7.4.20

Bhargav - EE25BTECH11013

September 20, 2025

Question

The point diametrically opposite to the point $P\left(1,0\right)$ on the circle

$$x^2 + y^2 + 2x + 2y - 3 = 0 \tag{1}$$

is

Solution

Let the diametrically opposite point be $\mathbf{Q}$.
The equation of the circle is:($\mathbf{V}$ is an identity matrix of order $= 2$)

$$\mathbf{x^T V x} + 2\mathbf{u^T x} + f = 0 \tag{2}$$

$$\mathbf{u} = \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \tag{3}$$

The center of the circle $\mathbf{c}$ is

$$\implies \mathbf{c} = -\mathbf{u} = \begin{pmatrix} -1 \\ -1 \end{pmatrix} \tag{4}$$

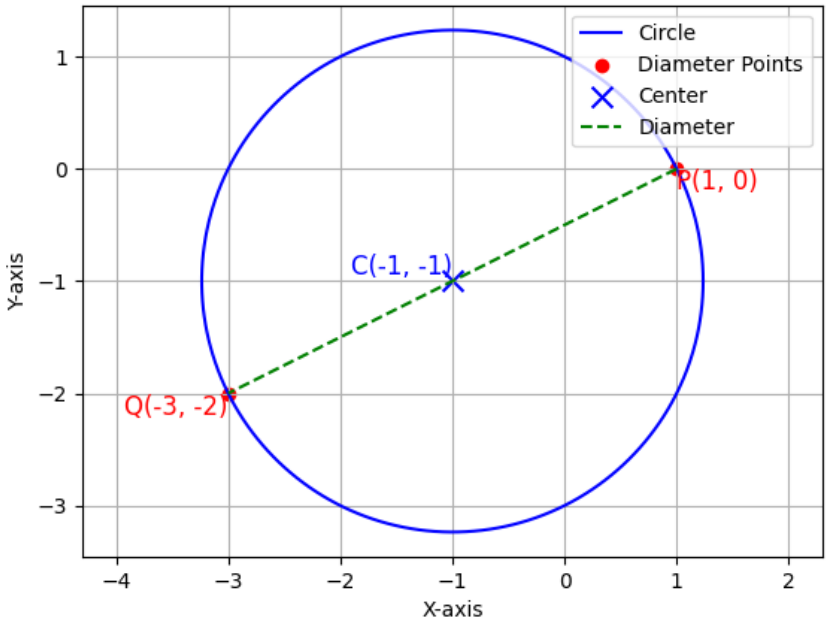
Solution

Using the property of diametrically opposite points:

$$\mathbf{c} = \frac{\mathbf{P} + \mathbf{Q}}{2} \tag{5}$$

$$\mathbf{Q} = 2\mathbf{c} - \mathbf{P} = 2\begin{pmatrix} -1 \\ -1 \end{pmatrix} - \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} -3 \\ -2 \end{pmatrix} \tag{6}$$

Plot


Circle
Diameter Points
Center
Diameter
P(1, 0)
C(-1, -1)
Q(-3, -2)
Y-axis
X-axis
1
0
−1
−2
−3
−4
−3
−2
−1
0
1
2

C Code

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

int xcenter(int c, int p, int m, int n){
    return (m*c + n*p)/(m+n);
}

int ycenter(int c, int p, int m, int n){
    return (m*c + n*p)/(m+n);
}
double dist(int x1, int y1, int x2, int y2){
    return sqrt((x1-x2)*(x1-x2) + (y1-y2)*(y1-y2));
}
```

Python + C Code

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

lib = ctypes.CDLL(./libcode.so)

lib.xcenter.argtypes = [ctypes.c_int, ctypes.c_int, ctypes.c_int,
     ctypes.c_int]
lib.xcenter.restype = ctypes.c_int

lib.ycenter.argtypes = [ctypes.c_int, ctypes.c_int, ctypes.c_int,
     ctypes.c_int]
lib.ycenter.restype = ctypes.c_int

lib.dist.argtypes = [ctypes.c_int, ctypes.c_int, ctypes.c_int,
    ctypes.c_int]
lib.dist.restype = ctypes.c_double

c = np.array([-1, -1])
```

Python + C Code

```
p = np.array([1, 0])



x = lib.xcenter(c[0], p[0], 2, -1)
y = lib.ycenter(c[1], p[1], 2, -1)
q = np.array([x, y])
print(The point Q is , q)

r = lib.dist(c[0], c[1], p[0], p[1])

theta = np.linspace(0, 2*np.pi, 200)
circle_x = c[0] + r*np.cos(theta)
circle_y = c[1] + r*np.sin(theta)

fig, ax = plt.subplots()

ax.plot(circle_x, circle_y, color=blue, label=Circle)
ax.scatter([p[0], q[0]], [p[1], q[1]], color=red, label=Diameter
```

Python + C Code

```
ax.scatter(c[0], c[1], color=blue, marker=x, s=100, label=Center)
ax.plot([p[0], q[0]], [p[1], q[1]], g--, label=Diameter)
ax.set_aspect(equal, adjustable=datalim)
ax.set_xlabel(X-axis)
ax.set_ylabel(Y-axis)
ax.text(c[0], c[1], fC({int(c[0])}, {int(c[1])}), fontsize=12,
    color=blue, ha=right, va=bottom)
ax.text(p[0], p[1], fP({int(p[0])}, {int(p[1])}), fontsize=12,
    color=red, ha=left, va=top)
ax.text(q[0], q[1], fQ({int(q[0])}, {int(q[1])}), fontsize=12,
    color=red, ha=right, va=top)
ax.legend()
ax.legend(loc=upper right)
ax.grid(True)
plt.savefig(/Users/bhargavkrish/Desktop/BackupMatrix/
    ee25btech11013/matgeo/7.4.20/figs/Figure_1.png)
plt.show()
```

Python Code

```
import numpy as np
import matplotlib.pyplot as plt
c = np.array([-1, -1])
p = np.array([1, 0])
def diam_opposite(center, point):
    return 2*center - point
def distance(pt1, pt2):
    return np.sqrt((pt1[0]-pt2[0])**2 + (pt1[1]-pt2[1])**2)
q = diam_opposite(c, p)
r = distance(c, p)
print(The point Q is:, q)
print(Radius r:, r)

theta = np.linspace(0, 2*np.pi, 200)
circle_x = c[0] + r*np.cos(theta)
circle_y = c[1] + r*np.sin(theta)
fig, ax = plt.subplots()
```

Python Code

```
ax.plot(circle_x, circle_y, color=blue, label=Circle)
ax.scatter([p[0], q[0]], [p[1], q[1]], color=red, label=Diameter
    Points)
ax.scatter(c[0], c[1], color=blue, marker=x, s=100, label=Center)
ax.plot([p[0], q[0]], [p[1], q[1]], g--, label=Diameter)
ax.text(c[0], c[1], fC({c[0]}, {c[1]}), fontsize=12, color=blue,
    ha=right, va=bottom)
ax.text(p[0], p[1], fP({p[0]}, {p[1]}), fontsize=12, color=red,
    ha=left, va=top)
ax.text(q[0], q[1], fQ({q[0]}, {q[1]}), fontsize=12, color=red,
    ha=right, va=top)
ax.set_aspect(equal, adjustable=datalim)
ax.set_xlabel(X-axis)
ax.set_ylabel(Y-axis)
ax.legend(loc=upper right)
ax.grid(True)
plt.savefig(/Users/bhargavkrish/Desktop/BackupMatrix/
    ee25btech11013/matgeo/7.4.20/figs/Figure_1.png)
plt.show()
```