UNIVERSITY AT ALBANY
STATE UNIVERSITY OF NEW YORK

CSI 401 (Fall 2025)

Numerical Methods

Lecture 4: Floating Point & Linear Systems

Chong Liu

Department of Computer Science

Sep 15, 2025

Announcement

- Starting from today, you can earn your participation points!
  - How?
  - If you answered questions (either correct/incorrect!), showed your solutions to in-class exercise problems,
  - You can come to me to register your name.
  - 1 point per lecture

- HW 1 due today

- HW 2 will be released soon

Agenda

- Recap of Lecture 2
- Floating point system
  - Truncation and rounding
- Linear system
  - Definitions and applications
  - Solvers:
    - Gaussian Elimination
    - Gauss-Jordan Elimination

Recap: Asymptotic notations

- $f(x) = O(g(x))$ as $x \to x_0$ if there is some positive constant C such that $\left|\frac{f(x)}{g(x)}\right| \leqslant C$.

- Growth rates in increasing order:
  - $\log n, \sqrt{n}, n, n \log n, n^3, 2^n, n!$

Recap: Decimal expansion

- Take 316.1415 for example:

$$316.1415 = 3 \cdot 10^2 + 1 \cdot 10^1 + 6 \cdot 10^0 + 1 \cdot 10^{-1} + 4 \cdot 10^{-2} + 1 \cdot 10^{-3} + 5 \cdot 10^{-4}.$$

- Any real number x can be written as

$$x = \pm \sum_{j=-\infty}^{\infty} d_j \cdot 10^j$$

Recap: Scientific notation

Recall how scientific notation works. In decimal, we can write any real number other than 0 as

$$x = \pm m \times 10^E, \tag{5.12}$$

for a unique **mantissa** m and exponent E, with $1 \leqslant m < 10$ and E some integer. For example, consider the number 314.159. In scientific notation, this is written as

$$3.14159 \times 10^2. \tag{5.13}$$

In the same fashion, a number can be written in base 2 scientific notation: it takes the form

$$x = \pm m \times 2^E, \tag{5.14}$$

where this time $1 \leqslant m < 2$. For instance, consider the number 3.25. We converted this to binary to get $(11.01)_2$. In scientific notation, this becomes

$$(1.101)_2 \times 2^1. \tag{5.15}$$

- In-class exercise: scientific notations of 4125, 40.125, 4.125

How data are stored? Floating point system

b_{sign}	$b^{mant}_{n_M-1} b^{mant}_{n_M-2} \dots b^{mant}_1 b^{mant}_0$	$b^{exp}_{n_E-1} b^{exp}_{n_E-2} \dots b^{exp}_1 b^{exp}_0$

Here, there is a single bit giving the sign of the number (0 for negative, 1 for positive). Next is the mantissa, stored as an n_M-bit number (usually 52 bits). Finally, the exponent is stored as an n_E-bit number (usually 11 bits). For a nonzero number, the mantissa is not stored directly: since it is between 1 and 2, the binary expansion always begins with a 1. This is redundant, so we **do not** explicitly store it in the floating point representation. It is simply assumed to be there, leading to the so-called **hidden bit representation**. The number 0 has a special representation as all 0s.

- Discussion: what's the number in [0|0100000...|...0000011]?
- Solution: -(1+0.25)*8=-10.

Example of storing x=3.125 in computers

Example 5.3. *Suppose that we want to store the number $x = 3.125$ in the floating point representation.*

First, we find the binary expansion of x. We do this using the algorithm that we covered last time, and we get

$$x = (11.001)_2. \tag{5.16}$$

We then convert it to binary scientific notation, which gives us

$$x = (1.1001)_2 \times 2^1. \tag{5.17}$$

Thus, the mantissa is $m = (1.1001)_2$, while the exponent is $E = (000000001)_2$. If the number of mantissa bits $n_M = 16$ and the number of exponent bits $n_E = 8$, then this would be stored as

1	1001 0000 0000 0000	0000 0001

Recall that the initial 1 in the mantissa is not explicitly stored.

