Echelon Forms

Geometric Algorithms
Lecture 3

CAS CS 132

Practice Problem

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \sim \begin{bmatrix} 4 & 5 & 6 \\ 1 & 2 & 3 \\ 7 & 8 & 9 \end{bmatrix}$$

Write a sequence of elementary row operations which transforms the left matrix to the right matrix **without using the exchange operation.**

Solution

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \sim \begin{bmatrix} 4 & 5 & 6 \\ 1 & 2 & 3 \\ 7 & 8 & 9 \end{bmatrix}$$

Objectives

1. Introduce echelon forms as a kind of matrix which "represents" solutions

2. Learn how to "read off" a solution from an echelon form matrix

3. Start discussing Gaussian elimination

Keywords

leading entries

echelon form

(row-)reduced echelon form (RREF)

pivot positions

pivot columns

free variables

basic variables

general form solutions

forward elimination

back substitution

Recap

Recall: Linear Systems (General-form)

$$
\begin{aligned}
a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n &= b_1 \\
a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n &= b_2 \\
&\vdots \\
a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n &= b_m
\end{aligned}
$$

Recall: Linear Systems (General-form)

$$
\begin{aligned}
a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n &= b_1 \\
a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n &= b_2 \\
&\vdots \\
a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n &= b_m
\end{aligned}
$$

Does a system have a solution?

How many solutions are there?

What are its solutions?

Recall: Matrix Representations

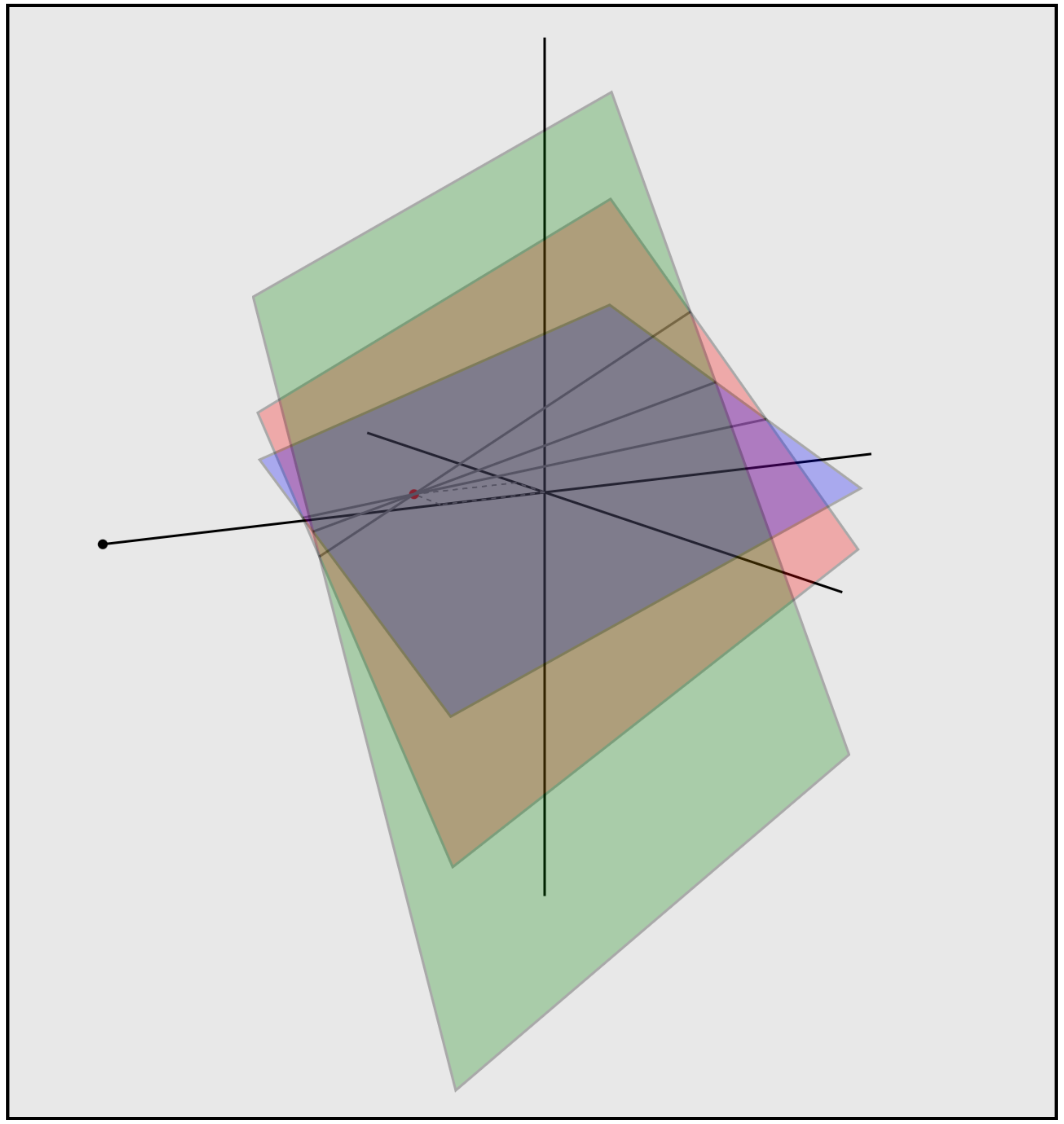
$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & b_m \end{bmatrix}$$

Recall: Matrix Representations

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & b_m \end{bmatrix}$$

augmented matrix

Recall: Linear Systems (Pictorially)



Recall: Number of Solutions

zero the system is inconsistent

one the system has a unique solution

many the system has infinity solutions

Recall: Number of Solutions

zero the system is inconsistent

one the system has a unique solution

many the system has infinity solutions

These are the **only** options

Motivating Questions

What matrices "represent solutions"? (which have solutions that are easy to "read off"?)

How does the number of solutions affect the shape of these matrix?

How do we use row operations to get to those matrices?

Motivating Questions

echelon forms

What matrices "represent solutions"? (which have solutions that are easy to "read off"?)

