Storytime

Carl Friedrich Gauss

30 April 1777 - 23 February 1855

A German mathematician who had a tremendous impact on many areas of mathematics and physics.

But he did not discover Gaussian elimination…

This was already a standard technique for solving linear equations before he was born.

A story from Gauss' childhood

Gauss was misbehaving in primary school.

As a punishment, his teacher asked him sum the numbers from 1 to 100.

$$1 + 2 + 3 + \ldots + 50 + 51 + \ldots + 98 + 99 + 100$$

Gauss immediately replied with the answer 5050.

How did Gauss compute this so fast?

A story from Gauss' childhood

$$1 + 2 + 3 + \ldots + 50 + 51 + \ldots + 98 + 99 + 100$$

Most likely, Gauss paired up the numbers like this:

$$\begin{aligned} 1 + 100 &= 101 \\ 2 + 99 &= 101 \\ 3 + 98 &= 101 \\ &\vdots \\ 50 + 51 &= 101 \end{aligned}$$

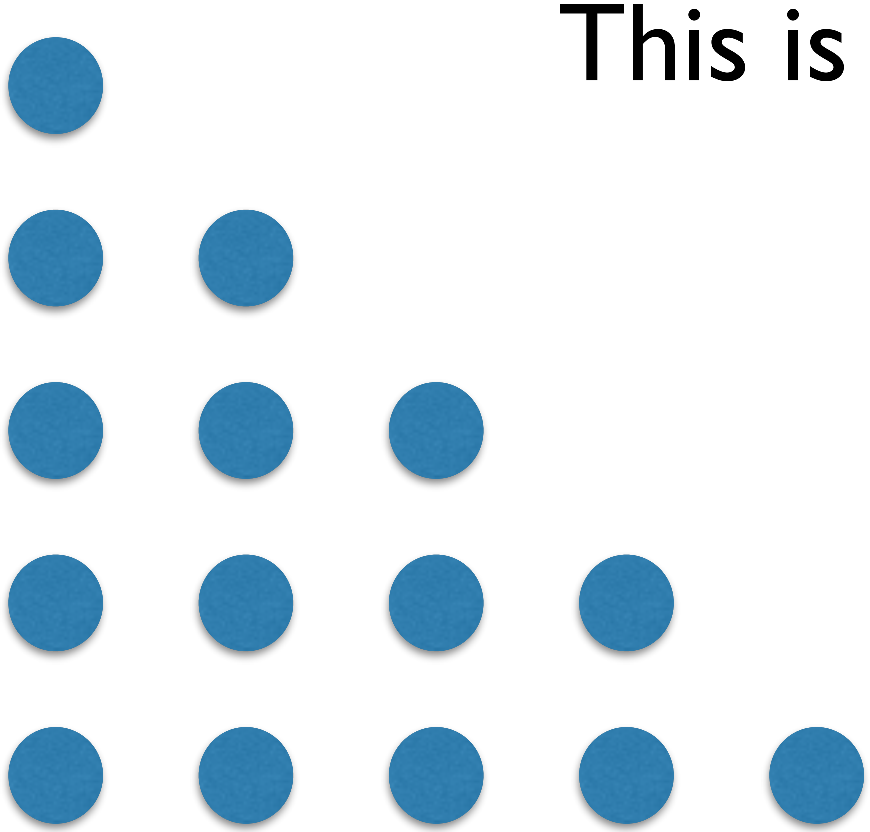
There are 50 pairs summing to 101.

The total is $50 \cdot 101 = 5050$.

General case

Let's look at the general question. What is the sum of the first n numbers?

$$1 + 2 + 3 + \ldots + n = \sum_{i=1}^{n} i = T_n$$



This is the number of dots in a triangle.

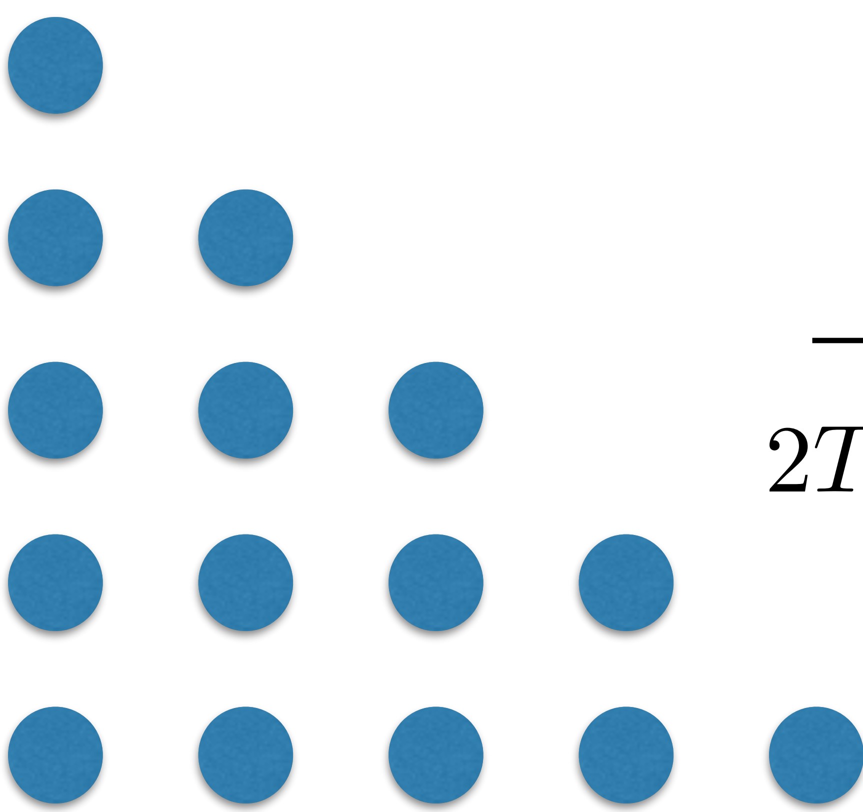
This sum is called a triangular number T_n .