Truncation and rounding

1. Chopping/Truncation: Here, we simply ignore the bits after a certain point. For instance, if $n_M = 4$, then the mantissa for $(0.1)_{10}$ after truncation would be 1001. This is equivalent to **rounding toward** 0. In other words, the absolute value of the rounded number is less than or equal to that of the original, but the sign remains the same.

$$(0.1)_{10} = (0.0001100110011...)_2$$

2. Rounding up: e.g., $(1.0110011)_2$, rounded up after the 4th place, would result in $(1.0111)_2$. In general, the rounded number is greater than or equal to the original (even if the number is negative). Note that rounding a negative number up is the same as truncating it.

3. Rounding down: e.g., $(1.0111)_2$, rounded down after the second decimal place, would result in $(1.010)_2$. In general,the rounded number is less than or equal to the original. Note that rounding a positive number down is the same as truncating it.

4. Rounding to nearest (the default mode in the IEEE standard): e.g., $(1.0110 \mid 11)_2$ becomes $(1.0111)_2$, but $(1.0110 \mid 011)_2$ becomes $(1.0110)_2$. This is because $(0.000011)_2$ is closer to $(0.0001)_2$ than to 0.0000, but $(0.0000011)_2$ is closer to 0 than to $(0.0001)_2$.

   The algorithm for rounding to the nearest k digits (in binary) after the decimal point is as follows: on input x,

   (a) If the $k + 1$st digit after the decimal point is 0, then truncate.

   (b) If the $k + 1$st digit after the decimal point is 1 and $x > 0$, then round up.

   (c) If the $k + 1$st digit after the decimal point is 1 and $x < 0$, then round down.

Recap: Types of errors

- Discretization error: we can only deal with values of a function at finitely many points. For example, a very simple way to do numerical differentiation for a function f is to use a finite difference formula:

$$\hat{f}(x) = \frac{f(x+h) - f(x)}{h}. \tag{2.1}$$

  Here, the parameter h is some small number. It cannot be 0, so this introduces discretization error. Recall that the definition of the derivative of a function at a point x is

$$\lim_{h \to 0} \frac{f(x+h) - f(x)}{h}. \tag{2.2}$$

  Later in the course, we'll considered better methods than this.

- Convergence error: in which we, say, truncate a power series expansion, stop an iterative algorithm after finitely many iterations, etc.

- Rounding error: This arises because computers have only finite precision. We can only store a finite amount of data in any given machine. Interestingly, in numerical differentiation, there is a tradeoff between discretization error and rounding error (since we cannot make h infinitely small), and this leads to some optimal choice of h! So multiple types of error can play an important role simultaneously in some problems.

Linear systems (linear equations)

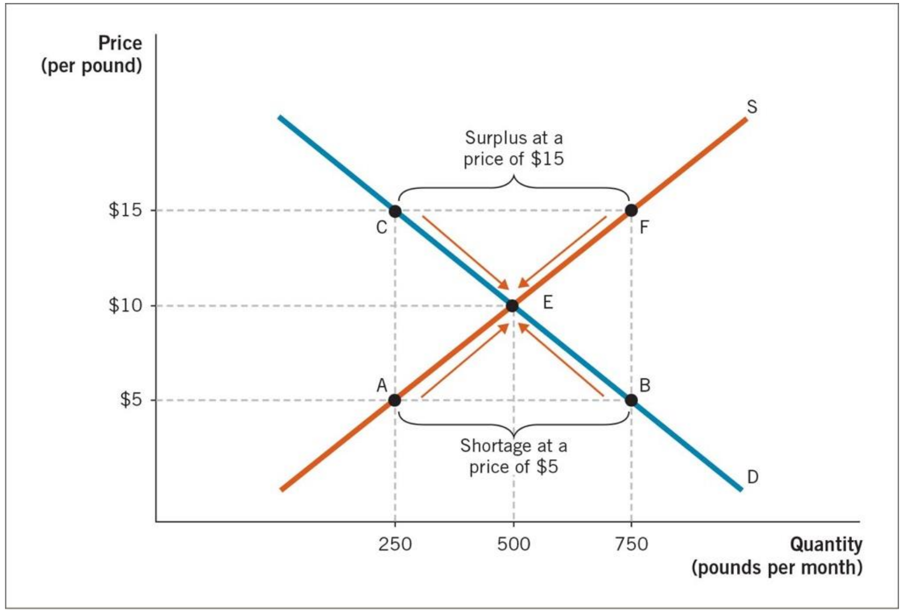
- An example of linear systems
  - Any linear system can always be rewritten in matrix form

$$
\begin{aligned}
a_{11}x_1 + a_{12}x_2 + a_{13}x_3 &= b_1 \\
a_{21}x_1 + a_{22}x_2 + a_{23}x_3 &= b_2 \\
a_{31}x_1 + a_{32}x_2 + a_{33}x_3 &= b_3
\end{aligned}
\qquad
\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}
\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}
=
\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}
$$

- More generally, $Ax = b$
  - A is an $m \times n$ matrix
  - x is an n-dimensional vector
  - b is an m-dimensional vector
- Problem: given A and b, how can you solve x?