How does the number of solutions affect the shape of these matrix?

How do we use row operations to get to those matrices?

Unique Solution Case

Unique Solution Case

$$\begin{bmatrix} 2 & -3 & 5 & 11 \\ 2 & -1 & 13 & 39 \\ 1 & -1 & 5 & 14 \end{bmatrix} \sim \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \end{bmatrix}$$

$$x = 1$$
$$y = 2$$
$$z = 3$$

Unique Solution Case

$$\begin{bmatrix} 2 & -3 & 5 & 11 \\ 2 & -1 & 13 & 39 \\ 1 & -1 & 5 & 14 \end{bmatrix} \sim \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \end{bmatrix}$$

$$x = 1$$
$$y = 2$$
$$z = 3$$

Like all the examples we've seen so far

The Identity Matrix

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The Identity Matrix

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

1s along the diagonal

0s elsewhere

Unique Solution Case

coefficient matrix

$$\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \end{bmatrix}$$

a system of linear equations whose **coefficient matrix** is the identity matrix represents a unique solution

No Solution Case

Example

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 2 \\ 1 & 2 & 3 & 4 \end{bmatrix}$$

No Solution Case

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 2 \\ 1 & 2 & 3 & 4 \end{bmatrix} \sim \begin{bmatrix} 1 & 2 & 3 & 4 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

No Solution Case

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 2 \\ 1 & 2 & 3 & 4 \end{bmatrix} \sim \begin{bmatrix} 1 & 2 & 3 & 4 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

two parallel
planes

No Solution Case

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 2 \\ 1 & 2 & 3 & 4 \end{bmatrix} \sim \begin{bmatrix} 1 & 2 & 3 & 4 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

two parallel planes

row representing $0 = 1$

No Solution Case

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

row representing $0 = 1$

a system with no solutions can be reduced to a matrix with the row

$$0\ 0\ \ldots\ 0\ 1$$

Infinite Solution Case

Example

$$\begin{bmatrix} 2 & 4 & 2 & 14 \\ 1 & 7 & 1 & 12 \end{bmatrix}$$

demo
(plane intersection)

Infinite Solution Case

$$\begin{bmatrix} 2 & 4 & 2 & 14 \\ 1 & 7 & 1 & 12 \end{bmatrix} \sim \begin{bmatrix} 1 & 0 & 1 & 2 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

Infinite Solution Case

$$\begin{bmatrix} 2 & 4 & 2 & 14 \\ 1 & 7 & 1 & 12 \end{bmatrix} \sim \begin{bmatrix} 1 & 0 & 1 & 2 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

$$x_1 + x_3 = 2$$

$$x_2 = 1$$

Infinite Solution Case

$$\begin{bmatrix} 2 & 4 & 2 & 14 \\ 1 & 7 & 1 & 12 \end{bmatrix} \sim \begin{bmatrix} 1 & 0 & 1 & 2 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

$$x_1 + x_3 = 2$$

$$x_2 = 1$$

a system with infinity solutions can be reduced to a system which leaves a variable <u>unrestricted</u>

Infinite Solution Case

$$x_1 + x_3 = 2$$
$$x_2 = 1$$

it doesn't matter what x_3 is if we want to satisfy this system of equations

$$x_1 = 2$$
$$x_2 = 1$$
$$x_3 = 0$$

Infinite Solution Case

$$x_1 + x_3 = 2$$
$$x_2 = 1$$

it doesn't matter what x_3 is if we want to satisfy this system of equations

$$x_1 = 1.5$$
$$x_2 = 1$$
$$x_3 = 0.5$$

Infinite Solution Case

$$x_1 + x_3 = 2$$
$$x_2 = 1$$

it doesn't matter what x_3 is if we want to satisfy this system of equations

$$x_1 = 20$$
$$x_2 = 1$$
$$x_3 = -18$$

Infinite Solution Case

$$x_1 + x_3 = 2$$
$$x_2 = 1$$

it doesn't matter what x_3 is if we want to satisfy this system of equations

$x_1 = 2 - x_3$

$x_2 = 1$

x_3 is free

Infinite Solution Case

$$x_1 + x_3 = 2$$
$$x_2 = 1$$

it doesn't matter what x_3 is if we want to satisfy this system of equations

$x_1 = 2 - x_3$
$x_2 = 1$
x_3 is free

general form

In Sum

none	reduces to a system with the equation $0 = 1$
one	reduces to a system whose coefficient matrix is the identity matrix
infinity	reduces to a system which leaves a variable unrestricted