General case

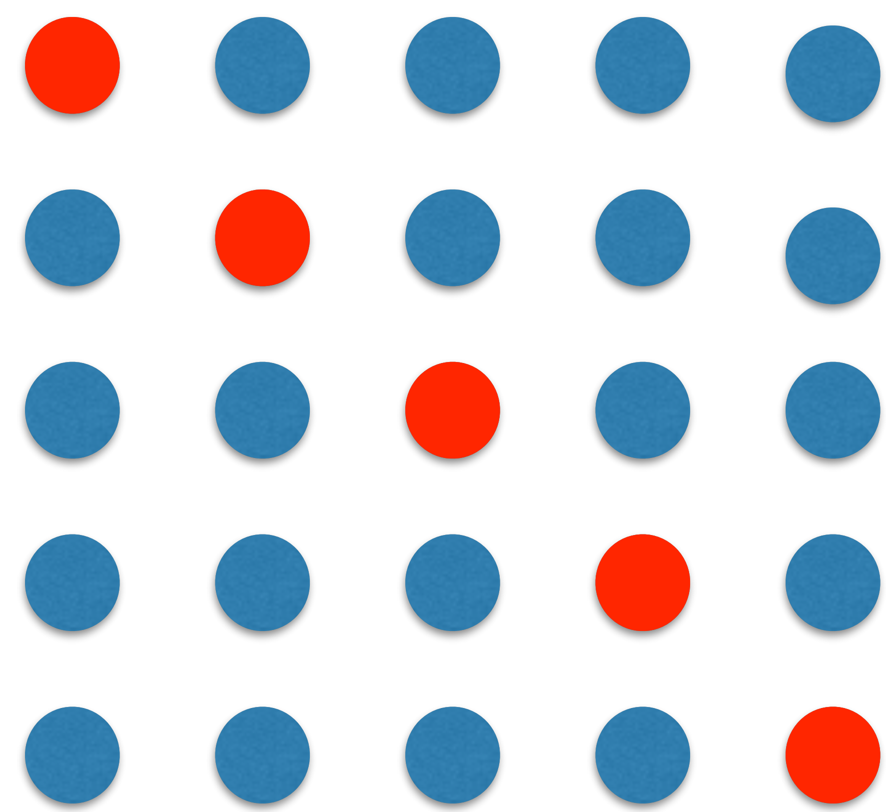
$$1 + 2 + 3 + \ldots + n \; = \sum_{i=1}^{n} i \; = T_n$$

If we double T_n we get a square, plus the diagonal.

$$T_n = \frac{n(n+1)}{2}$$



$$2T_n = n^2 + n$$



Next stop squares

Let's get back to some linear algebra.

Now say we want to compute a sum of squares.

$$S_n = 0^2 + 1^2 + 2^2 + 3^2 + \ldots + n^2 = \sum_{i=0}^{n} i^2$$

I tell you that $S_n = a + b \cdot n + c \cdot n^2 + d \cdot n^3$ for some values of a, b, c, d. Can you figure out what they are?

Next stop squares

We can do this by setting up a system of linear equations!

$$S_n = 0^2 + 1^2 + 2^2 + 3^2 + \ldots + n^2 = \sum_{i=0}^{n} i^2$$

Let's look at some values.

If $S_n = a + b \cdot n + c \cdot n^2 + d \cdot n^3$, this means:

$S_0 = 0$	$a = 0$
$S_1 = 1$	$a + b + c + d = 1$
$S_2 = 5$	$a + 2b + 4c + 8d = 5$
$S_3 = 14$	$a + 3b + 9c + 27d = 14$

Simplifying the system by removing a we have:

$$\begin{aligned} b + c + d &= 1 \\ 2b + 4c + 8d &= 5 \\ 3b + 9c + 27d &= 14 \end{aligned} \quad \xrightarrow[\substack{R_2' = R_2 - 2R_1 \\ R_3' = R_3 - 3R_1}]{} \quad \begin{aligned} b + c + d &= 1 \\ 2c + 6d &= 3 \\ 6c + 24d &= 11 \end{aligned}$$

$$\begin{aligned} b + c + d &= 1 \\ 2c + 6d &= 3 \\ 6c + 24d &= 11 \end{aligned} \quad \xrightarrow[R_3' = R_3 - 3R_2]{} \quad \begin{aligned} b + c + d &= 1 \\ 2c + 6d &= 3 \\ 6d &= 2 \end{aligned}$$

Now the system is upper triangular.

