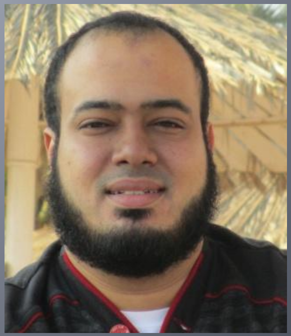
Competitive Programming

From Problem 2 Solution in O(1)

Linear Algebra

Gaussian Elimination - 1

Mostafa Saad Ibrahim
PhD Student @ Simon Fraser University



Systems of linear equations

- System = Consider ALL equations (NOT individually)...each equation is **constraint**
- Linear Equ = Each term => Coefficient * var
  - E.g. 2x, -y NOT x^2 or log(x) or sin(x)...etc
- Solution Variables can be **real** values

$$\begin{aligned} 3x + 2y - z &= 1 \\ 2x - 2y + 4z &= -2 \\ -x + \tfrac{1}{2}y - z &= 0 \end{aligned} \qquad\qquad \begin{aligned} x &= 1 \\ y &= -2 \\ z &= -2 \end{aligned}$$

System Independence

- System is independent if none of the equations can be derived algebraically from the others
  - E.g. by scaling one of them
  - E.g. by adding some of them
- $3x + 2y = 6 \quad \text{and} \quad 6x + 4y = 12$
  - 2nd equation is 2 * first equation
- $$\begin{aligned} x - 2y &= -1 \\ 3x + 5y &= \quad 8 \\ 4x + 3y &= \quad 7 \end{aligned}$$
  - Third equation is the sum of the other two

Free Variables

- Some variables are free to take any value
  - E.g. variable has coefficient zero in all equations
  - E.g. $x = -7z - 1 \quad \text{and} \quad y = 3z + 2.$
  - Put value for z, then x and y are computed
- These variables cause system to have **infinite** solutions..because equations don't force strict constraints on them

Systems of linear equations

- 1 solution
  - 1X + 2Y = 6 , 1X + -4Y = -3
  - -9X + -9Y = -9 , -7X + -8Y = -8
- Infinite solutions
  - 0X + 0Y = 0 , 0X + 0Y = 0
  - 2X + 3Y = 9 , -2X + -3Y = -9
  - 1X + 1Y = 1 , 0X + 0Y = 0
- No solutions
  - 0X + 0Y = 0 , 0X + 0Y = 4
  - -6X + 67 = -7 , -1X + 1Y = -6
  - 1X + 3Y = 1 , 2X + 6Y = -1

Solving 2 equations

```
pair<int, int> prepareFraction(int n, int d)
{
    int div = gcd(n, d);
    n /= div, d /= div;
    if(d < 0)   n *= -1,  d*= -1;
    return make_pair(n, d);
}

pair<string, vector<int> > solve2Equations( int ax, int ay, int az,
                                            int bx, int by, int bz)
{
    if( (!ax && !ay && az) || (!bx && !by && bz) )
        return make_pair("NO_SOLUTIONS", vector<int>(4) );

    if( (!ax && !ay && !az) || (!bx && !by && !bz) )
        return make_pair("INFINITE_SOLUTIONS", vector<int>(4) );

    if( ax*by == ay*bx && ax*bz==az*bx && ay*bz==by*az )
        return make_pair("INFINITE_SOLUTIONS", vector<int>(4) );

    if( ax*by == ay*bx )
        return make_pair("NO_SOLUTIONS", vector<int>(4) );

    pair<int, int> X = prepareFraction(by*az-ay*bz, by*ax-bx*ay);
    pair<int, int> Y = prepareFraction(bx*az-ax*bz, bx*ay-by*ax);

    int sol[] = {X.first, X.second, Y.first, Y.second };
    return make_pair("SOLVED", vector<int>(sol, sol+4) );
}
```