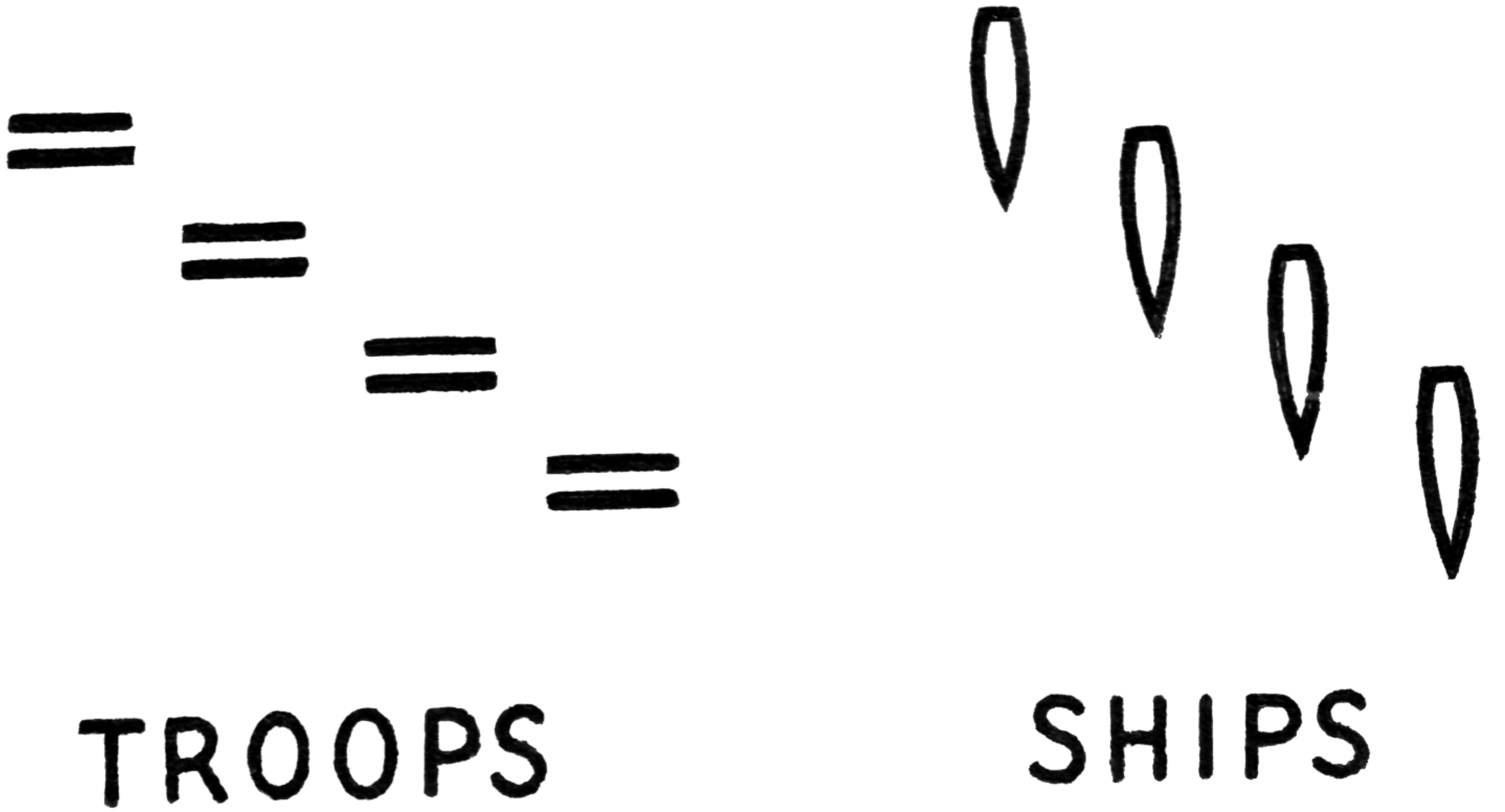
In Sum

none	reduces to a system with the equation $0 = 1$
one	reduces to a system whose coefficient matrix is the identity matrix
infinity	reduces to a system which leaves a variable unrestricted

Ideally, we want one *form* that handles all three cases

Echelon Form

The Picture (and a bit of history)



https://commons.wikimedia.org/wiki/File:Echelon_1_(PSF).png

Echelon Form (Pictorially)

$$\begin{bmatrix} 0 & \blacksquare & * & * & * & * & * & * & * & * \\ 0 & 0 & 0 & \blacksquare & * & * & * & * & * & * \\ 0 & 0 & 0 & 0 & \blacksquare & * & * & * & * & * \\ 0 & 0 & 0 & 0 & 0 & \blacksquare & * & * & * & * \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \blacksquare & * \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

■ = nonzero, * = anything

Leading Entries

Definition. the ***leading entry*** of a row is the first nonzero value

$$\begin{bmatrix} 1 & 2 & 3 \\ 0 & -3 & 3 \\ 0 & 0 & 0 \\ 1 & -1 & 10 \end{bmatrix}$$

← no leading entry

Echelon Form

Echelon Form

Definition. A matrix is in ***echelon form*** if

Echelon Form

Definition. A matrix is in ***echelon form*** if

1. The leading entry of each row appears to the right of the leading entry above it

Echelon Form

Definition. A matrix is in ***echelon form*** if

1. The leading entry of each row appears to the right of the leading entry above it
2. Every all-zeros row appears below any non-zero rows

Echelon Form (Pictorially)

$$\begin{bmatrix} 0 & \blacksquare & * & * & * & * & * & * & * & * \\ 0 & 0 & 0 & \blacksquare & * & * & * & * & * & * \\ 0 & 0 & 0 & 0 & \blacksquare & * & * & * & * & * \\ 0 & 0 & 0 & 0 & 0 & \blacksquare & * & * & * & * \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \blacksquare & * \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

■ = nonzero, * = anything

Echelon Form (Pictorially)

next leading entry to the right

$$\begin{bmatrix} 0 & \blacksquare & * & * & * & * & * & * & * & * \\ 0 & 0 & 0 & \blacksquare & * & * & * & * & * & * \\ 0 & 0 & 0 & 0 & \blacksquare & * & * & * & * & * \\ 0 & 0 & 0 & 0 & 0 & \blacksquare & * & * & * & * \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \blacksquare & * \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

all-zero rows at the bottom

$\blacksquare$ = nonzero, * = anything

Question

Is the identity matrix in echelon form?

Answer: Yes

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

the leading entries of each row appears to the right of the leading entry above it

it has no all–zero rows

Question

Is this matrix in echelon form?

$$\begin{bmatrix} 2 & 3 & -8 \\ 0 & 1 & 2 \\ 0 & 2 & 0 \end{bmatrix}$$

Answer: No

$$\begin{bmatrix} 2 & 3 & -8 \\ 0 & 1 & 2 \\ 0 & 2 & 0 \end{bmatrix}$$

The leading entry of the least row is not to the right of the leading entry of the second row

What's special about Echelon forms?

Theorem. Let A be the augmented matrix of an inconsistent linear system. If $A \sim B$ and B is in echelon form then B has the row

$$[0\ 0\ \ldots\ 0\ 0\ \blacksquare]$$

What's special about Echelon forms?

