Numerical Computing

Lecture 1: Introduction & Foundations

Francisco Richter Mendoza

Università della Svizzera Italiana (USI)
Faculty of Informatics
Lugano, Switzerland

Today's Topics

- Well-posed problems
- Sources of error
- Stability & conditioning
- Computational complexity
- Educational demonstrations

Well-Posed Problems

Definition (Hadamard)

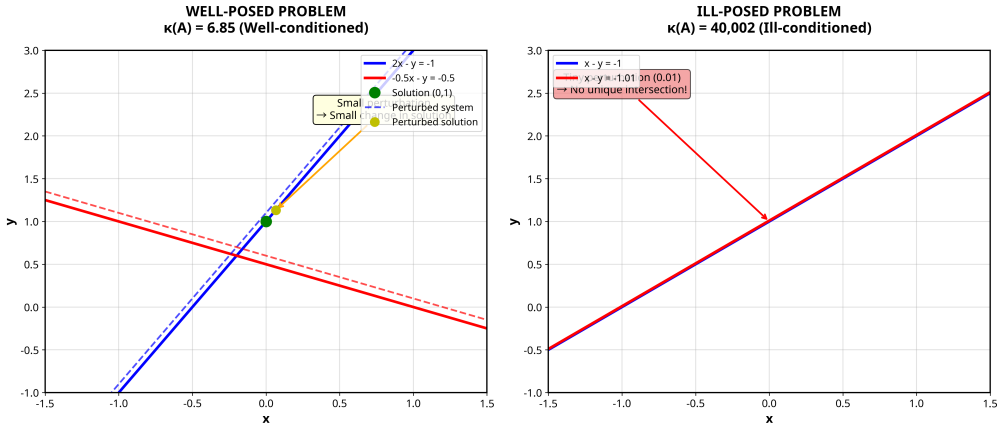
A problem is **well-posed** if:

1. **Existence:** Solution exists
2. **Uniqueness:** Solution is unique
3. **Stability:** Solution depends continuously on data

$$\|S(f_1) - S(f_2)\| \leq C\|f_1 - f_2\| \tag{1}$$

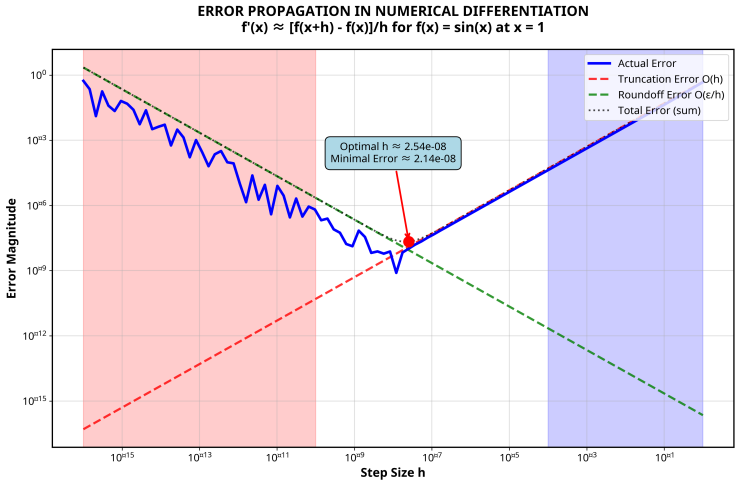
S: solution operator, C: stability constant

Well-Posed vs Ill-Posed Problems



Key Insight: Small perturbations in well-posed problems lead to small changes in solutions. Ill-posed problems amplify small errors dramatically.

Error Propagation in Numerical Methods



Lesson: There exists an optimal step size that balances truncation and roundoff errors. Too small $h \to$ roundoff dominates; too large $h \to$ truncation dominates.

Sources of Error

Types of Error:

- Modeling error
- Discretization error
- Roundoff error
- Iteration error

Error Measures:

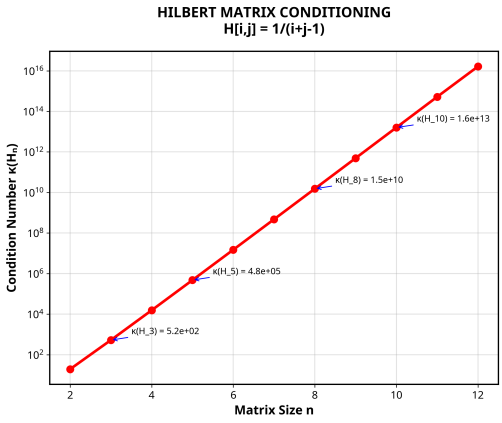
$$\text{Absolute:} \quad |x - \tilde{x}| \tag{2}$$

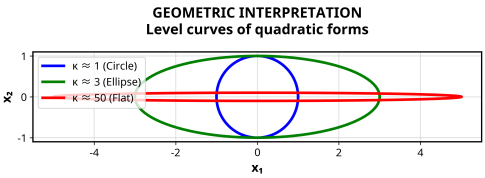
$$\text{Relative:} \quad \frac{|x - \tilde{x}|}{|x|} \tag{3}$$

Error Bounds:

$$\text{Total Error} = \text{Truncation} + \text{Roundoff} \tag{4}$$

Condition Numbers & Geometric Interpretation





Higher condition number
→ More elongated ellipse
→ Greater sensitivity

Understanding: Hilbert matrices become exponentially ill-conditioned. Geometric shapes reveal sensitivity: circles (well-conditioned) vs flat ellipses (ill-conditioned).