System Representation

- One way to do that: augmented **matrix**:
  - Every row is an equation including its right side
  - Left part is equation and right value = right hand side

$$
\begin{aligned}
x + 3y - 2z &= 5 \\
3x + 5y + 6z &= 7 \\
2x + 4y + 3z &= 8
\end{aligned}
$$

$$
\left[\begin{array}{rrr|r}
1 & 3 & -2 & 5 \\
3 & 5 & 6 & 7 \\
2 & 4 & 3 & 8
\end{array}\right].
$$

Elementary row operations

- Three operations don’t change system constraints
- **Swap** the positions of two rows.
  - Swap (Eq1, Eq2)
- **Multiply** a row by a nonzero scalar.
  - Eq2 *= 4
- **Add** to one row a scalar multiple of **another**.
  - E.g. Eq2 -= 3 Eq1

Unknown Variables Elimination

- Can we use first equation to **eliminate x** in the other 2 equations?
- Eq2 += -2 Eq1
- Eq3 += 1 Eq1

$$
\begin{array}{rcrcrcr}
2x & + & 4y & - & 2z & = & 2 \\
4x & + & 9y & - & 3z & = & 8 \\
-2x & - & 3y & + & 7z & = & 10
\end{array}
$$

$$
\begin{array}{rcrcrcr}
2x & + & 4y & - & 2z & = & 2 \\
 & & y & + & z & = & 4 \\
 & & y & + & 5z & = & 12
\end{array}
$$

Src: Todo

Gauss-Jordan elimination

- Algorithm eliminates unknowns from the equations using the 3 operations
- Iterate over every row:
- For ith row, make matrix diagonal element 1
- **Eliminate** this variable from other equations
- By end of that, right matrix will turn to an **identity matrix**, and left one is the answer
  - The popular variation converts system to triangular one then do back substitution

Gauss-Jordan elimination

$$
\begin{aligned}
2x + y - z &= 8 \\
-3x - y + 2z &= -11 \\
-2x + y + 2z &= -3
\end{aligned}
\quad \Longrightarrow \quad
\left[\begin{array}{ccc|c}
2 & 1 & -1 & 8 \\
-3 & -1 & 2 & -11 \\
-2 & 1 & 2 & -3
\end{array}\right]
$$

3 major steps

$$
\Downarrow
$$

$$
\left[\begin{array}{ccc|c}
1 & 0 & 0 & 2 \\
0 & 1 & 0 & 3 \\
0 & 0 & 1 & -1
\end{array}\right]
$$

Gauss-Jordan elimination

2	1	-1		8
-3	-1	2		-11
-2	1	2		-3

Convert M(1, 1) = 2 to 1 => Eq1 / 2

1	1/2	-1/2		4
-3	-1	2		-11
-2	1	2		-3

Convert M(1, 2) = -3 to 0
Eq2 -= -3 Eq1

1	1/2	-1/2		4
0	1/2	1/2		1
-2	1	2		-3

Convert M(1, 3) = -2 to 0
Eq3 -= -2 Eq1

1	1/2	-1/2		4
0	1/2	1/2		1
0	2	1		5

Gauss-Jordan elimination

1	1/2	-1/2		4
0	**1/2**	1/2		1
0	2	1		5

Convert M(2, 2) = 1/2 to 1 => Eq2 / (1 / 2)