Theorem. Let A be the augmented matrix of an inconsistent linear system. If $A \sim B$ and B is in echelon form then B has the row

$$[0\ 0\ \ldots\ 0\ 0\ \blacksquare]$$

If all we care about is consistency then we just need to find an echelon form

Example

$$x - 2z = 4$$
$$-x + y + 5z = -3$$
$$x + 2y + 4z = 7$$

The Problem with Echelon Forms

If our system *is* consistent, we can't get a solution quite yet.

The Problem with Echelon Forms

If our system *is* consistent, we can't get a solution quite yet.

We need to simplify our matrix a bit more until it "represents" a solution

Reduced Echelon Form

Row-Reduced Echelon Form (RREF)

Definition. A matrix is in ***(row-)reduced echelon form*** if

1. The leading entry of each row appears to the right of the leading entry above it
2. Every all-zeros row appears below any non-zero rows
3. The leading entries of non-zero rows are 1
4. the leading entries are the only non-zero entries of their columns

Reduced Echelon Form (Pictorially)

$$\begin{bmatrix} 0 & 1 & * & 0 & 0 & 0 & * & * & 0 & * \\ 0 & 0 & 0 & 1 & 0 & 0 & * & * & 0 & * \\ 0 & 0 & 0 & 0 & 1 & 0 & * & * & 0 & * \\ 0 & 0 & 0 & 0 & 0 & 1 & * & * & 0 & * \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & * \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Reduced Echelon Form (Pictorially)

leading entries are 1

$$\begin{bmatrix} 0 & 1 & * & 0 & 0 & 0 & * & * & 0 & * \\ 0 & 0 & 0 & 1 & 0 & 0 & * & * & 0 & * \\ 0 & 0 & 0 & 0 & 1 & 0 & * & * & 0 & * \\ 0 & 0 & 0 & 0 & 0 & 1 & * & * & 0 & * \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & * \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

other column entries are 0

Reduced Echelon Form (A Simple Example)

$$x_1 + x_3 = 2$$
$$x_2 = 1$$

Reduced Echelon Form (A Simple Example)

$$x_1 + x_3 = 2$$
$$x_2 = 1$$

$x_1 = 2 - x_3$

$x_2 = 1$

x_3 is free

The Fundamental Points

The Fundamental Points

Point 1. we can "read off" the solutions of a system of linear equations from its RREF

The Fundamental Points

Point 1. we can "read off" the solutions of a system of linear equations from its RREF

Point 2. *every* matrix is row equivalent to a <u>unique</u> matrix in reduced echelon form

How-To: Solving a System of Linear Equations

How-To: Solving a System of Linear Equations

1. Write your system as an augmented matrix

How-To: Solving a System of Linear Equations

1. Write your system as an augmented matrix

2. Find the RREF of that matrix

How-To: Solving a System of Linear Equations

1. Write your system as an augmented matrix

2. Find the RREF of that matrix

3. Read off the solution from the RREF

How-To: Solving a System of Linear Equations

1. Write your system as an augmented matrix

2. Find the RREF of that matrix

3. Read off the solution from the RREF

Our next topic

demo
(a.rref())

What's special about RREF?

Every leading variable can be written in terms of only non-leading variables.

$$\begin{bmatrix} 0 & 1 & * & 0 & 0 & 0 & * & * & 0 & * \\ 0 & 0 & 0 & 1 & 0 & 0 & * & * & 0 & * \\ 0 & 0 & 0 & 0 & 1 & 0 & * & * & 0 & * \\ 0 & 0 & 0 & 0 & 0 & 1 & * & * & 0 & * \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & * \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 1 & 2 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

Example

$$\begin{bmatrix} 0 & 1 & 2 & 0 & 0 & 0 & 3 & -1 & 0 & 2 \\ 0 & 0 & 0 & 1 & 0 & 0 & 4 & 5 & 0 & 20 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 16 \\ 0 & 0 & 0 & 0 & 0 & 1 & 12 & 3 & 0 & -4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -8 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Why we care about Reduced Echelon Forms?

Why we care about Reduced Echelon Forms?

the goal of back–substitution is to reduce an echelon form matrix to a ***reduced*** *echelon form*

Why we care about Reduced Echelon Forms?

*the goal of back–substitution is to reduce an echelon form matrix to a **reduced** echelon form*

*the goal of Gaussian elimination is to reduce an **augmented** matrix to a **reduced** echelon form*

Why we care about Reduced Echelon Forms?

*the goal of back–substitution is to reduce an echelon form matrix to a **reduced** echelon form*

*the goal of Gaussian elimination is to reduce an **augmented** matrix to a **reduced** echelon form*

reduced echelon forms describe solutions to linear equations

General-Form Solutions

What's Left?

What's Left?

We know how to use an RREF to see if a system is inconsistent.

What's Left?

We know how to use an RREF to see if a system is inconsistent.

We know how to use an RREF to read of a unique solution, if there is one.

What's Left?

We know how to use an RREF to see if a system is inconsistent.

We know how to use an RREF to read of a unique solution, if there is one.

But how do we characterize *all* solutions in the infinite solution case?

Basic and Free Variables

Basic and Free Variables

Definition. a ***pivot position*** (i, j) in a matrix is the position of a leading entry in it's reduced echelon form

Basic and Free Variables

Definition. a ***pivot position*** (i, j) in a matrix is the position of a leading entry in it's reduced echelon form

Definition. A variable is ***basic*** if its column has a pivot position (this is called a ***pivot column***). It is ***free*** otherwise.

