Fast Matrix Inversion

- strong scaling: `t1 / ( N * tN ) * 100%`
- weak scaling: `( t1 / tN ) * 100%`

Eigen

	Time	Strong Scaling	Weak Scaling
Dense Serial	136 ms	N/A	N/A
Dense 24 thread	138 ms	4.1	98.0

Gaussian Elimination

	Time	Strong Scaling	Weak Scaling
Dense Serial	1821 ms	N/A	N/A
Dense 24 thread	281 ms	27.0	648.0

Best Configuration for 1024x1024: Eigen w/ 1 thread inverts 1024x1024 matrix in 136ms NO SEPARATE IMPLEMENTAITON FOR SPARSE MATRICES. tf, timing is the same

L1 Cache Size Estimation

Continue to increment B until the difference in timing results between B=N and B=N+1 are disproportionately different. The disproportionate increase in timing represents the new delays introduced by needing to access memory outside of the L1 cache.

Web: X5650 L1 Cache Size: 384 KB

Experiments:

- Set `M = 512`

Size B	Time
8	629 ms
16	595 ms
32	608 ms

These results show that the block size that produced the best timing was 16. Based on the algorithm being used, we can back calculate to determine what this stride length B corresponds to in terms of memory accesses.

Monte Carlo Estimations of PI

Time to Calculate PI using 1e6 Throws

Num Processes	Time
1	64 ms
2	47 ms
4	23 ms
8	15 ms
10	10 ms

MPI on Millions of Cores

Developments to positively impact ability to fully exploit parallelism

- Faster means of communication (improved interconnects) reduces propagation delay of information
- Improved communication protocols that improve end to end latency to reduce overhead of communication
- Larger cache sizes - allows more data to be stored and rapidly accessed to reduce the likelihood of a cache miss resulting in delays
- Better "many core architectures" and corresponding programming models allows for a more efficient use of cores (minimize overhead parallel programming)