Quick Overview of Scientific Computing

Prof. Eric Polizzi
The Role of Computing in Science and Engineering

- Computing as the relative new third component of science

- Computing is a primary tool for discovery and innovation in science & engineering disciplines, it is critical to advancing a diverse collection of applications.
The Role of Computing in Science and Engineering

- Truly interdisciplinary field focusing on the whole computational modeling process for solving large-scale/scope problems in science & engineering
Scientific Computing - How to proceed?

Numerical modeling is a multiple-step process:

Physical Modeling

How to describe the physics?

Mathematical Modeling

What are the equations to solve?

Numerical Modeling

How to discretize them?

Computing

Which algorithms to use?

Two difficulties:
(i) answering the questions above, multidisciplinary approach
(ii) making links between communities, interdisciplinary activity
\[
\left( -\frac{1}{2m} \nabla^2 + V_{\text{eff}}[n(r)] \right) \psi_i(r) = E_i \psi_i(r),
\]
\[
n(r) = 2 \sum_{i=1}^{N_E} |\psi_i(r)|^2.
\]
\[
V_{\text{eff}}[n(r)] = V_{\text{ion}}(r) + V_H[n(r)] + V_{\text{XC}}[n(r)],
\]
\[
-\nabla^2 V_H(r) = \frac{\rho(r)}{\varepsilon}.
\]

\[A x = \lambda B x\]

- Pick \( Y_{N \times M_0} = \{y_1, y_2, \ldots, y_{M_0}\} \) (\( M_0 > M \))
- Repeat
  1. Compute \( Q_{N \times M_0} = \alpha (B^{-1} A) Y_{N \times M_0} \)
  2. Orthogonalize \( Q \)
  3. Rayleigh-Ritz \( A_Q \hat{x}_Q = B_Q \hat{x}_Q \hat{\lambda} \)
     \( A_Q = (Q^H A Q)_{M_0 \times M_0} \)
     \( B_Q = (Q^H B Q)_{M_0 \times M_0} \)
     \( \hat{x}_Q = (N_{0 \times M_0}) \)
  4. Check convergence of \( X = Q \hat{x}_Q \) and \( \hat{\lambda} \)
  5. Go back step 1 if needed using \( Y = X \)
Computational Nanosciences: From atoms to Nanostructures
Scientific Computing

- **Math View → Numerical Analysis**
  - The overall goal of the field of numerical analysis is the design and analysis of numerical techniques to give approximate but accurate solutions to mathematical equations whose complexity make their intractable otherwise
  - “practical mathematical calculations”

- **CS view → High-Performance Computing (HPC)**
  - With the significant advances in parallel architectures (multicore workstations to petascale), there is a strong need for rethinking the computational strategies and numerical algorithms
Parallel computing

- New computational power at hand for the end user,
- Most existing software is not ready to directly utilize the power of parallel (multicore) processors since they are written in traditional sequential programming languages. The user/developer had the limited scope of only one processor in mind, expecting after the Moore’s law to continuously provide optimization of his sequential code.

  *The hardware scaling limit (the end of Moore’s law) is going to precipitate the need for the end user for new efficient parallel numerical libraries (suitable for multicore)*
Computing Paradigms

Old “conventional wisdom”
- Multiply is slow, but load & store is fast
- Do not bother “parallelizing” your application; just wait a little while and run on a much faster uniprocessor
- Less than linear scaling for a multiprocessor application is a failure

New “conventional wisdom”
- “memory wall” – load & store is slow but multiply is fast
- it will be a long wait for a faster uniprocessor (end of free-speedup)
- any speed improvement via parallelism is a success.
Graphs and Matrices

Model 'Real World' Problem:
- Internet traffic, social network, etc.

Model 'Mathematical' Problem:
- Solving differential equations via discretization (set of linear equations)
- Equation examples: wave, Maxwell, heat, Schrodinger, Poisson, etc.

Non-Numerical Algorithms
- Searches, MST, Shortest Path, Coloring, etc.

Numerical Algorithms
- Basics: Root Finding, Numerical integrations, Fast Fourier Transform
- Matrix Computations- Numerical Linear Algebra
Scientific Computing

- **Problem types (Matrix Computations)**
  - Matrix-Vector multiplications $Ax$
  - Linear systems $Ax=f$ (find $x$)
  - Eigenvalue problems $Ax=\lambda x$

- **Data formats**
  - Dense, Sparse

- **Implementation**
  - Data locality, low-level numerical libraries (e.g. BLAS, LAPACK) such as INTEL-MKL
  - Array-based implementation

- **Language**
  - Scripting: Matlab/Python (Education or special purpose)
  - C, C++, Fortran (for performance/research/production)

- **Parallelism**
  - Shared-memory (OpenMP)
  - Distributed memory (MPI)
'OpenMP' vs 'MPI'