Problem 5.2.26

ee25btech11023-Venkata Sai

September 14, 2025

1 Problem

2 Solution
- Matrix Equation
- Conclusion
- Plot

3 C Code

4 Python Code

Problem

Slope of a line passing through **P** $(2, 3)$ and intersecting the line $x + y = 7$ at a distance of 4 units from **P**, is

Matrix Equation

Given

$$\frac{x}{a} - \frac{y}{b} = 0 \implies bx = ay \tag{2.1}$$

$$bx - ay = 0 \tag{2.2}$$

$$ax + by = a^2 + b^2 \tag{2.3}$$

As a matrix equation

$$\left(\begin{array}{cc|c} b & -a & 0 \\ a & b & a^2 + b^2 \end{array}\right) \xrightarrow{R_2 \to bR_2 - aR_1} \left(\begin{array}{cc|c} b & -a & 0 \\ 0 & b^2 + a^2 & \left(a^2 + b^2\right) b \end{array}\right) \tag{2.4}$$

$$\left(b^2 + a^2\right) y = \left(a^2 + b^2\right) b \tag{2.5}$$

$$\implies y = b \tag{2.6}$$

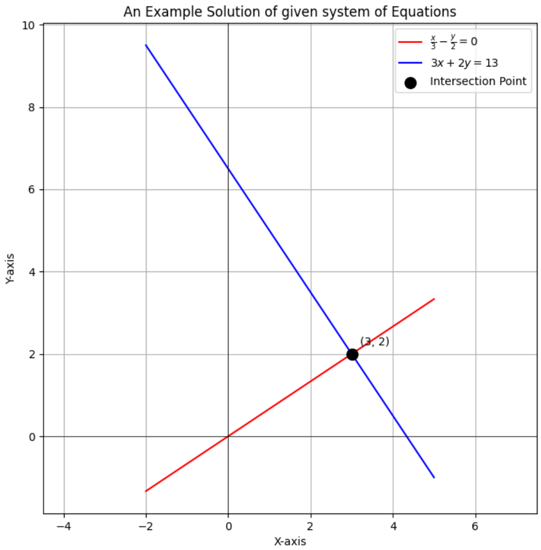
Conclusion

Substituting in equation (3)

$$bx = ab \implies x = a \tag{2.7}$$

Hence $x = a, y = b$ is the solution for given system of linear equations

Plot



An Example Solution of given system of Equations
$\frac{x}{3} - \frac{y}{2} = 0$
$3x + 2y = 13$
Intersection Point
(3, 2)
Y-axis
X-axis
10
8
6
4
2
0
−4
−2
0
2
4
6

C Code

```c
void get_system_coeffs(double* out_coeffs) {

    double a = 3.0;
    double b = 2.0;

    out_coeffs[0] = b;
    out_coeffs[1] = -a;
    out_coeffs[2] = a;
    out_coeffs[3] = b;

    out_coeffs[4] = 0;
    out_coeffs[5] = a*a + b*b;

    out_coeffs[6] = a;
    out_coeffs[7] = b;
}
```

Python Code for Calling

```python
import ctypes
import numpy as np
from sympy import Matrix

def solve_and_prepare_data():

    lib = ctypes.CDLL('./code.so')
    double_array_8 = ctypes.c_double * 8
    lib.get_system_coeffs.argtypes = [ctypes.POINTER(ctypes.
        c_double)]
    out_data_c = double_array_8()
    lib.get_system_coeffs(out_data_c)
    raw_data = list(out_data_c)

    # Unpack the raw data
    m_coeffs = raw_data[0:4]
    c_coeffs = raw_data[4:6]
    a, b = raw_data[6], raw_data[7]
```

Python Code for Solving

```python
    aug_M = Matrix([
        [m_coeffs[0], m_coeffs[1], c_coeffs[0]],
        [m_coeffs[2], m_coeffs[3], c_coeffs[1]]
    ])
    rref_matrix, _ = aug_M.rref()
    solution_P = np.array(rref_matrix[:, -1]).astype(np.float64).
        flatten()
    x_start, x_end = -2.0, a + 2.0
    y1_start, y1_end = (b/a) * x_start, (b/a) * x_end
    y2_start, y2_end = (-a/b) * x_start + (a**2 + b**2)/b, (-a/b)
         * x_end + (a**2 + b**2)/b

    return {
        "line1_x": [x_start, x_end], "line1_y": [y1_start, y1_end
            ],
        "line2_x": [x_start, x_end], "line2_y": [y2_start, y2_end
            ],
        "solution_point": solution_P,
        "a": a, "b": b
    }
```

Python Code for Plotting

```python
#Code by GVV Sharma
#September 12, 2023
#Revised July 21, 2024
#released under GNU GPL
import sys #for path to external scripts
sys.path.insert(0, '/workspaces/urban-potato/matgeo/codes/
    CoordGeo/')
import matplotlib.pyplot as plt
import numpy as np
from call import solve_and_prepare_data

plot_data = solve_and_prepare_data()

line1_x = plot_data["line1_x"]
line1_y = plot_data["line1_y"]
line2_x = plot_data["line2_x"]
line2_y = plot_data["line2_y"]
x_sol, y_sol = plot_data["solution_point"]
a, b = plot_data["a"], plot_data["b"]
```

Python Code for Plotting

```python
print(f"Plotting solution for a={a:.0f}, b={b:.0f}: Point is ({
    x_sol:.0f}, {y_sol:.0f})")

fig, ax = plt.subplots(figsize=(8, 8))

# Use the unpacked lists directly
ax.plot(line1_x, line1_y, 'r-', label=f'$\\frac{{x}}{{{a:.0f}}} -
     \\frac{{y}}{{{b:.0f}}} = 0$')
ax.plot(line2_x, line2_y, 'b-', label=f'${a:.0f}x + {b:.0f}y = {a
    **2 + b**2:.0f}$')
ax.scatter(x_sol, y_sol, color='black', s=100, zorder=5, label='
    Intersection Point')
ax.text(x_sol + 0.2, y_sol + 0.2, f'({x_sol:.0f}, {y_sol:.0f})')

ax.set_title('An Example Solution of given system of Equations')
ax.set_xlabel('X-axis')
ax.set_ylabel('Y-axis')
```

Python Code for Plotting

```python
ax.grid(True)
ax.axhline(0, color='k', linewidth=0.5)
ax.axvline(0, color='k', linewidth=0.5)
ax.axis('equal')
ax.legend()
plt.show()
plt.savefig('fig1.png')
```