Full set of pivots implies a unique solution.

$$\begin{aligned} b + c + d &= 1 \\ 2c + 6d &= 3 \\ 6d &= 2 \end{aligned}$$

Back substitution:

$$d = \frac{1}{3}$$

$$2c + 2 = 3 \implies c = \frac{1}{2}$$

$$b + \frac{1}{2} + \frac{1}{3} = 1 \implies b = \frac{1}{6}$$

$$\boxed{\textbf{Conclusion:} \quad \sum_{i=0}^{n} i^2 = \frac{n}{6} + \frac{n^2}{2} + \frac{n^3}{3}}$$

Questions

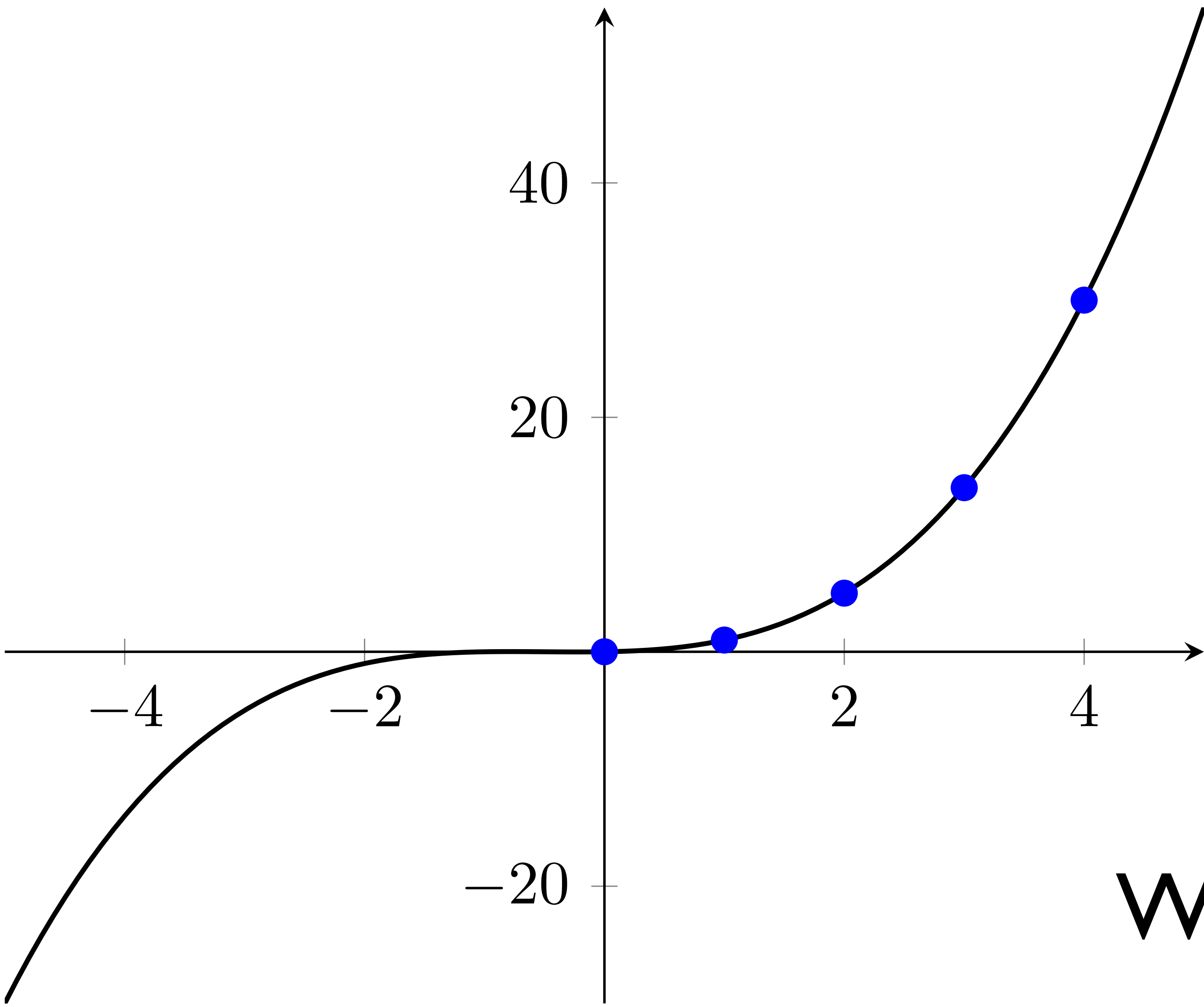
$$\sum_{i=0}^{n} i^2 = \frac{n}{6} + \frac{n^2}{2} + \frac{n^3}{3}$$

Is this a proof ?

Can you compute $\displaystyle\sum_{i=0}^{n} i^3$?