Condition Number Theory

Definition

For problem $y = f(x)$:

$$\kappa = \lim_{\delta \to 0} \sup_{\|\delta x\| \leq \delta} \frac{\|f(x + \delta x) - f(x)\| / \|f(x)\|}{\|\delta x\| / \|x\|} \tag{5}$$

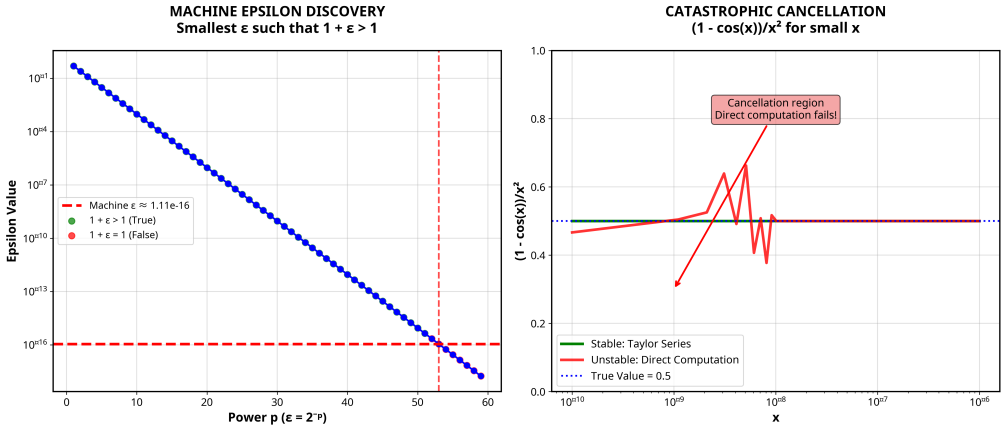
For linear systems $Ax = b$:

$$\kappa(A) = \|A\| \|A^{-1}\| \quad \text{and} \quad \frac{\|\delta x\|}{\|x\|} \leq \kappa(A) \frac{\|\delta b\|}{\|b\|} \tag{6}$$

Interpretation:

- $\kappa \approx 1$: **Well-conditioned** (stable)
- $\kappa \gg 1$: **Ill-conditioned** (sensitive)

Floating Point Arithmetic & Catastrophic Cancellation



Critical Insight: Machine epsilon $\approx 2.22 \times 10^{-16}$ defines precision limits. Catastrophic cancellation occurs when subtracting nearly equal numbers.

Avoiding Catastrophic Cancellation

Problem: Computing $(1 - \cos(x))/x^2$ for small x

BAD - Direct computation:

```
x = 1e-8
result = (1 - np.cos(x)) / x**2
# Result: 0.0 (wrong!)
```

GOOD - Taylor series:

```
x = 1e-8
result = 0.5 - x**2/24 + x**4/720
# Result: 0.5 (correct!)
```

Why it fails:

- $\cos(10^{-8}) \approx 1 - 5 \times 10^{-17}$
- $1 - \cos(x) \approx 0$ (cancellation!)
- Division by $x^2 = 10^{-16}$ amplifies error

Solution: Use mathematically equivalent but numerically stable formulations.

Algorithm Stability

Forward Stability

Algorithm produces nearly correct answer to nearly correct problem:

$$\tilde{y} = f(\tilde{x}) \quad \text{where } \|\tilde{x} - x\| \text{ is small} \tag{7}$$

Backward Stability

Algorithm produces exact answer to nearby problem:

$$\tilde{y} = f(x + \delta x) \quad \text{where } \|\delta x\| \text{ is small} \tag{8}$$

Backward stability is stronger: If an algorithm is backward stable, then it's also forward stable (assuming the problem is well-conditioned).

Computational Complexity

Big-O Notation

$f(n) = O(g(n))$ if $\exists C, n_0$ such that:

$$|f(n)| \leq C|g(n)| \quad \forall n \geq n_0 \tag{9}$$

Common complexities in numerical methods:

$$O(n) : \text{Vector operations, simple iterations} \tag{10}$$

$$O(n^2) : \text{Matrix-vector products, forward/back substitution} \tag{11}$$

$$O(n^3) : \text{Matrix factorizations (LU, QR), matrix multiplication} \tag{12}$$

$$O(n \log n) : \text{Fast Fourier Transform (FFT)} \tag{13}$$

Algorithm Design Principles

Accuracy Requirements:

- Minimize truncation error
- Control roundoff accumulation
- Provide error bounds
- Verify convergence

Efficiency Considerations:

- Optimize time complexity
- Minimize memory usage
- Enable parallelization
- Scale to large problems

The Fundamental Trade-off

Accuracy $\leftrightarrow$ **Efficiency** $\leftrightarrow$ **Stability**

Choose algorithms that balance these competing requirements for your specific application.

Practical Guidelines for Numerical Computing

1. **Check well-posedness:** Ensure your problem has a unique, stable solution
2. **Monitor condition numbers:** $\kappa(A) > 10^{12}$ signals trouble
3. **Avoid cancellation:** Reformulate expressions to prevent subtraction of nearly equal quantities
4. **Use stable algorithms:** Prefer backward stable methods when available
5. **Validate results:** Check residuals, perform sensitivity analysis

Golden Rule

"The condition number of a problem determines how accurately it can be solved, regardless of the algorithm used."

Key Takeaways

- **Well-posedness** is essential: existence, uniqueness, stability
- **Condition numbers** quantify problem sensitivity to perturbations
- **Error sources** include modeling, discretization, roundoff, and iteration
- **Floating point** arithmetic has fundamental limitations ($\epsilon_{\text{mach}} \approx 10^{-16}$)
- **Algorithm design** must balance accuracy, efficiency, and stability

Next Lecture

Computer Arithmetic & Error Analysis in Detail