So many applications of linear systems

- **Computer graphics & vision:**
  - Linear transforms (rotation/scale/shear), camera models, color space conversion
- **Machine learning & data science:**
  - Linear regression, least squares
- **Optimization & operations research:**
  - Network flow, scheduling, logistics
- **Economics & finance:**
  - Supply–demand equilibrium
- **Chemistry & biology:**
  - metabolic flux analysis solving steady-state linear constraints
- **Robotics:**
  - Jacobian-based small-motion models
- **Computer networks:**
  - Traffic routing and capacity planning via linear flow conservation constraints

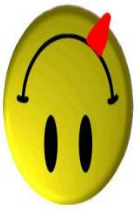


Recap: In-class exercise: map each pixel to a new location

b) The **smiley face** visible to the right is transformed with various linear transformations represented by matrices $A - F$. Find out which matrix does which transformation:

$$A = \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix},$$

$$D = \begin{bmatrix} 1 & -1 \\ 0 & -1 \end{bmatrix}, \quad E = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}, \quad F = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} / 2$$

A-F	image	A-F	image	A-F	image
					
					

Discussion

$$\begin{aligned} x + y + z &= 7 \\ 3x + 2y + z &= 11 \\ 4x - 2y + 2z &= 8 \end{aligned} \qquad \begin{aligned} x + y + z &= 7 \\ -y - 2z &= -10 \\ 10z &= 40 \end{aligned}$$

- Which of two systems is easier to be solved?
- How can you solve the easier one?

- Actually, these two are equivalent to each other! How??

Gaussian elimination

- Rewrite the problem in augmented matrix form
  - Original augmented matrix and manipulated augmented matrix

$$\begin{bmatrix} 1 & 1 & 1 & 7 \\ 3 & 2 & 1 & 11 \\ 4 & -2 & 2 & 8 \end{bmatrix} \longrightarrow \begin{bmatrix} 1 & 1 & 1 & 7 \\ 0 & -1 & -2 & -10 \\ 0 & 0 & 10 & 40 \end{bmatrix}$$

- Elementary row operations:
  - 2^{nd} line = 2^{nd} line – 3 * 1^{st} line [0 -1 -2 -10], done!
  - 3^{rd} line = 3^{rd} line – 4 * 1^{st} line [0 -6 -2 -20]
  - Discussion: how to proceed?
  - 3^{rd} line = 3^{rd} line – 6 * 2^{nd} line [0 0 10 40], done!

Gaussian elimination

- Rewrite the problem in augmented matrix form
  - Original augmented matrix and manipulated augmented matrix

$$\begin{bmatrix} 1 & 1 & 1 & 7 \\ 3 & 2 & 1 & 11 \\ 4 & -2 & 2 & 8 \end{bmatrix} \longrightarrow \begin{bmatrix} 1 & 1 & 1 & 7 \\ 0 & -1 & -2 & -10 \\ 0 & 0 & 10 & 40 \end{bmatrix}$$

- Three elementary row operations:
  - Multiply a row by a non-zero constant [Notation: $3R_1$]
  - Exchange two rows [Notation: $R_{2,3}$]
  - Add one row with a multiplied row [Notation: $R_1 + (-8)R_3$]

Gaussian elimination

- Repeat this process in equation form

$$\begin{aligned} x + y + z &= 7 \\ 3x + 2y + z &= 11 \\ 4x - 2y + 2z &= 8 \end{aligned} \qquad \begin{aligned} x + y + z &= 7 \\ -y - 2z &= -10 \\ 10z &= 40 \end{aligned}$$

- Elementary row operations:
  - 2nd line = 2nd line – 3 * 1st line [-y-2z=-10], done!
  - 3rd line = 3rd line – 4 * 1st line [-6y-2z=-20]
  - 3rd line = 3rd line – 6 * 2nd line [10z=40], done!