Basic and Free Variables

Definition. a ***pivot position*** (i, j) in a matrix is the position of a leading entry in it's reduced echelon form

Definition. A variable is ***basic*** if its column has a pivot position (this is called a ***pivot column***). It is ***free*** otherwise.

$$\begin{bmatrix} 1 & 0 & 1 & 2 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

Basic and Free Variables

Definition. a ***pivot position*** (i, j) in a matrix is the position of a leading entry in it's reduced echelon form

Definition. A variable is ***basic*** if its column has a pivot position (this is called a ***pivot column***). It is ***free*** otherwise.

x_1 is basic

x_2 is basic

$$\begin{bmatrix} 1 & 0 & 1 & 2 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

x_3 is free

Solutions of Reduced Echelon Forms

the row i of a <u>pivot position</u> describes the <u>value of x_i in a solution</u> to the system, in terms of the free variables

$$\begin{bmatrix} 1 & 0 & 1 & 2 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

How-To: General Form Solution

$$\begin{bmatrix} 1 & 0 & 1 & 2 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

$x_1 = 2 - x_3$

$x_2 = 1$

x_3 `is free`

How-To: General Form Solution

$$\begin{bmatrix} 1 & 0 & 1 & 2 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

$x_1 = 2 - x_3$

$x_2 = 1$

x_3 is free

1. For each pivot position (i, j), isolate x_j in the equation in row i

How-To: General Form Solution

$$\begin{bmatrix} 1 & 0 & 1 & 2 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

$x_1 = 2 - x_3$

$x_2 = 1$

x_3 is free

1. For each pivot position (i, j), isolate x_j in the equation in row i

2. If x_i is not in a pivot column then write

x_i **is free**

Example

$$\begin{bmatrix} 1 & 2 & 0 & -2 & 4 \\ 0 & 0 & 1 & 3 & 5 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Question

$$\begin{bmatrix} 1 & 0 & 0 & 3 & 1 \\ 0 & 0 & 1 & 2 & 4 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Circle the pivot positions, highlight the pivot rows.

Which variables are free? Which are basic?

Write down a solution in general form for this reduced echelon form matrix.

*Write down a **particular** solution given the general form.*

Answer

$$\begin{bmatrix} 1 & 0 & 0 & 3 & 1 \\ 0 & 0 & 1 & 2 & 4 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Defining the Gaussian Elimination (GE) Algorithm

At a High Level

At a High Level

```
eliminations + back–substitution
```

At a High Level

`eliminations + back–substitution`

`we've already done this`

At a High Level

eliminations + back–substitution

we've already done this

but we'll take one step further and write down the algorithm as pseudocode

At a High Level

eliminations + back–substitution

we've already done this

but we'll take one step further and write down the algorithm as <u>pseudocode</u>

Keep in mind. How do we turn our intuitions into a formal procedure?

A Word of Warning

A Word of Warning

```
The details of Gaussian elimination are tricky.
```

A Word of Warning

The details of Gaussian elimination are tricky.

The goal is not to understand it entirely, but to get enough intuition to emulate it.

A Word of Warning

The details of Gaussian elimination are tricky.

The goal is not to understand it entirely, but to get enough intuition to emulate it.

You should roughly use Gaussian Elimination when solving a system by hand.

demo
(Step-throughs)

The Algorithm

Gaussian Elimination (Specification)

```
FUNCTION GE(A):
  # INPUT: m × n matrix A
  # OUTPUT: equivalent m × n RREF matrix
  ...
```

Gaussian Elimination (High Level)

```
FUNCTION fwd_elim(A):
  # INPUT: m × n matrix A
  # OUTPUT: equivalent m × n echelon form matrix
  ...

FUNCTION back_sub(A):
  # INPUT: m × n echelon form matrix A
  # OUTPUT: equivalent m × n RREF matrix
  ...

FUNCTION GE(A):
  RETURN back_sub(fwd_elim(A))
```

Elimination Stage

Elimination Stage (High Level)

Elimination Stage (High Level)

Input: matrix A of size $m \times n$

Output: echelon form of A

Elimination Stage (High Level)

Input: matrix A of size $m \times n$

Output: echelon form of A

starting at the top left and move down, find a leading entry and eliminate it from latter equations

Edge cases

Edge cases

What if the first equation doesn't have the variable x_1?

Edge cases

What if the first equation doesn't have the variable x_1?

Swap rows with an equation that does.

Edge cases

What if the first equation doesn't have the variable x_1?

Swap rows with an equation that does.

What if *none* of the equations have the variable x_1?

Edge cases

What if the first equation doesn't have the variable x_1?

Swap rows with an equation that does.

What if *none* of the equations have the variable x_1?

Find the *leftmost* variable which appears in *any* of the remaining equations.

Elimination Stage (Pseudocode)

Elimination Stage (Pseudocode)

```
FUNCTION fwd_elim(A):
```

Elimination Stage (Pseudocode)

```
FUNCTION fwd_elim(A):
  FOR [i from 1 to m]: # for each row from top to bottom
```

Elimination Stage (Pseudocode)

```
FUNCTION fwd_elim(A):
  FOR [i from 1 to m]: # for each row from top to bottom
    IF [rows i...m are all-zeros]: # if remaining rows are zero
```

Elimination Stage (Pseudocode)

```
FUNCTION fwd_elim(A):
  FOR [i from 1 to m]: # for each row from top to bottom
    IF [rows i...m are all-zeros]: # if remaining rows are zero
      RETURN A
```

Elimination Stage (Pseudocode)

```
FUNCTION fwd_elim(A):
  FOR [i from 1 to m]: # for each row from top to bottom
    IF [rows i...m are all-zeros]: # if remaining rows are zero
      RETURN A
    ELSE:
```

Elimination Stage (Pseudocode)

```
FUNCTION fwd_elim(A):
  FOR [i from 1 to m]: # for each row from top to bottom
    IF [rows i...m are all-zeros]: # if remaining rows are zero
      RETURN A
    ELSE:
      (j, k) ← [position of leftmost entry in the rows i...m]
```

Elimination Stage (Pseudocode)

```
FUNCTION fwd_elim(A):
  FOR [i from 1 to m]: # for each row from top to bottom
    IF [rows i...m are all-zeros]: # if remaining rows are zero
      RETURN A
    ELSE:
      (j, k) ← [position of leftmost entry in the rows i...m]
      [swap row i and row j]
```

Elimination Stage (Pseudocode)

```
FUNCTION fwd_elim(A):
  FOR [i from 1 to m]: # for each row from top to bottom
    IF [rows i...m are all-zeros]: # if remaining rows are zero
      RETURN A
    ELSE:
      (j, k) ← [position of leftmost entry in the rows i...m]
      [swap row i and row j]
      FOR [l from i + 1 to m]: # for all remaining rows
```

