Closing the Performance Gap

- Does a performance gap exist ? If so, are we making progress in closing it?
- Is it mostly a software gap or a hardware gap?
- How important is ease-of-use vs raw performance ?
- What are the one or two outstanding challenges for scientific computing ?

Does a performance gap exist ? If so, are we making progress in closing it?

What does ‘Performance Gap” refer to?

- Between the US and Japan ?
  - Exist? Yes,
  - Progress? Not clear since it is a moving target
- Between the peak and actual achieved performance on high end machines
  - Exist? Yes, both in single processor performance and in parallel performance
  - Progress? At least it’s encouraging that people are recognizing the problem and talking about it.

Is it mostly a software gap or a hardware gap?

- Software in the following sense:
  - Compilers and higher-level software packages are not fully compensating for multi-level memory access times
  - Need to switch to MPI style of programming has prevented the parallelization of many important “legacy codes”.
  - However, high level software packages like PETSc are a step in the right direction.
  - Can even more be done to hide memory latency via automatically prefetching, etc, especially for sparse matrix solves? Possibly within PETSc…hidden from the user
- Hardware in the following sense:
  - More effort going into increasing CPU clock frequency than in improving actual performance on real problems (like sparse matrix solves).
  - Memory access times are not decreasing as fast as compute cycle times (I am told), thus increasing the gap

How important is ease-of-use vs raw performance ?

- It is very important that application scientists can program and understand their own codes.
  - People in our community like Fortran, but are willing to evolve as Fortran evolves F77-> F90-> F2000. MPI not universally embraced.
  - Backwards compatibility is very important
- Programming in assembly language is a non-starter.
- High-level software packages like PETSc are very valuable
  - These packages should be both easy to use and optimized for performance
  - Note that widespread use of these packages puts a special responsibility on the groups that maintain them to make them available and optimized on new machines (like the Cray X1)

What are the one or two outstanding challenges for scientific computing ?

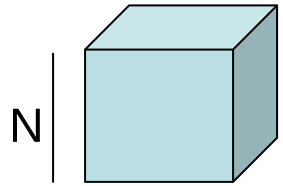
- For the (software) applications scientists:
  - Multiphysics or Integrated Modeling codes.
  - Software frameworks that make these possible
- For the hardware needs of these scientists:
  - Can we back to the days when the high-end machines were designed for scientific problems
  - …like in Japan
  - Efficient machine for sparse-matrix solves (that come from elliptic operators) are needed for long-time fusion (and other) simulations.

Note that more processors is not a substitute for faster processors unless code exhibits perfect strong scaling.

ie: weak scaling is not good enough

For example:

Let **N** be the number of mesh points per dimension.



For semi-implicit MHD code with perfect weak scaling: wall clock time to solution still increases as $\mathbf{N^2}$ as number of processors increases as $\mathbf{N^3}$

For a fixed problem time T, as N increases from 50 to 200, and the number of processors increases from 32 to 2048, the time to solution will increase from 24 Hours to 384 Hours …Not realistic!