Gaussian elimination

- Key idea:
  - Use elementary row operations to make A become a right-triangular matrix
  - So that you can sequentially solve the linear systems bottom-up!

$$
\left[\begin{array}{cccc|c}
a'_{11} & a'_{12} & \cdots & a'_{1k} & b'_1 \\
0 & a'_{22} & \cdots & a'_{2k} & b'_2 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \cdots & a'_{kk} & b'_k
\end{array}\right].
$$

In-class exercise: Gaussian Elimination

- Solve this linear system:

$$y + 3z = 4$$
$$-x + 2y = 3$$
$$2x - 3y + 4z = 1$$

- Solution:

$$\begin{bmatrix} 0 & 1 & 3 & 4 \\ -1 & 2 & 0 & 3 \\ 2 & -3 & 4 & 1 \end{bmatrix} \to R_{1,2} \to \begin{bmatrix} -1 & 2 & 0 & 3 \\ 0 & 1 & 3 & 4 \\ 2 & -3 & 4 & 1 \end{bmatrix} \to (-1)R_1 \to \begin{bmatrix} 1 & -2 & 0 & -3 \\ 0 & 1 & 3 & 4 \\ 2 & -3 & 4 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & -2 & 0 & -3 \\ 0 & 1 & 3 & 4 \\ 2 & -3 & 4 & 1 \end{bmatrix} \to R_3 + (-2)R_1 \to \begin{bmatrix} 1 & -2 & 0 & -3 \\ 0 & 1 & 3 & 4 \\ 0 & 1 & 4 & 7 \end{bmatrix} \to R_3 + (-1)R_2 \to \begin{bmatrix} 1 & -2 & 0 & -3 \\ 0 & 1 & 3 & 4 \\ 0 & 0 & 1 & 3 \end{bmatrix}$$

Row 3: $z = 3$

Row 2: $y + 3(3) = 4$, so $y = -5$

Row 1: $x - 2(-5) = -3$, so $x = -13$

Beyond Gaussian Elimination …

- Consider this linear system:

$$
\begin{aligned}
x - 2y + 3z &= 9 \\
-x + 3y \quad\;\; &= -4 \\
2x - 5y + 5z &= 17
\end{aligned}
$$

- In-class exercise: Write in in Gaussian elimination style.

$$
\begin{bmatrix} 1 & -2 & 3 & 9 \\ -1 & 3 & 0 & -4 \\ 2 & -5 & 5 & 17 \end{bmatrix}
\xrightarrow[\text{Gaussian elimination}]{\text{Elementary row operations}}
\begin{bmatrix} 1 & -2 & 3 & 9 \\ 0 & 1 & 3 & 5 \\ 0 & 0 & 1 & 2 \end{bmatrix}
$$

- Discussion: is it possible to further simplify A as an identity matrix?

Gauss-Jordan Elimination: Beyond Gaussian Elimination

- Consider this linear system:

$$
\begin{aligned}
x - 2y + 3z &= 9 \\
-x + 3y \quad\quad &= -4 \\
2x - 5y + 5z &= 17
\end{aligned}
$$

- Yes! It’s Gauss-Jordan Elimination.
  - Key idea:
    - Use elementary row operations to make A become an identity matrix
    - So that you can directly read the results!

$$
\begin{bmatrix} 1 & -2 & 3 & 9 \\ -1 & 3 & 0 & -4 \\ 2 & -5 & 5 & 17 \end{bmatrix}
\xrightarrow[\text{Gaussian elimination}]{\text{Elementary row operations}}
\begin{bmatrix} 1 & -2 & 3 & 9 \\ 0 & 1 & 3 & 5 \\ 0 & 0 & 1 & 2 \end{bmatrix}
\xrightarrow[\text{Gauss-Jordan elimination}]{\text{Elementary row operations}}
\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 2 \end{bmatrix}
$$

In-class exercise: Gauss-Jordan Elimination

- Solve this linear system:

$$
\begin{aligned}
2x + 4y &= -2 \\
x + 2y + 2z &= 7 \\
3x - 3y - z &= 11
\end{aligned}
$$