Elimination Stage (Pseudocode)

```
FUNCTION fwd_elim(A):
  FOR [i from 1 to m]: # for each row from top to bottom
    IF [rows i...m are all-zeros]: # if remaining rows are zero
      RETURN A
    ELSE:
      (j, k) ← [position of leftmost entry in the rows i...m]
      [swap row i and row j]
      FOR [l from i + 1 to m]: # for all remaining rows
        [zero out A[l, k] using a replacement operation]
```

Elimination Stage (Pseudocode)

```
FUNCTION fwd_elim(A):
  FOR [i from 1 to m]: # for each row from top to bottom
    IF [rows i...m are all-zeros]: # if remaining rows are zero
      RETURN A
    ELSE:
      (j, k) ← [position of leftmost entry in the rows i...m]
      [swap row i and row j]
      FOR [l from i + 1 to m]: # for all remaining rows
        [zero out A[l, k] using a replacement operation]
  RETURN A
```

Elimination Stage (Example)

$$\begin{bmatrix} 0 & 3 & -6 & 6 & 4 & -5 \\ 3 & -7 & 8 & -5 & 8 & 9 \\ 3 & -9 & 12 & -9 & 6 & 15 \end{bmatrix}$$

Elimination Stage (Example)

$$\begin{bmatrix} 0 & 3 & -6 & 6 & 4 & -5 \\ 3 & -7 & 8 & -5 & 8 & 9 \\ 3 & -9 & 12 & -9 & 6 & 15 \end{bmatrix}$$

leftmost
nonzero
entry

Elimination Stage (Example)

leftmost nonzero entry

$$\begin{bmatrix} 0 & 3 & -6 & 6 & 4 & -5 \\ 3 & -7 & 8 & -5 & 8 & 9 \\ 3 & -9 & 12 & -9 & 6 & 15 \end{bmatrix}$$

Swap R_1 and R_3

Elimination Stage (Example)

$$\begin{bmatrix} 3 & -9 & 12 & -9 & 6 & 15 \\ 3 & -7 & 8 & -5 & 8 & 9 \\ 0 & 3 & -6 & 6 & 4 & -5 \end{bmatrix}$$

Elimination Stage (Example)

next entry
to zero

$$\begin{bmatrix} 3 & -9 & 12 & -9 & 6 & 15 \\ 3 & -7 & 8 & -5 & 8 & 9 \\ 0 & 3 & -6 & 6 & 4 & -5 \end{bmatrix}$$

Elimination Stage (Example)

next entry
to zero

$$\begin{bmatrix} 3 & -9 & 12 & -9 & 6 & 15 \\ 3 & -7 & 8 & -5 & 8 & 9 \\ 0 & 3 & -6 & 6 & 4 & -5 \end{bmatrix}$$

$$R_3 \leftarrow R_3 - R_1$$

Elimination Stage (Example)

$$\begin{bmatrix} 3 & -9 & 12 & -9 & 6 & 15 \\ 0 & 2 & -4 & 4 & 2 & -6 \\ 0 & 3 & -6 & 6 & 4 & -5 \end{bmatrix}$$

Elimination Stage (Example)

leftmost
nonzero
entry

$$\begin{bmatrix} 3 & -9 & 12 & -9 & 6 & 15 \\ 0 & 2 & -4 & 4 & 2 & -6 \\ 0 & 3 & -6 & 6 & 4 & -5 \end{bmatrix}$$

Elimination Stage (Example)

leftmost
nonzero
entry

$$\begin{bmatrix} 3 & -9 & 12 & -9 & 6 & 15 \\ 0 & 2 & -4 & 4 & 2 & -6 \\ 0 & 3 & -6 & 6 & 4 & -5 \end{bmatrix}$$

swap R_2 with R_2

Elimination Stage (Example)

$$\begin{bmatrix} 3 & -9 & 12 & -9 & 6 & 15 \\ 0 & 2 & -4 & 4 & 2 & -6 \\ 0 & 3 & -6 & 6 & 4 & -5 \end{bmatrix}$$

Elimination Stage (Example)

$$\begin{bmatrix} 3 & -9 & 12 & -9 & 6 & 15 \\ 0 & 2 & -4 & 4 & 2 & -6 \\ 0 & 3 & -6 & 6 & 4 & -5 \end{bmatrix}$$

next entry to zero

Elimination Stage (Example)

next entry to zero

$$\begin{bmatrix} 3 & -9 & 12 & -9 & 6 & 15 \\ 0 & 2 & -4 & 4 & 2 & -6 \\ 0 & 3 & -6 & 6 & 4 & -5 \end{bmatrix}$$

$$R_3 \leftarrow R_3 - \frac{3R_2}{2}$$

Elimination Stage (Example)

$$\begin{bmatrix} 3 & -9 & 12 & -9 & 6 & 15 \\ 0 & 2 & -4 & 4 & 2 & -6 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

Elimination Stage (Example)

$$\begin{bmatrix} 3 & -9 & 12 & -9 & 6 & 15 \\ 0 & 2 & -4 & 4 & 2 & -6 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

leftmost
nonzero
entry

Elimination Stage (Example)

leftmost nonzero entry

$$\begin{bmatrix} 3 & -9 & 12 & -9 & 6 & 15 \\ 0 & 2 & -4 & 4 & 2 & -6 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

swap R_3 with R_3

Elimination Stage (Example)

$$\begin{bmatrix} 3 & -9 & 12 & -9 & 6 & 15 \\ 0 & 2 & -4 & 4 & 2 & -6 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

Elimination Stage (Example)

$$\begin{bmatrix} 3 & -9 & 12 & -9 & 6 & 15 \\ 0 & 2 & -4 & 4 & 2 & -6 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

done with elimination stage
going to back substitution stage

Back Substitution Stage

Back Substitution Stage (High Level)

Back Substitution Stage (High Level)

