2.7.21

Anshu kumar ram-EE25BTECH11009

September 14, 2025

Question

Find the values of k so that the area of the triangle with vertices $A(1, -1),\ B(-4, 2k),\ C(-k, -5)$ is 24 sq. units.

Step 1: Vertices

Point	Coordinates
A	$(1, -1)$
B	$(-4, 2k)$
C	$(-k, -5)$

Table: Vertices of $\triangle ABC$ before substituting k

Step 2: Vectors

$$\mathbf{u} = \mathbf{B} - \mathbf{A} = \begin{pmatrix} -5 \\ 2k+1 \end{pmatrix}, \tag{1}$$

$$\mathbf{v} = \mathbf{C} - \mathbf{A} = \begin{pmatrix} -k-1 \\ -4 \end{pmatrix} \tag{2}$$

$$\Delta = \frac{1}{2}\|\mathbf{u} \times \mathbf{v}\| \tag{3}$$

Step 3: Norm Identity

$$\left\|\mathbf{u} \times \mathbf{v}\right\|^2 = \left\|\mathbf{u}\right\|^2 \left\|\mathbf{v}\right\|^2 - (\mathbf{u}^\top \mathbf{v})^2 \tag{4}$$

$$\implies \left\|\mathbf{u} \times \mathbf{v}\right\| = |2k^2 + 3k + 21| \tag{5}$$

$$\Delta = \frac{1}{2}|2k^2 + 3k + 21| \tag{6}$$

Step 4: Solve for k

$$\frac{1}{2}|2k^2 + 3k + 21| = 24 \tag{7}$$
$$|2k^2 + 3k + 21| = 48 \tag{8}$$

Case 1:

$$2k^2 + 3k - 27 = 0 \implies k = 3,\ -\tfrac{9}{2} \tag{9}$$

Case 2:

$$2k^2 + 3k + 69 = 0 \ \text{(no real roots)} \tag{10}$$

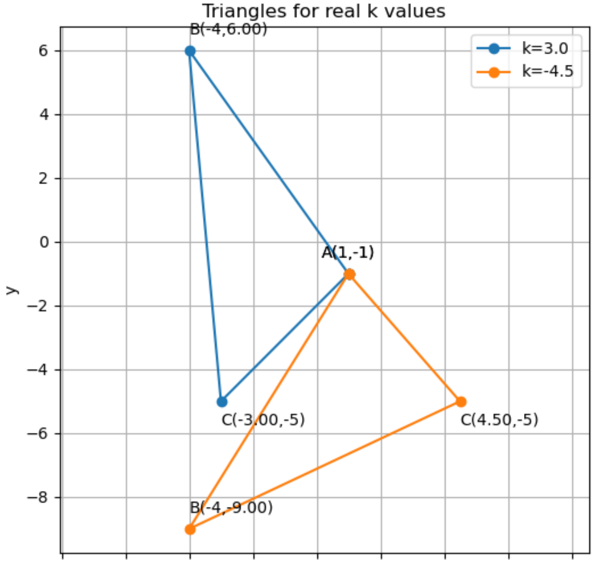
Step 5: Final Answer

$$\therefore k \in \{3,\ -\tfrac{9}{2}\} \tag{11}$$

Point	For $k = 3$	For $k = -\frac{9}{2}$
A	$(1,\ -1)$	$(1,\ -1)$
B	$(-4,\ 6)$	$(-4,\ -9)$
C	$(-3,\ -5)$	$\left(\frac{9}{2},\ -5\right)$

Table: Vertices of $\triangle ABC$ after substituting k values

Graph



Triangles for real k values
k=3.0
k=-4.5
B(-4,6.00)
A(1,-1)
C(-3.00,-5)
C(4.50,-5)
B(-4,-9.00)
y
6
4
2
0
−2
−4
−6
−8

C Code (Part 1)

```c
#include <stdio.h>
#include <math.h>

// Function to compute area of a triangle given coordinates
double triangle_area(double *A, double *B, double *C) {
    // A, B, C are arrays of size 2: [x, y]
    double x1 = A[0], y1 = A[1];
    double x2 = B[0], y2 = B[1];
    double x3 = C[0], y3 = C[1];
```

