Matgeo Presentation - Problem 1.6.11

EE25BTECH11061 - Vankudoth Sainadh

August 29, 2025

Problem Statement

If the points $A(1, 2)$, $O(0, 0)$ and $C(a, b)$ are collinear, find the relation between a and b.

Method

Condition for Collinearity:
Three points A, O, C are collinear iff the collinearity matrix

$$M = \begin{pmatrix} \mathbf{O} - \mathbf{A} & \mathbf{C} - \mathbf{A} \end{pmatrix}^\top$$

has $rank\left(M\right) = 1$.

Echelon Form and Row Operations

$$M = \begin{pmatrix} \mathbf{O} - \mathbf{A} & \mathbf{C} - \mathbf{A} \end{pmatrix}^T = \begin{pmatrix} -1 & -2 \\ a-1 & b-2 \end{pmatrix},$$

$$\text{Collinearity} \iff \text{rank}(M) = 1.$$

$$R_2 \longrightarrow R_2 - \frac{a-1}{-1} R_1 \quad (\text{note: } -1 \neq 0),$$

$$\begin{pmatrix} -1 & -2 \\ a-1 & b-2 \end{pmatrix} \longrightarrow \begin{pmatrix} -1 & -2 \\ 0 & b-2a \end{pmatrix}.$$

$$\text{rank}(M) = 1 \iff \text{second row is the zero row} \iff b - 2a = 0.$$

$$\boxed{b = 2a}.$$

Final Answer

$$\boxed{b = 2a}$$

C Code: points.c

```c
#include <stdio.h>

// Function to return relation value (0 => COLLINEAR)
int relation(int a, int b) {
    return b - 2*a; // For A(1,2), O(0,0), C(a,b): collinear <=> b - 2a = 0
}

int main(void) {
    // Given points: A(1,2) and O(0,0)
    int x1 = 0, y1 = 0; // O
    int x2 = 1, y2 = 2; // A

    // Step 1: Compute slope of line through O and A
    float m = (float)(y2 - y1) / (x2 - x1);
    printf(Step 1: Compute slope using two points O(0,0) and A(1,2):\n);
    printf( m = (y2 - y1) / (x2 - x1) = (%d - %d) / (%d - %d) = %.2f\n\n,
           y2, y1, x2, x1, m);


    // Step 2: Point-slope form using point O(0,0)
    printf(Step 2: Equation using point-slope form (through O):\n);
    printf( (y - %d) = m (x - %d)\n, y1, x1);
    printf( => y = m x\n\n);

    // Step 3: Substitute m = 2 (from Step 1)
    printf(Step 3: With m = %.0f, the line is: y = 2x\n\n, m);
    // Step 4: Final relation for C(a,b) lying on this line
    printf(Step 4: Substitute C(a,b) into y = 2x => b = 2a\n);
    printf(Final Relation: b - 2a = 0\n\n);

    // (Optional) quick test: uncomment to verify with numbers
    // int a = 3, b = 6;
    // printf(Test with a=%d, b=%d -> residual (b - 2a) = %d\n, a, b, relation(a,b));

    return 0;
}
```

Python: call_c.py

```
import ctypes, argparse

# Load the shared object produced above
lib = ctypes.CDLL(./collinear.so)

# Configure signatures
lib.relation.argtypes = [ctypes.c_int, ctypes.c_int]
lib.relation.restype = ctypes.c_int

lib.collinear_AO_C.argtypes = [ctypes.c_double, ctypes.c_double,
    ctypes.c_double,
                            ctypes.POINTER(ctypes.c_double)]
lib.collinear_AO_C.restype = ctypes.c_int

def main():
    ap = argparse.ArgumentParser(description=Check collinearity
        for A(1,2), O(0,0), C(a,b))
    ap.add_argument(--a, type=float, default=3.0)
    ap.add_argument(--b, type=float, default=6.0)
    ap.add_argument(--tol, type=float, default=1e-9)
    args = ap.parse_args()

    # int API (residual = b - 2a as an int)
    r_int = lib.relation(int(round(args.a)), int(round(args.b)))

    # double API (residual + boolean)
    resid = ctypes.c_double()
    ok = lib.collinear_AO_C(args.a, args.b, args.tol, ctypes.
        byref(resid))

    print(fResidual (b - 2a) via int API: {r_int})
    print(fResidual (b - 2a) via double API: {resid.value:.6e})
    print(Status:, COLLINEAR if ok else NOT collinear)

if __name__ == __main__:
    main()
```

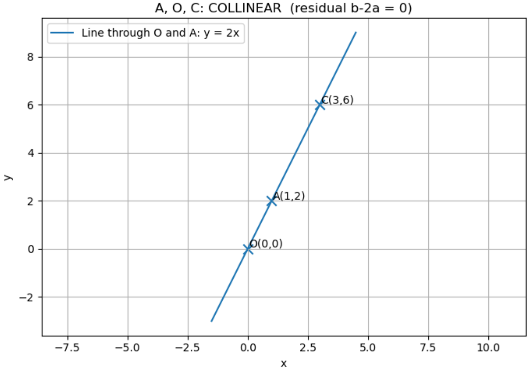
Python: plot.py

```
import argparse, os, ctypes
import numpy as np
import matplotlib
if not os.environ.get(DISPLAY):
    matplotlib.use(Agg)
import matplotlib.pyplot as plt

# load C lib
lib = ctypes.CDLL(./collinear.so)
lib.relation.argtypes = [ctypes.c_int, ctypes.c_int]
lib.relation.restype = ctypes.c_int

def main():
    ap = argparse.ArgumentParser(description=Plot A(1,2), O(0,0),
         C(a,b) and line y=2x)
    ap.add_argument(--a, type=float, default=3.0)
    ap.add_argument(--b, type=float, default=6.0)
    ap.add_argument(--save, type=str, default=collinearity_plot.png)
    ap.add_argument(--no-show, action=store_true)
    args = ap.parse_args()

    A = (1.0, 2.0)
    O = (0.0, 0.0)
    C = (args.a, args.b)

    # residual from shared lib (int API, for display)
    r_int = lib.relation(int(round(args.a)), int(round(args.b)))

   # line y = 2x through O and A
    x_min = min(-1.0, O[0], A[0], C[0]) - 0.5
    x_max = max(4.0, O[0], A[0], C[0]) + 0.5
    xs = np.linspace(x_min, x_max, 400)
    ys = 2 * xs
    plt.figure(figsize=(7,5))
```

```
plt.plot(xs, ys, label=Line through O and A: y = 2x)
plt.scatter([O[0], A[0], C[0]], [O[1], A[1], C[1]], s=80,
    marker=x)


plt.text(O[0]+0.05, O[1]+0.05, O(0,0))
plt.text(A[0]+0.05, A[1]+0.05, A(1,2))
plt.text(C[0]+0.05, C[1]+0.05, fC({C[0]:.3g},{C[1]:.3g}))

status = COLLINEAR if r_int == 0 else NOT collinear
plt.title(fA, O, C: {status} (residual b-2a = {r_int}))
plt.xlabel(x); plt.ylabel(y); plt.grid(True); plt.legend();
    plt.axis(equal); plt.tight_layout()

plt.savefig(args.save, dpi=150)
print(fResidual (b - 2a) = {r_int} -> {status})
print(fPlot saved to: {args.save})
```

```
    if not args.no_show and matplotlib.get_backend().lower() not in {agg}:
        plt.show()
if __name__ == __main__:
    main()
```

Plot by python using shared output from c


A, O, C: COLLINEAR (residual b-2a = 0)
Line through O and A: y = 2x
C(3,6)
A(1,2)
O(0,0)
y
x