Input: matrix A of size $m \times n$ in echelon form

Output: reduced echelon form of A

Back Substitution Stage (High Level)

Input: matrix A of size $m \times n$ in echelon form

Output: reduced echelon form of A

scale pivot positions and eliminate the variables for that column from the other equations

Back Substitution (Psuedocode)

Back Substitution (Psuedocode)

```
FUNCTION back_sub(A):
```

Back Substitution (Psuedocode)

```
FUNCTION back_sub(A):
  FOR [i from 1 to m]: # for each row from top to bottom
```

Back Substitution (Psuedocode)

```
FUNCTION back_sub(A):
  FOR [i from 1 to m]: # for each row from top to bottom
    IF [row i has a leading entry]:
```

Back Substitution (Psuedocode)

```
FUNCTION back_sub(A):
  FOR [i from 1 to m]: # for each row from top to bottom
    IF [row i has a leading entry]:
      j ← index of leading entry of row i
```

Back Substitution (Psuedocode)

```
FUNCTION back_sub(A):
  FOR [i from 1 to m]: # for each row from top to bottom
    IF [row i has a leading entry]:
      j ← index of leading entry of row i
      R_i(A) ← R_i(A) / A[i, j] # divide by leading entry
```

Back Substitution (Psuedocode)

```
FUNCTION back_sub(A):
  FOR [i from 1 to m]: # for each row from top to bottom
    IF [row i has a leading entry]:
      j ← index of leading entry of row i
      R_i(A) ← R_i(A) / A[i, j] # divide by leading entry
      FOR [k from 1 to i - 1]: # for the rows above the current one
```

Back Substitution (Psuedocode)

```
FUNCTION back_sub(A):
  FOR [i from 1 to m]: # for each row from top to bottom
    IF [row i has a leading entry]:
      j ← index of leading entry of row i
      R_i(A) ← R_i(A) / A[i, j] # divide by leading entry
      FOR [k from 1 to i − 1]: # for the rows above the current one
        R_k(A) ← R_k(A) − R[k, j] ∗ R_i(A)
        # zero out R[k, j] above the leading entry
```

Back Substitution (Psuedocode)

```
FUNCTION back_sub(A):
  FOR [i from 1 to m]: # for each row from top to bottom
    IF [row i has a leading entry]:
      j ← index of leading entry of row i
      R_i(A) ← R_i(A) / A[i, j] # divide by leading entry
      FOR [k from 1 to i − 1]: # for the rows above the current one
        R_k(A) ← R_k(A) − R[k, j] ∗ R_i(A)
        # zero out R[k, j] above the leading entry
  RETURN A
```

You will have to implement this part in HW2...

Gaussian Elimination (Example)

$$\begin{bmatrix} 3 & -9 & 12 & -9 & 6 & 15 \\ 0 & 2 & -4 & 4 & 2 & -6 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

Gaussian Elimination (Example)

pivot position

$$\begin{bmatrix} 3 & -9 & 12 & -9 & 6 & 15 \\ 0 & 2 & -4 & 4 & 2 & -6 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

Gaussian Elimination (Example)

pivot
position

$$\begin{bmatrix} 3 & -9 & 12 & -9 & 6 & 15 \\ 0 & 2 & -4 & 4 & 2 & -6 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

$R_1 \leftarrow R_1 \,/\, 3$

Gaussian Elimination (Example)

$$\begin{bmatrix} 1 & -3 & 4 & -3 & 2 & 5 \\ 0 & 2 & -4 & 4 & 2 & -6 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

Gaussian Elimination (Example)

pivot position

$$\begin{bmatrix} 1 & -3 & 4 & -3 & 2 & 5 \\ 0 & \boxed{2} & -4 & 4 & 2 & -6 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

Gaussian Elimination (Example)

pivot position

$$\begin{bmatrix} 1 & -3 & 4 & -3 & 2 & 5 \\ 0 & \boxed{2} & -4 & 4 & 2 & -6 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

$R_2 \leftarrow R_2 \ / \ 2$

Gaussian Elimination (Example)

$$\begin{bmatrix} 1 & -3 & 4 & -3 & 2 & 5 \\ 0 & 1 & -2 & 2 & 1 & -3 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

Gaussian Elimination (Example)

next entry to zero

$$\begin{bmatrix} 1 & -3 & 4 & -3 & 2 & 5 \\ 0 & 1 & -2 & 2 & 1 & -3 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

Gaussian Elimination (Example)

next entry
to zero

$$\begin{bmatrix} 1 & -3 & 4 & -3 & 2 & 5 \\ 0 & 1 & -2 & 2 & 1 & -3 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

$$R_1 \leftarrow R_1 + 3R_2$$

Gaussian Elimination (Example)

$$\begin{bmatrix} 1 & 0 & -2 & 3 & 5 & -4 \\ 0 & 1 & -2 & 2 & 1 & -3 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

Gaussian Elimination (Example)

$$\begin{bmatrix} 1 & 0 & -2 & 3 & 5 & -4 \\ 0 & 1 & -2 & 2 & 1 & -3 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

pivot
position

Gaussian Elimination (Example)

$$\begin{bmatrix} 1 & 0 & -2 & 3 & 5 & -4 \\ 0 & 1 & -2 & 2 & 1 & -3 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

pivot position

$R_3 \leftarrow R_3 \ / \ 1$

Gaussian Elimination (Example)

$$\begin{bmatrix} 1 & 0 & -2 & 3 & 5 & -4 \\ 0 & 1 & -2 & 2 & 1 & -3 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

Gaussian Elimination (Example)

next entry
to zero

$$\begin{bmatrix} 1 & 0 & -2 & 3 & 5 & -4 \\ 0 & 1 & -2 & 2 & 1 & -3 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

Gaussian Elimination (Example)

next entry to zero

$$\begin{bmatrix} 1 & 0 & -2 & 3 & 5 & -4 \\ 0 & 1 & -2 & 2 & 1 & -3 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

$$R_2 \leftarrow R_2 - R_1$$

Gaussian Elimination (Example)

$$\begin{bmatrix} 1 & 0 & -2 & 3 & 5 & -4 \\ 0 & 1 & -2 & 2 & 0 & -7 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