C Code (Part 2)

```c
    // Determinant method for area
    double det = x1*(y2 - y3) + x2*(y3 - y1) + x3*(y1 - y2);
    return fabs(det) / 2.0;
}

/* Build as shared library:
   gcc -fPIC -shared -o func.so func.c
*/
```

Python + C (Part 1)

```python
import ctypes
import numpy as np
import matplotlib.pyplot as plt

# Load shared library
handc = ctypes.CDLL("./func.so")

# Define argument and return types for the C function
handc.triangle_area.argtypes = [
    ctypes.POINTER(ctypes.c_double), # A
    ctypes.POINTER(ctypes.c_double), # B
    ctypes.POINTER(ctypes.c_double) # C
]
handc.triangle_area.restype = ctypes.c_double
```

Python + C (Part 2)

```
# Convert numpy arrays to C pointers
def np_to_c(arr):
    return arr.ctypes.data_as(ctypes.POINTER(ctypes.c_double))

# Fixed point A
A = np.array([1.0, -1.0], dtype=np.float64)

# k values we found
k_vals = [3.0, -9.0/2.0]
plt.figure(figsize=(6,6))
```

Python + C (Part 3)

```
for k in k_vals:
    B = np.array([-4.0, 2.0*k], dtype=np.float64)
    C = np.array([-k, -5.0], dtype=np.float64)

    # Call the C function for area
    area = handc.triangle_area(np_to_c(A), np_to_c(B), np_to_c(C)
        )
    print(f"k = {k}, area = {area}")

    # Plot triangle
    x_coords = [A[0], B[0], C[0], A[0]]
    y_coords = [A[1], B[1], C[1], A[1]]
    plt.plot(x_coords, y_coords, marker='o', label=f"k={k}")
```

Python + C (Part 4)

```
    # Plot points with labels
    plt.scatter([A[0], B[0], C[0]], [A[1], B[1], C[1]], s=50)
    plt.annotate("A(1,-1)", (A[0], A[1]), textcoords="offset
        points", xytext=(0,10), ha='center')
    plt.annotate(f"B(-4,{2*k:.1f})", (B[0], B[1]), textcoords="
        offset points", xytext=(0,10))
    plt.annotate(f"C({-k:.1f},-5)", (C[0], C[1]), textcoords="
        offset points", xytext=(0,-15))

plt.xlabel("x")
plt.ylabel("y")
plt.title("Triangles (Python + C area function)")
plt.legend()
plt.axis("equal")
plt.grid(True)
plt.savefig("../figs/triangle_area_c.png")
plt.show()
```

Pure Python (Part 1)

```
import math
import sys
sys.path.insert(0, '/home/anshu-ram/matgeo/codes/CoordGeo')
import numpy as np
import numpy.linalg as LA
import matplotlib.pyplot as plt

# Given vertex A
A = np.array([1.0, -1.0]).reshape(-1,1)

# Real k solutions found
k_vals = [3.0, -9.0/2.0]
plt.figure(figsize=(6,6))
```

Pure Python (Part 2)

```
for k in k_vals:
    B = np.array([-4.0, 2.0*k]).reshape(-1,1)
    C = np.array([-k, -5.0]).reshape(-1,1)

    tri = np.hstack((A, B, C, A))
    plt.plot(tri[0,:], tri[1,:], linestyle='-', marker='o', label
        =f'k={k}')

    # area using cross product
    u = (B - A).flatten()
    v = (C - A).flatten()
    cross = abs(u[0]*v[1] - u[1]*v[0])
    area = 0.5*cross
    print(f"k={k} => computed area = {area}")
```

Pure Python (Part 3)

```python
    # annotate vertices
    plt.annotate(f'A(1,-1)', (A[0,0], A[1,0]), textcoords="offset
         points", xytext=(0,10), ha='center')
    plt.annotate(f'B(-4,{2*k:.2f})', (B[0,0], B[1,0]), textcoords
        ="offset points", xytext=(0,10))
    plt.annotate(f'C({-k:.2f},-5)', (C[0,0], C[1,0]), textcoords=
        "offset points", xytext=(0,-15))

plt.xlabel('x')
plt.ylabel('y')
plt.title('Triangles for real k values')
plt.legend()
plt.axis('equal')
plt.grid(True)
plt.savefig("../figs/triangle_area.png")
plt.show()
```