Polynomial Interpolation

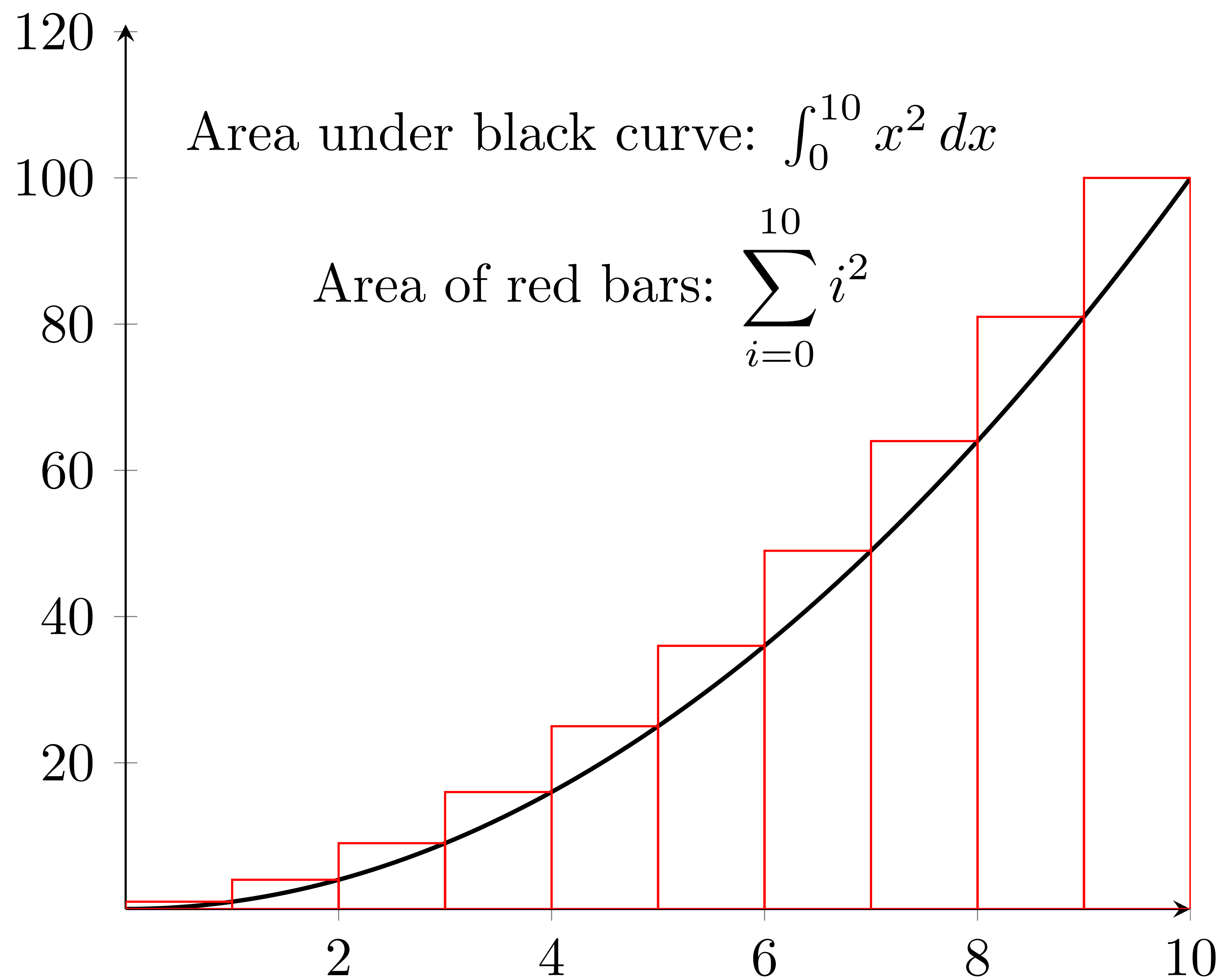
This example is an instance of a polynomial interpolation problem.



Given a set of data points, is there a degree 3 polynomial going through them?

We will study this problem more generally later on.