Gaussian Elimination (Example)

next entry
to zero

$$\begin{bmatrix} 1 & 0 & -2 & 3 & 5 & -4 \\ 0 & 1 & -2 & 2 & 0 & -7 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

Gaussian Elimination (Example)

next entry
to zero

$$\begin{bmatrix} 1 & 0 & -2 & 3 & 5 & -4 \\ 0 & 1 & -2 & 2 & 0 & -7 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

$$R_1 \leftarrow R_1 - 5R_3$$

Gaussian Elimination (Example)

$$\begin{bmatrix} 1 & 0 & -2 & 3 & 0 & -24 \\ 0 & 1 & -2 & 2 & 0 & -7 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

Gaussian Elimination (Example)

$$\begin{bmatrix} 1 & 0 & -2 & 3 & 0 & -24 \\ 0 & 1 & -2 & 2 & 0 & -7 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

done with back substitution phase

Gaussian Elimination (Example)

$$x_1 = (-24) + 2x_3 - 3x_4$$

$$x_2 = (-7) + 2x_3 - 2x_4$$

x_3 `is free`

x_4 `is free`

$$x_5 = 4$$

Gaussian Elimination (Example)

$$\begin{bmatrix} 1 & 0 & -2 & 3 & 0 & -24 \\ 0 & 1 & -2 & 2 & 0 & -7 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

How-To: Solving a System of Linear Equations

How-To: Solving a System of Linear Equations

1. Write your system as an augmented matrix

How-To: Solving a System of Linear Equations

1. Write your system as an augmented matrix

2. Find the RREF of that matrix

How-To: Solving a System of Linear Equations

1. Write your system as an augmented matrix

2. Find the RREF of that matrix

3. Read off the solution from the RREF

How-To: Solving a System of Linear Equations

1. Write your system as an augmented matrix

2. Find the RREF of that matrix
   Gaussian elimination

3. Read off the solution from the RREF

Extra Topic:
Analyzing the Algorithm

Analyzing the Algorithm

Analyzing the Algorithm

`We will not use` $O(\cdot)$ `notation!`

Analyzing the Algorithm

We will not use $O(\cdot)$ notation!

For numerics, we care about number of **FL**oating-oint **OP**erations (FLOPs):

>> addition
>> subtraction
>> multiplication
>> division
>> square root

Analyzing the Algorithm

We will not use $O(\cdot)$ notation!

For numerics, we care about number of **FL**oating-oint **OP**erations (FLOPs):

- >> addition
- >> subtraction
- >> multiplication
- >> division
- >> square root

$2n$ vs. n is very different when $n \sim 10^{20}$

Dominant Terms

Dominant Terms

`that said, we don't care about` *`exact`* `bounds`

Dominant Terms

that said, we don't care about *exact* bounds

A function $f(n)$ is ***asymptotically equivalent*** to $g(n)$ if

$$\lim_{i \to \infty} \frac{f(i)}{g(i)} = 1$$

Dominant Terms

that said, we don't care about *exact* bounds

A function $f(n)$ is ***asymptotically equivalent*** to $g(n)$ if

$$\lim_{i \to \infty} \frac{f(i)}{g(i)} = 1$$

for polynomials, they are equivalent to their dominant term

Dominant Terms

the dominant term of a polynomial is the monomial with the highest degree

$$\lim_{i \to \infty} \frac{3x^3 + 100000x^2}{3x^3} = 1$$

$3x^3$ dominates the function even though the coefficient for x^2 is so large

Parameters

n : number of variables

m : number of equations (we will assume $m = n$)

$n+1$: number of rows in the augmented matrix

The Cost of a Row Operation

$$R_i \leftarrow R_i + aR_j$$

$n+1$ multiplications for the scaling

$n+1$ additions for the row additions

Tally: $2(n+1)$ FLOPS

Cost of First Iteration of Elimination

$$R_2 \leftarrow R_2 + a_2 R_1$$
$$R_3 \leftarrow R_3 + a_3 R_1$$
$$\vdots$$
$$R_n \leftarrow R_n + a_n R_1$$

`repeated row operations for each row except the first`

`Tally:` $\approx 2n(n+1)$ `FLOPS`

Rough Cost of Elimination

repeating this last process at most n times gives us a dominant term $2n^3$

we can give a better estimation...

Tally: $\approx 2n^2(n+1)$ FLOPS

Cost of Elimination

$$\begin{bmatrix} 0 & \blacksquare & * & * & * & * & * & * & * & * \\ 0 & 0 & 0 & \blacksquare & * & * & * & * & * & * \\ 0 & 0 & 0 & 0 & * & * & * & * & * & * \\ 0 & 0 & 0 & 0 & * & * & * & * & * & * \\ 0 & 0 & 0 & 0 & * & * & * & * & * & * \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

At iteration i, we're only interested in rows after i

And to the right of column i

Cost of Elimination

`Iteration 1:` $2n(n+1)$
`Iteraiton 2:` $2(n-1)n$
`Iteration 3:` $2(n-2)(n-1)$
$\vdots$ $+$

$$\sum_{k=1}^{n} 2k(k+1) \approx \frac{2n(n+1)(2n+1)}{6} \sim (2/3)n^3$$

`Tally:` $\sim (2/3)n^3$ `FLOPS`

Cost of Back Substitution

```
(Let's assume no free variables)

for each pivot, we only need to:

  >> zero out a position in 1 row (0 FLOPS)
  >> add a value to the last row (1 FLOP)
```

at most 1 FLOP per row per pivot $\sim n^2$

Tally: $\sim (2/3)n^3$ FLOPS

Cost of Gaussian Elimination

Tally: $\sim (2/3)n^3$ FLOPS

(dominated by elimination)

Summary

Echelon form "represent solutions"

General form solutions can be used to describe the infinite solution sets

Gaussian elimination uses forward elimination and back-substitution to solve linear equations in general