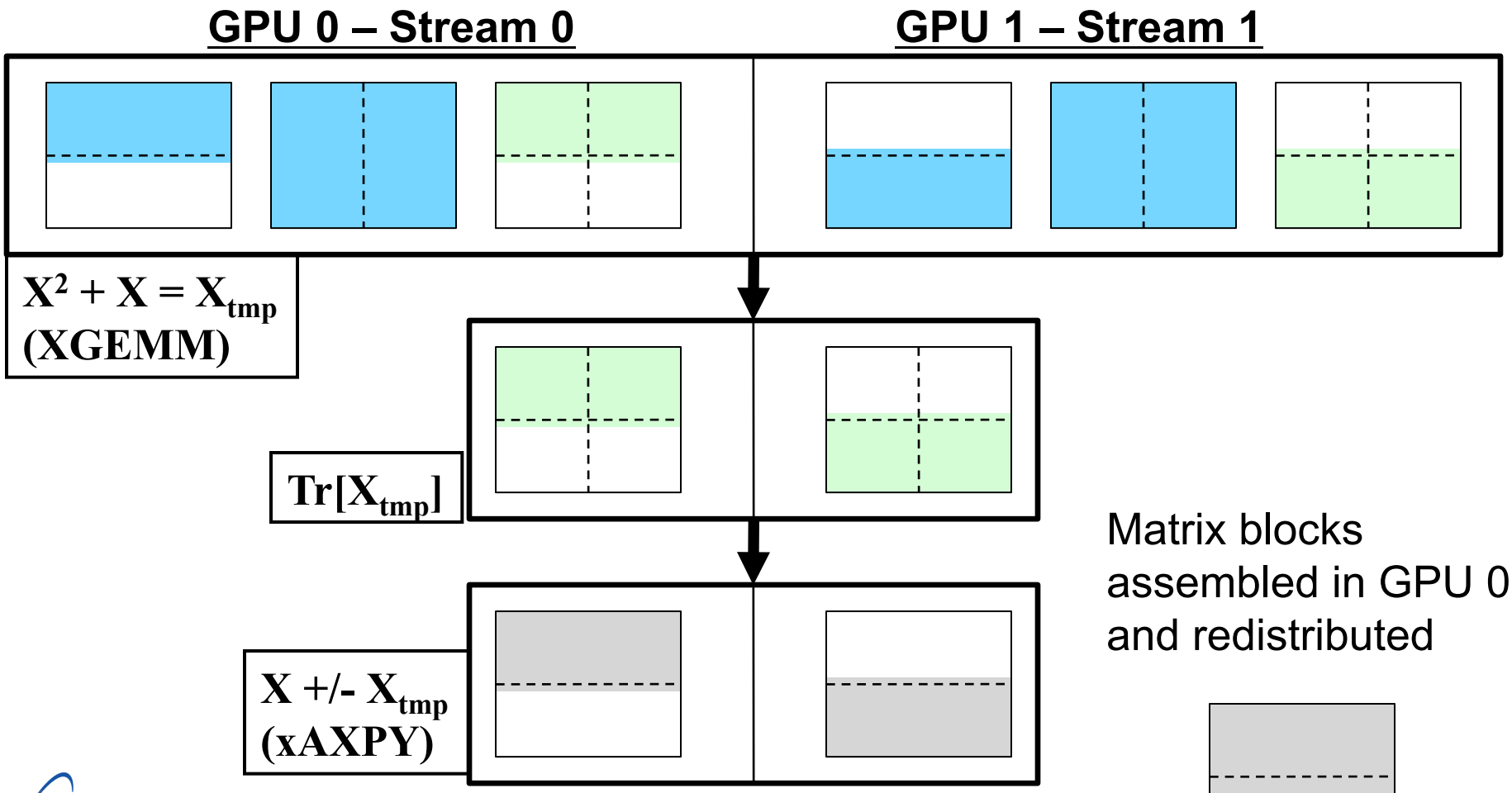
The SP2 algorithm has errors that are independent of system size whereas traditional diagonalization yields errors that increase with the number of atoms.