Connection with calculus



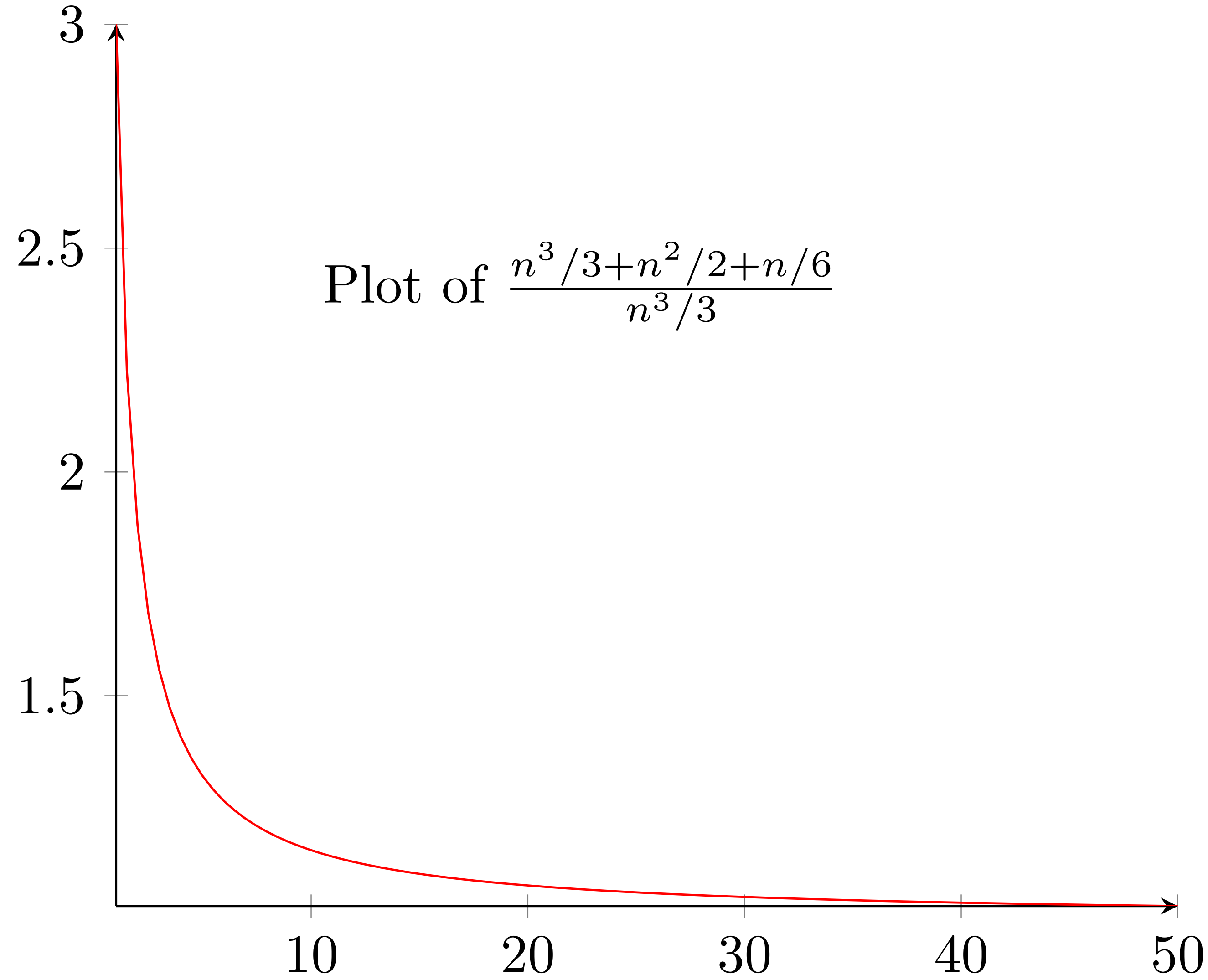
$$\sum_{i=0}^{n} i^2 = \frac{n}{6} + \frac{n^2}{2} + \frac{n^3}{3} > \frac{n^3}{3} = \int_0^n x^2\,dx$$

Approximation

For later on, we will just use the approximation

$$\sum_{i=0}^{n} i^2 \approx \frac{n^3}{3}$$

As n goes to infinity, the ratio of these quantities goes to 1.



Gaussian Elimination: Matrix View

Reading: Strang 2.2,2.3

Learning objective: Be able to work with the augmented matrix to solve systems of linear equations.

Matrix Form of Linear Equations

Consider the following system of linear equations:

$$\begin{aligned} x_1 - 3x_2 + 2x_3 &= 1 \\ 2x_1 - 7x_2 + 5x_3 &= 1 \\ -x_1 + 5x_2 + x_3 &= 1 \end{aligned}$$

We can write this system as one vector equation:

$$x_1 \cdot \begin{bmatrix} 1 \\ 2 \\ -1 \end{bmatrix} + x_2 \cdot \begin{bmatrix} -3 \\ -7 \\ 5 \end{bmatrix} + x_3 \cdot \begin{bmatrix} 2 \\ 5 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Matrix Form of Linear Equations

$$x_1 \cdot \begin{bmatrix} 1 \\ 2 \\ -1 \end{bmatrix} + x_2 \cdot \begin{bmatrix} -3 \\ -7 \\ 5 \end{bmatrix} + x_3 \cdot \begin{bmatrix} 2 \\ 5 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Placing the column vectors into a matrix, we can write this a a matrix-vector product:

$$\begin{bmatrix} 1 & -3 & 2 \\ 2 & -7 & 5 \\ -1 & 5 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Matrix Form of Linear Equations

$$\begin{bmatrix} 1 & -3 & 2 \\ 2 & -7 & 5 \\ -1 & 5 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Call the matrix on the left A, and the vector on the right $\vec{b}$.

We can very compactly write this equation as

$$A\vec{x} = \vec{b}$$

where $\vec{x} = (x_1, x_2, x_3)$.

Matrix Form of Linear Equations

$$\begin{bmatrix} 1 & -3 & 2 \\ 2 & -7 & 5 \\ -1 & 5 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

We can very compactly write this equation as

$$A\vec{x} = \vec{b}$$

where $\vec{x} = (x_1, x_2, x_3)$.

This is how I will typically write systems of linear equations on problem sets, quizzes, etc.

Augmented Matrix

$$\begin{bmatrix} 1 & -3 & 2 \\ 2 & -7 & 5 \\ -1 & 5 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

We can further develop some notation to facilitate doing Gaussian elimination.

The vector $\vec{x} = (x_1, x_2, x_3)$ does not contain essential data of the problem.