1	**1/2**	-1/2		4
0	1	1		2
0	2	1		5

Convert M(1, 2) = 1/2 to 0
Eq1 -= 1/2 Eq2

1	0	-1		3
0	1	1		2
0	**2**	1		5

Convert M(3, 2) = 2 to 0
Eq3 -= 2 Eq2

1	0	-1		3
0	1	1		2
0	0	-1		1

Gauss-Jordan elimination

1	0	-1		3
0	1	1		2
0	0	**-1**		1

Convert M(3, 3) = -1 to 1 => Eq3 / -1

1	0	**-1**		3
0	1	1		2
0	0	1		-1

Convert M(1, 3) = -1 to 0
Eq1 -= -1 Eq3

1	0	0		2
0	1	**1**		2
0	0	1		-1

Convert M(2, 3) = 1 to 0
Eq2 -= 1 Eq3

1	0	0		2
0	1	0		3
0	0	1		-1

Gauss-Jordan elimination

$$\left[\begin{array}{cccc|c} 3 & 3 & 12 & 21 & 18 \\ 2 & 4 & 10 & 14 & 16 \\ -2 & 0 & -8 & -18 & -10 \\ 3 & 0 & 11 & 28 & 11 \end{array}\right] \to \left[\begin{array}{cccc|c} 1 & 1 & 4 & 7 & 6 \\ 2 & 4 & 10 & 14 & 16 \\ -2 & 0 & -8 & -18 & -10 \\ 3 & 0 & 11 & 28 & 11 \end{array}\right]$$

$$\to \left[\begin{array}{cccc|c} 1 & 1 & 4 & 7 & 6 \\ 0 & 2 & 2 & 0 & 4 \\ 0 & 2 & 0 & -4 & 2 \\ 0 & -3 & -1 & 7 & -7 \end{array}\right] \to \left[\begin{array}{cccc|c} 1 & 1 & 4 & 7 & 6 \\ 0 & 1 & 1 & 0 & 2 \\ 0 & 2 & 0 & -4 & 2 \\ 0 & -3 & -1 & 7 & -7 \end{array}\right]$$

$$\to \left[\begin{array}{cccc|c} 1 & 0 & 3 & 7 & 4 \\ 0 & 1 & 1 & 0 & 2 \\ 0 & 0 & -2 & -4 & -2 \\ 0 & 0 & 2 & 7 & -1 \end{array}\right] \to \left[\begin{array}{cccc|c} 1 & 0 & 3 & 7 & 4 \\ 0 & 1 & 1 & 0 & 2 \\ 0 & 0 & 1 & 2 & 1 \\ 0 & 0 & 2 & 7 & -1 \end{array}\right]$$

$$\to \left[\begin{array}{cccc|c} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & -2 & 1 \\ 0 & 0 & 1 & 2 & 1 \\ 0 & 0 & 0 & 3 & -3 \end{array}\right] \to \left[\begin{array}{cccc|c} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & -2 & 1 \\ 0 & 0 & 1 & 2 & 1 \\ 0 & 0 & 0 & 1 & -1 \end{array}\right]$$

$$\to \left[\begin{array}{cccc|c} 1 & 0 & 0 & 0 & 2 \\ 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & 3 \\ 0 & 0 & 0 & 1 & -1 \end{array}\right]$$

Src: http://linearalgebra4fun.blogspot.ca/2015_06_01_archive.html

Gauss-Jordan elimination

- In the ith step, we may notice all the column of ith variable are zeros!
  - So we can’t make diagonal 1, even with swap operation
  - Then this variable is free, and system has infinite solution
- We may notice equation is invalid, so no solution (equation e.g. $0x + 0y = 5$)
- Generally, we can have Equations != Unknowns..but we can solve it

Gauss-Jordan elimination: Pivot

- **Partial pivoting** = more stable solution
  - What if current equation has 0 value in diagonal element?
  - What if current equation is small (e.g. 0.00001)
- Swap current equation with another one such that |diagonal element| is maximum
- E.g. current diagonal value 3, but other row is -7...so swap the 2 rows
- However, for some matrices still may give inaccurate computations..out of scope

Gauss-Jordan elimination

2	1	-1		8
-3	-1	2		-11
-2	1	2		-3

What max |M(?, 1)|? Eq2 has 3
Swap Eq2 with Eq1

-3	-1	2		-11
2	1	-1		8
-2	1	2		-3

Now do normal processing

Convert M(1, 1) = -3 to 1 => Eq1 / -3

1	1/3	-2/3		11/3
2	1	-1		8
-2	1	2		-3

تم بحمد الله

علمكم الله ما ينفعكم

ونفعكم بما تعلمتم

وزادكم علماً