Multi-GPU Generalized Matrix-Matrix Multiplication

- **Using multiple streams for sub-block matrix-matrix multiplications, additions, and matrix traces**
- **Efficient reassembly of blocked matrix via native functionality of CUBLAS DGEMM/SGEMM**

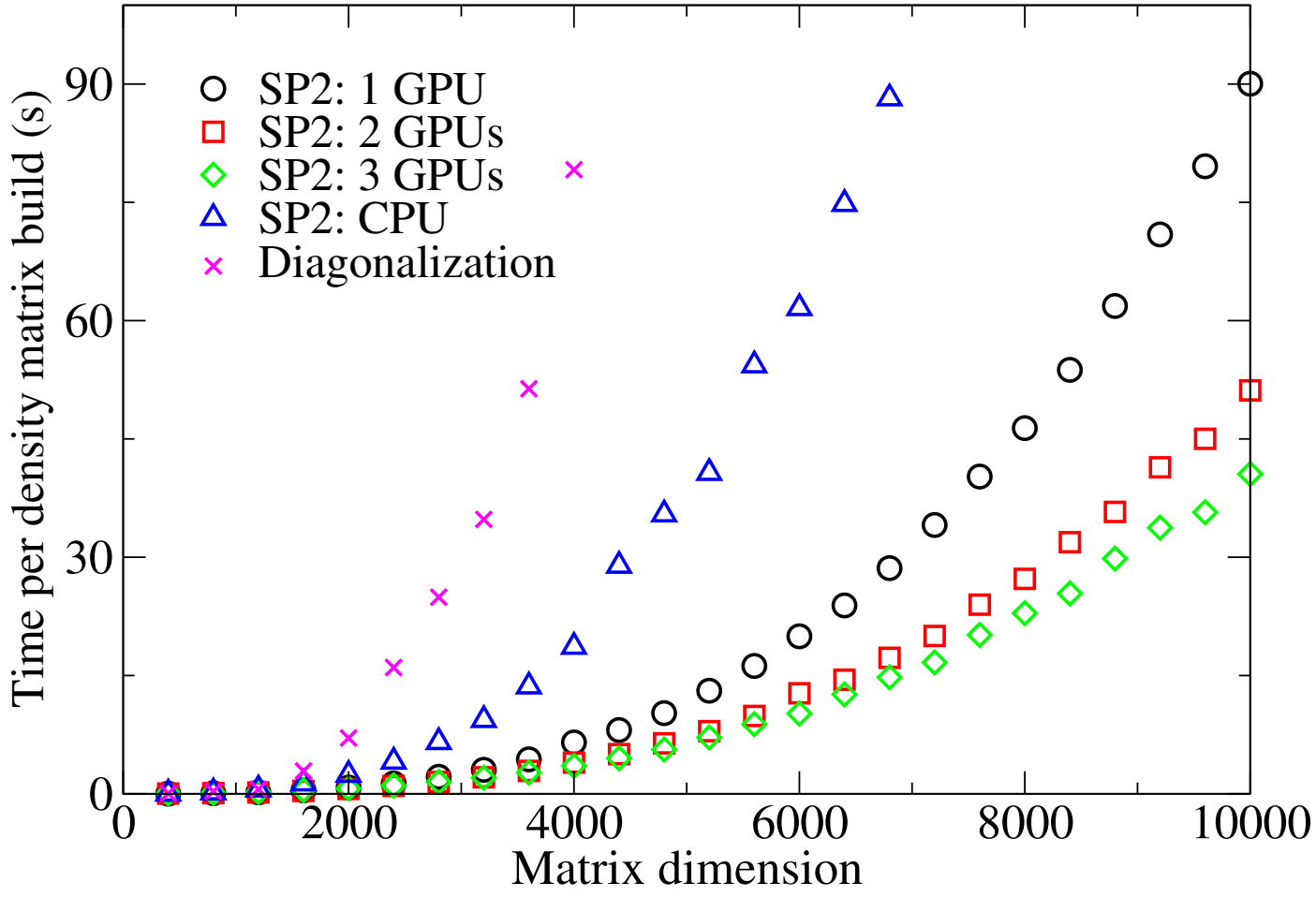


Using Streams for SP2 Sub-block Matrix Operations

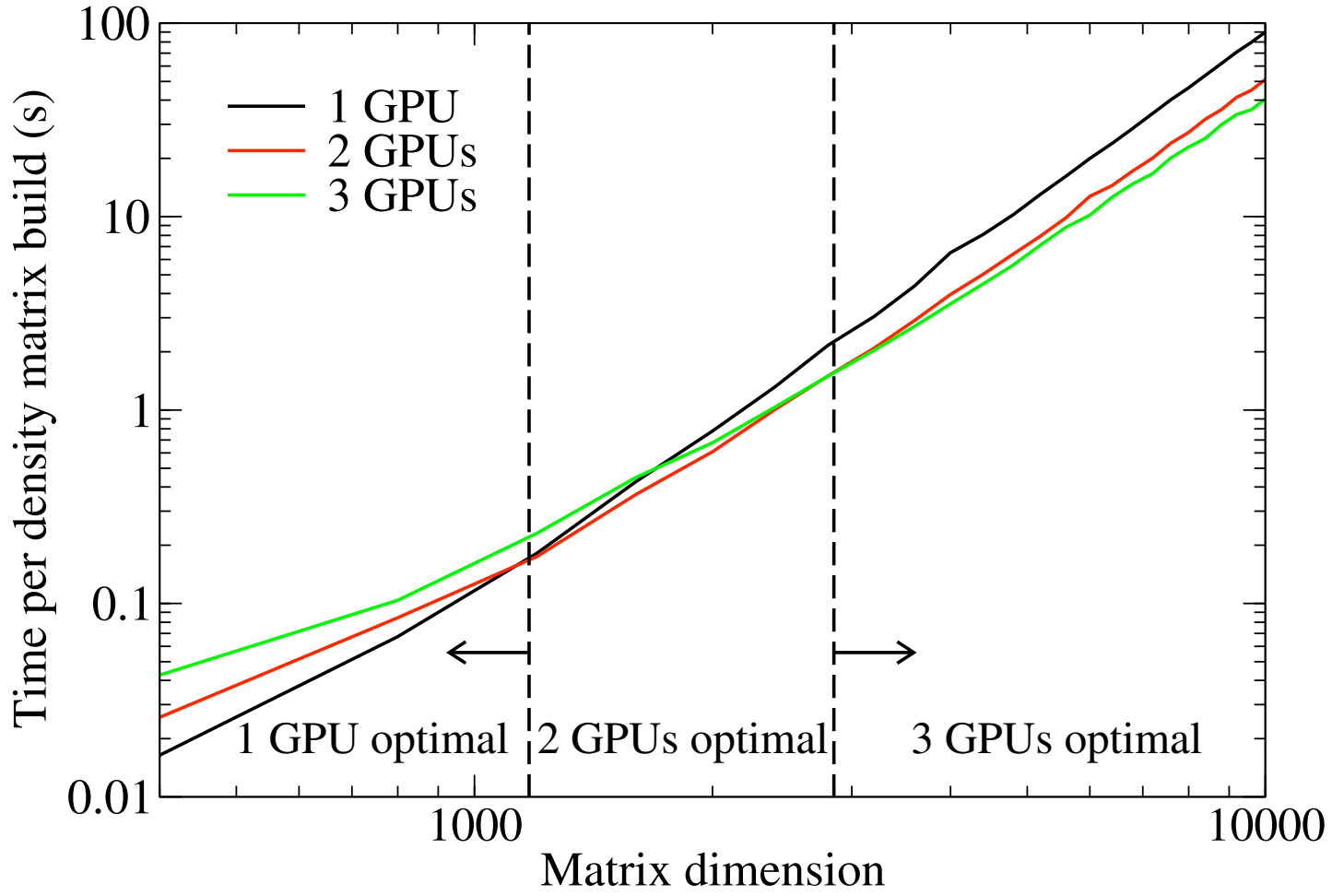


Matrix blocks assembled in GPU 0 and redistributed

Performance Analysis: Density Matrix Calculation (Nvidia M2090) – Liquid Methane (10 – 1250 molecules)



Performance Analysis for 1-3 GPUs



Summary

- GPUs can be effectively used for the density matrix computation in quantum mechanical models
- The recursive SP2 algorithm is well suited to the GPU architecture
- Transfer of arrays between the CPU and GPU are a minor performance contribution
- Array padding is important for performance
- The GPU version of the SP2 algorithm provides comparable or better accuracy over traditional diagonalization
- Massive speed-ups with respect to traditional algorithms have been seen with no loss of accuracy

Related Publications

1. Sanville EJ, Bock N, Coe J, Mniszewski SM, Niklasson AMN, Cawkwell MJ, 2010, LATTE. Los Alamos National Laboratory (LA-CC-10-004), http://savannah.nongpu.org/projects/latte.
2. Cawkwell MJ, Sanville EJ, Mniszewski SM, Niklasson AMN, 2012, Computing the Density Matrix in Electronic Structure Theory on Graphics Processing Units. J. Chem. Theory Comput., Vol 8, Issue 11, pp. 4094-4101, http://pubs.acs.org/doi/full/10.1021/ct300442w.
3. Mniszewski SM, Cawkwell MJ, Niklasson AMN, 2013, Quantum-based Dynamics on Graphics Processing Units. 2013 Associate Directorate for Theory, Simulation, and Computation (ADTSC) Highlights (in press).