It is more of a place holder: the first column of A corresponds to x_1, the second to x_2, etc.

Augmented Matrix

$$\begin{bmatrix} 1 & -3 & 2 \\ 2 & -7 & 5 \\ -1 & 5 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

The augmented matrix contains just the essential data of the problem .

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

coefficient matrix A

right hand side $\vec{b}$

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

coefficient matrix A right hand side $\vec{b}$

The coefficients of $\vec{b}$ are written in **bold** to distinguish them from the rest of the matrix (Strang's notation).

When writing by hand, do it like this:

$$
\left[\begin{array}{ccc|c}
1 & -3 & 2 & 1 \\
2 & -7 & 5 & 1 \\
-1 & 5 & 1 & 1
\end{array}\right]
$$

Question

What system of linear equations corresponds to this augmented matrix?

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

Gaussian Elimination

We can do Gaussian elimination directly on the augmented matrix.

$$\begin{aligned} x_1 - 3x_2 + 2x_3 &= 1 \\ 2x_1 - 7x_2 + 5x_3 &= 1 \\ -x_1 + 5x_2 + x_3 &= 1 \end{aligned} \qquad \begin{bmatrix} 1 & -3 & 2 & \mathbf{1} \\ 2 & -7 & 5 & \mathbf{1} \\ -1 & 5 & 1 & \mathbf{1} \end{bmatrix}$$

$$\downarrow \qquad R_2' = R_2 - 2R_1 \qquad \downarrow$$

$$\begin{aligned} x_1 - 3x_2 + 2x_3 &= 1 \\ -x_2 + x_3 &= -1 \\ -x_1 + 5x_2 + x_3 &= 1 \end{aligned} \qquad \begin{bmatrix} 1 & -3 & 2 & \mathbf{1} \\ 0 & -1 & 1 & -\mathbf{1} \\ -1 & 5 & 1 & \mathbf{1} \end{bmatrix}$$

We can do Gaussian elimination directly on the augmented matrix.

$$\begin{aligned} x_1 - 3x_2 + 2x_3 &= 1 \\ -x_2 + x_3 &= -1 \\ -x_1 + 5x_2 + x_3 &= 1 \end{aligned}$$

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

$$\Big\downarrow \qquad R_3' = R_3 + R_1 \qquad \Big\downarrow$$

$$\begin{aligned} x_1 - 3x_2 + 2x_3 &= 1 \\ -x_2 + x_3 &= -1 \\ 2x_2 + 3x_3 &= 2 \end{aligned}$$

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

$$
\begin{aligned}
x_1 - 3x_2 + 2x_3 &= 1 \\
-x_2 + x_3 &= -1 \\
2x_2 + 3x_3 &= 2
\end{aligned}
\qquad
\begin{bmatrix} 1 & -3 & 2 & \mathbf{1} \\ 0 & -1 & 1 & \mathbf{-1} \\ 0 & 2 & 3 & \mathbf{2} \end{bmatrix}
$$

$$\downarrow \qquad R_3' = R_3 + 2R_2 \qquad \downarrow$$

$$
\begin{aligned}
x_1 - 3x_2 + 2x_3 &= 1 \\
-x_2 + x_3 &= -1 \\
5x_3 &= 0
\end{aligned}
\qquad
\begin{bmatrix} 1 & -3 & 2 & \mathbf{1} \\ 0 & -1 & 1 & \mathbf{-1} \\ 0 & 0 & 5 & \mathbf{0} \end{bmatrix}
$$

We have reached upper triangular form.

How many solutions are there?

Back Substitution

$$\begin{aligned} x_1 - 3x_2 + 2x_3 &= 1 \\ -x_2 + x_3 &= -1 \\ 5x_3 &= 0 \end{aligned}$$

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

To do back substitution with the augmented matrix, recall that the third column corresponds to the third variable, etc.

$$5x_3 = 0 \implies x_3 = 0$$

$$-x_2 + 0 = -1 \implies x_2 = 1$$

$$x_1 - 3 + 2 \cdot 0 = 1 \implies x_1 = 4$$

Check with Python

$$\underbrace{\begin{bmatrix} 1 & -3 & 2 \\ 2 & -7 & 5 \\ -1 & 5 & 1 \end{bmatrix}}_{A} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \underbrace{\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}}_{\vec{b}}$$

First we import the package numpy.

$$>>> \text{import numpy as np}$$

Then we enter the matrix A and the vector $\vec{b}$.

$$>>> A = \text{np.array}([[1, -3, 2], [2, -7, 5], [-1, 5, 1]])$$

$$>>> b = \text{np.array}([[1], [1], [1]])$$

Check with Python

$$>>> A = \text{np.array}([[1, -3, 2], [2, -7, 5], [-1, 5, 1]])$$

$$>>> b = \text{np.array}([[1], [1], [1]])$$

Each row of the matrix is put between brackets. Then all the rows are again put between brackets to tell numpy it is a 2D array.

$$>>> x = \text{np.linalg.solve}(A, b)$$

$$>>> x$$

$$\begin{aligned}&\text{array}([[4.],\\ &\qquad [1.],\\ &\qquad [-0.]])\end{aligned}$$

Note: linalg.solve only works for square systems.

