# CS 450: Numerical Anlaysis

## Chapter 1 – Scientific Computing

## Lecture 1

### Numerical analysis introduction, motivation, and applications

### Posedness, error, and conditioning

Edgar Solomonik

Department of Computer Science
University of Illinois at Urbana-Champaign

January 18, 2018

# What is Numerical Analysis?

- **Numerical Problems**:

  *Given input $\boldsymbol{x} \in \mathbb{R}^n$, approximate output $\boldsymbol{y} = f(\boldsymbol{x})$*

  - *Problem is well-posed if $f$ is a smoothly varying function, $f(\hat{\boldsymbol{x}}) \rightarrow f(\boldsymbol{x})$ as $\hat{\boldsymbol{x}} \rightarrow \boldsymbol{x}$.*
  - *Otherwise, problem is ill-posed*

- **Error Analysis**:

  *Quality of approximation is quantified by distance to the solution*

  - *If solution $y = f(\boldsymbol{x})$ is a scalar, distance from computed solution $\hat{y}$ to correct answer is the absolute error*

    $$|\Delta y| = |\hat{y} - y|,$$

    *while the normalized distance is the relative error*

    $$|\Delta y|/|y| = |\hat{y} - y|/|y|$$

  - *More generally, we are interested in the error*

    $$\Delta \boldsymbol{y} = \hat{\boldsymbol{y}} - \boldsymbol{y}$$

    *the magnitude of which is measured by a given vector norm*

# Example: Mechanics[1]

- Newton's laws provide incomplete particle-centric picture
- Physical systems can be described in terms of *degrees of freedom* (DoFs)
  - A piston moving up and down requires __1__ DoFs
  - 1-particle system requires __3__ DoFs
  - 2-particle system requires __6__ DoFs
  - 2-particles at a fixed distance require __5__ DoFs
- $N$-particle system *configuration* described by $3N$ DoFs
  - *Trajectories in configuration space ($\mathbb{R}^{3N}$) describe free energy configuration*
  - *Various choice of basis functions (i.e. coordinate system) for configuration space are possible*

[1] *Variational Principles of Mechanics*, Cornelius Lanczos, Dover Books on Physics, 1949.

# Scientific Computing Applications and Context

- **Mathematical Modelling for Computational Science**

  *Typical scientific computing problems are numerical solutions to PDEs*

  - *Newtonian dynamics: simulating particle systems in time*
  - *Fluid and air flow models for engineering*
  - *PDE-constrained numerical optimization: finding optimal configurations (used in engineering of control systems)*
  - *Quantum chemistry (electronic structure calculations): many-electron Schrödinger equation*

- **Linear Algebra and Computation**

  - *Linear algebra and numerical optimization are building blocks for machine learning methods*
  - *Computer architecture, compilers, and parallel computing use numerical algorithms (matrix multiplication, Gaussian elimination) as benchmarks*

# Sources of Error

- **Representation of Numbers**:
  - *We cannot represent arbitrary real numbers in a finite amount of space, e.g. a computer cannot exactly represent* $\pi$
  - *Moreover, hardware architectures are only well-fit to work with fixed-length (32-bit or 64-bit) representations*
  - *As we will see, the best we can do is represent a wide range of numbers with a relatively uniform relative accuracy, which corresponds to scientific notation*
  - *With scientific notation, we seek to store the most significant digits of each number, so that the magnitude of the relative error in our representation for most real numbers* $x$ *will be* $|\hat{x} - x|/|x| \leq \epsilon$
- **Propagated Data Error**: *error due approximations in the input,* $f(\hat{x}) - f(x)$
- **Computational Error** = $\hat{f}(x) - f(x)$ = **Truncation Error + Rounding Error**
  - *Truncation error is the error made due to approximations made by the algorithm (simplified models used in our approximation)*
  - *Rounding error is the error made due to inexact representation of quantities computed by the algorithm*

# Error Analysis

- **Forward Error**:

  *Forward error is the computational error of an algorithm*

  - *Absolute:* $\hat{f}(x) - f(x)$
  - *Relative:* $(\hat{f}(x) - f(x))/f(x)$
  - *Usually, we care about the magnitude of the final error, but carrying through signs is important when analyzing error*

- **Backward Error**:

  *It can be hard to tell what a 'good' forward error is, but backward error analysis enables us to measure computational error with respect to data propagation error*

  - *An algorithm is backward stable if its a solution to a nearby problem*
  - *If the computed solution* $\hat{f}(x) = f(\hat{x})$ *then*

    $$\text{backward error} = \hat{x} - x$$

  - *More precisely, we want the nearest* $\hat{x}$ *to* $x$ *with* $\hat{f}(x) = f(\hat{x})$
  - *If the backward error is smaller than the propagated data error, the solution computed by the algorithm is as good as possible*

# Conditioning

- **Absolute Condition Number**:

  *The absolute condition number is a property of the problem, which measures its sensitivity to perturbations in input*

$$\kappa_{abs}(f) = \lim_{\text{size of input perturbation} \to 0} \max_{\text{inputs}} \max_{\text{perturbations in input}} \left| \frac{\text{perturbation in output}}{\text{perturbation in input}} \right|$$

  *For problem $f$ at input $x$ it is simply the derivative of $f$ at $x$,*

$$\kappa_{abs}(f) = \lim_{\Delta x \to 0} \left| \frac{f(x+\Delta x) - f(x)}{\Delta x} \right| = \left| \frac{df}{dx}(x) \right|$$

  *When considering a space of inputs $\mathcal{X}$ it is $\kappa_{abs} = \max_{x \in \mathcal{X}} \left| \frac{df}{dx}(x) \right|$*

- **(Relative) Condition Number**:

  *The relative condition number considers relative perturbations in input and output, so that*

$$\kappa(f) = \kappa_{rel}(f) = \max_{x \in \mathcal{X}} \lim_{\Delta x \to 0} \left| \frac{(f(x+\Delta x) - f(x))/f(x)}{\Delta x / x} \right| = \frac{\kappa_{abs}(f)|x|}{|f(x)|}$$

# Posedness and Conditioning

- **What is the condition number of an ill-posed problem?**
  - *If the condition number is bounded and the solution is unique, the problem is well-posed*
  - *An ill-posed problem $f$ either has no unique solution or has a (relative) condition number of $\kappa(f) = \infty$*
  - *This condition implies that the solutions to problem $f$ are continuous and differentiable in the given space of possible inputs to $f$*
  - *Sometimes well-posedness is defined to only require continuity*
  - *Generally, $\kappa(f)$ can be thought of as the distance (in an appropriate geometric embedding of problem configurations) from $f$ to the nearest ill-posed problem*

# Stability and Accuracy

- **Accuracy**:

  *An algorithm is accurate if $\hat{f}(x) = f(x)$ for all inputs $x$ when $\hat{f}(x)$ is computed in infinite precision*

  - *In other words, the truncation error is zero (rounding error is ignored)*
  - *More generally, an algorithm is accurate if its truncation error is negligible in the desired context*
  - *Yet more generally, the accuracy of an algorithm is expressed in terms of bounds on the magnitude of its truncation error*

- **Stability**:

  *An algorithm is stable if its output in finite precision (floating point arithmetic) is always near its output in exact precision*

  - *Stability measures the sensitivity of an algorithm to roundoff error*
  - *In some cases, such as the approximation of a derivative using a finite difference formula, there is a trade-off between stability and accuracy*