Gaussian Elimination: The Algorithm

Reading: Strang 2.6 section “The Cost of Elimination”

Learning objective: Appreciate the efficiency of Gaussian elimination and back substitution.

How long does Gaussian elimination take?

Numpy's linalg.solve essentially solves linear systems using Gaussian elimination.

In practice, we want to solve linear systems with thousands of equations and thousands of unknowns?

Can this be done?

How fast is Gaussian elimination?

Experiment

Let's set up a system of 1000 equations in 1000 unknowns. How long will this take to solve on my laptop?

Theoretical Justification

Let's try to estimate the speed of Gaussian elimination from a theoretical perspective.

This is necessarily an inexact science. There are many things about the implementation that affect the speed of an algorithm.

As a basic estimate, we will count the number of arithmetic operations: the number of additions and multiplications.

The first thing we need to do is write out the algorithm more formally.

Step One

We describe the algorithm in terms of the augmented matrix.

Step 1: Locate the leftmost coefficient column that is not all zero.

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

this column

this column

Step One

We describe the algorithm in terms of the augmented matrix.

Step 1: Locate the leftmost coefficient column that is not all zero.

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

If there are no nonzero coefficient columns then stop! The system is already upper triangular.

Step One

We describe the algorithm in terms of the augmented matrix.

Step 1: Locate the leftmost coefficient column that is not all zero.

How many arithmetic operations does this step take?

Step Two

Step 2: Swap the top row with another row, if necessary, to bring a nonzero entry to the top of the column found in step 1.

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

no row swap needed

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

$$\Big\downarrow \; R_1 \leftrightarrow R_3$$

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

How many arithmetic operations are needed for step two?

Step Three

Step 3: For each row below the top row, add a suitable multiple of the top row so that the entry below the leading entry of the top row becomes zero.

$$\begin{bmatrix} 1 & -3 & 2 & \mathbf{1} \\ 2 & -1 & 2 & \mathbf{1} \\ 3 & 2 & 6 & \mathbf{1} \end{bmatrix} \xrightarrow[\substack{R_2' = R_2 - 2R_1 \\ R_3' = R_3 - 3R_1}]{} \begin{bmatrix} 1 & -3 & 2 & \mathbf{1} \\ 0 & 5 & -2 & -\mathbf{1} \\ 0 & 11 & 0 & -\mathbf{2} \end{bmatrix}$$

Step Four

Step 4: Cover the top row and column just finished and go back to step 1 with the submatrix that remains.

$$\begin{bmatrix} 1 & -3 & 2 & \mathbf{1} \\ 0 & 5 & -2 & -\mathbf{1} \\ 0 & 11 & 0 & -\mathbf{2} \end{bmatrix}$$

Questions

Is the algorithm guaranteed to stop?

If the original system has n equations and n unknowns, how many times might we have to repeat steps 1-4 before stopping?

Answers

$$\begin{bmatrix} 1 & -3 & 2 & 1 \\ 0 & \boxed{5} & -2 & -\mathbf{1} \\ 0 & 11 & 0 & -\mathbf{2} \end{bmatrix}$$

Every time through steps 1-4 we process (at least) one column.

After a column is processed it will have zeros in the appropriate locations—we don't need to return to it.

After (at most) n iterations, all columns will be processed.

Number of Arithmetic Operations

Step 1: Locate the leftmost coefficient column that is not all zero.

None.

Step 2: Swap the top row with another row, if necessary, to bring a nonzero entry to the top of the column found in step 1.

None.

Step 4: Cover the top row and column just finished and go back to step 1 with the submatrix that remains.

None.

Step Three

We only do arithmetic operations in Step Three.

$$\begin{bmatrix} 1 & -3 & 2 & \mathbf{1} \\ 2 & -1 & 2 & \mathbf{1} \\ 3 & 2 & 6 & \mathbf{1} \end{bmatrix}$$

For this example, let's count how many operations go into forming the new second row.

$$R_2' = R_2 - 2R_1$$

We have to form

$$2R_1 = 2 \cdot (1, -3, 2, 1) = (2, -6, 4, 2)$$

This takes 4 multiplications.

Number of Operations

$$\begin{bmatrix} 1 & -3 & 2 & \mathbf{1} \\ 2 & -1 & 2 & \mathbf{1} \\ 3 & 2 & 6 & \mathbf{1} \end{bmatrix}$$

$$R_2' = R_2 - 2R_1$$

We have to form

$$2R_1 = 2 \cdot (1, -3, 2, 1) = (2, -6, 4, 2)$$

This takes 4 multiplications.

Then we have to add

$$(2, -1, 2, 1) - (2, -6, 4, 2) = (0, 5, -2, -1)$$

This takes 4 additions.

Number of Operations

$$\begin{bmatrix} 1 & -3 & 2 & \mathbf{1} \\ 2 & -1 & 2 & \mathbf{1} \\ 3 & 2 & 6 & \mathbf{1} \end{bmatrix} \xrightarrow{} \begin{bmatrix} 1 & -3 & 2 & \mathbf{1} \\ 0 & 5 & -2 & -\mathbf{1} \\ 0 & 11 & 0 & -\mathbf{2} \end{bmatrix}$$

$$R_2' = R_2 - 2R_1$$
$$R_3' = R_3 - 3R_1$$

Similarly forming the new third row

$$R_3' = R_3 - 3R_1$$

takes 4 multiplications and 4 additions.

In total this step takes

$2 \cdot 4$ multiplications and $2 \cdot 4$ additions

Number of Operations

Now let's think about the general Step 3 case. Suppose the current submatrix is m-by-$m+1$.

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} & \mathbf{b_1} \\ a_{21} & a_{22} & \cdots & a_{2m} & \mathbf{b_2} \\ \vdots & & \cdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mm} & \mathbf{b_m} \end{bmatrix}$$

To do the operation

$$R_2' = R_2 + c \cdot R_1$$

takes $m+1$ multiplications and $m+1$ additions.

Now let's think about the general Step 3 case. Suppose the current submatrix is m-by-$m+1$.

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} & \mathbf{b_1} \\ a_{21} & a_{22} & \cdots & a_{2m} & \mathbf{b_2} \\ \vdots & & \cdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mm} & \mathbf{b_m} \end{bmatrix}$$

To do the operation

$$R_2' = R_2 + c \cdot R_1$$

takes $m+1$ multiplications and $m+1$ additions.

We have to do this for each row $R_2, \ldots, R_m$, so a total of $m-1$ many rows.

Now let's think about the general Step 3 case. Suppose the current submatrix is m-by-$m+1$.

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} & \mathbf{b_1} \\ a_{21} & a_{22} & \cdots & a_{2m} & \mathbf{b_2} \\ \vdots & & \cdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mm} & \mathbf{b_m} \end{bmatrix}$$

In total we have

$$(m-1)\cdot(m+1) = m^2 - 1$$

many multiplications and the same number of additions.

As we just want a rough estimate, let's approximate this by m^2.

Total Count

Say the initial system has n equations and n unknowns.

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & \mathbf{b_1} \\ a_{21} & a_{22} & \cdots & a_{2n} & \mathbf{b_2} \\ \vdots & & \cdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} & \mathbf{b_n} \end{bmatrix}$$

Processing the first column takes approximately

n^2 multiplications and n^2 additions.

Total Count

Say the initial system has n equations and n unknowns.

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & \mathbf{b_1} \\ a_{21} & a_{22} & \cdots & a_{2n} & \mathbf{b_2} \\ \vdots & & \cdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} & \mathbf{b_n} \end{bmatrix}$$

After covering up the first row and column, the remaining submatrix is $(n-1)$-by-n.

Processing the second column takes approximately

$(n-1)^2$ multiplications and $(n-1)^2$ additions.

Total Count

Say the initial system has n equations and n unknowns.

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & \mathbf{b_1} \\ a_{21} & a_{22} & \cdots & a_{2n} & \mathbf{b_2} \\ \vdots & & \cdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} & \mathbf{b_n} \end{bmatrix}$$

After covering up the first two rows and columns, the remaining submatrix is $(n-2)$-by-$(n-1)$.

Processing the third column takes approximately

$(n-2)^2$ multiplications and $(n-2)^2$ additions.

Total Count

Say the initial system has n equations and n unknowns.

Proceeding in the same fashion, we see that the total number of multiplications will be approximately

$$n^2 + (n-1)^2 + (n-2)^2 + \cdots + 2^2 + 1^2 \approx \frac{1}{3}n^3$$

and the same for the number of additions.

Gaussian elimination requires about $\frac{1}{3}n^3$ multiplications and $\frac{1}{3}n^3$ additions to arrive at an upper triangular system.

Back Substitution

We are not quite done yet as we still have to solve the upper triangular system by back substitution.

How many arithmetic operations does this take?

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

Solving for x_3 takes one multiplication (division).

Back Substitution

How many arithmetic operations does this take?

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

Solving for x_2 takes two multiplications and one addition.

$$5x_2 - 2 \cdot (-1) = -1$$

$$x_2 = -\frac{3}{5}$$

Back Substitution

How many arithmetic operations does this take?

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

Solving for x_1 takes three multiplications and two additions.

$$x_1 - 3 \cdot \left(-\frac{3}{5}\right) + 2 \cdot (-1) = 1$$

Back Substitution

In general we see that back substitution on a triangular n-by-n system takes about

$$1 + 2 + 3 + \cdots + (n-1) + n = \frac{n(n+1)}{2}$$

many multiplications, and about the same number of additions.

Back substitution takes about $\frac{1}{2}n^2$ many multiplications and $\frac{1}{2}n^2$ many additions.

Summary

In total, solving a system of n equations in n unknowns takes about

$$\frac{2}{3}n^3 + n^2 \approx \frac{2}{3}n^3$$

many arithmetic operations.

Back substitution is much faster than Gaussian elimination.

Check

For a system of $1000 = 10^3$ equations and unknowns, our estimate is $\frac{2}{3}10^9$ many arithmetical operations.

The processor on my laptop is 2.8 GHz. Let's say roughly $3 \cdot 10^9$ many operations per second.

This gives an estimate of about

$$\frac{\frac{2}{3}10^9}{3 \cdot 10^9} = \frac{2}{9} = .22\bar{2} \quad \text{seconds}$$

to solve the system.