

## 7.4.39

Kartik Lahoti - EE25BTECH11032

October 1, 2025

# Question

If  $\left(m_i, \frac{1}{m_i}\right)$ ,  $m_i > 0$ ,  $i = 1, 2, 3, 4$  are four distinct points on a circle, then show that  $m_1 m_2 m_3 m_4 = 1$

# Theoretical Solution

Let the circle equation be

$$\|\mathbf{x}\|^2 + 2\mathbf{u}^\top \mathbf{x} + f = 0 \quad (1)$$

where,  $\mathbf{u} = \begin{pmatrix} a \\ b \end{pmatrix}$  with  $a$  and  $b$  as constants.

Let  $\mathbf{P} = \begin{pmatrix} m \\ \frac{1}{m} \end{pmatrix}$  be an arbitrary vector in space.

Putting  $\mathbf{P}$  in the circle, we get

$$\|\mathbf{P}\|^2 + 2\mathbf{u}^\top \mathbf{P} + f = 0 \quad (2)$$

# Theoretical Solution

$$m^2 + \frac{1}{m^2} + 2am + \frac{2b}{m} + f = 0 \quad (3)$$

Let,

$$p(m) = m^4 + 2am^3 + fm^2 + 2bm + 1 = 0 \quad (4)$$

# Theoretical Solution

A general polynomial of degree  $n$ , has companion matrix as

$$\mathbf{C} = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & -c_0 \\ 1 & 0 & 0 & \cdots & 0 & -c_1 \\ 0 & 1 & 0 & \cdots & 0 & -c_2 \\ 0 & 0 & 1 & \cdots & 0 & -c_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & -c_{n-1} \end{pmatrix} \quad (5)$$

The eigen values of the Companion Matrix  $\mathbf{C}$  are the roots of the polynomial.

# Theoretical Solution

For the question ,

$$\mathbf{C} = \begin{pmatrix} 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & -2b \\ 0 & 1 & 0 & -f \\ 0 & 0 & 1 & -2a \end{pmatrix} \quad (6)$$

$$|m\mathbf{I} - \mathbf{C}| = p(m) \quad (7)$$

Here Eigen values of  $\mathbf{C}$  are  $m_i$  where  $i \in \{1, 2, 3, 4\}$

## Introducing Reversal Matrix

$$\mathbf{J} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \quad (8)$$

$$\mathbf{J}^2 = \mathbf{I} \quad (9)$$

The Matrix  $\mathbf{J}$  flips Rows when pre-multiplied and flips column when post-multiplied.

# Theoretical Solution

Then,

$$\left| \frac{1}{m} \mathbf{I} - \mathbf{JCJ} \right| = p \left( \frac{1}{m} \right) \quad (10)$$

Since  $p \left( \frac{1}{m} \right)$  has eigen values  $1/m_i$ , we can say

$$\mathbf{JCJ} = \mathbf{C}^{-1} \quad (11)$$

$$(12)$$



Taking determinant, using 9

$$|\mathbf{C}| = |\mathbf{C}^{-1}| \quad (13)$$

$$|\mathbf{C}|^2 = 1 \quad (14)$$

# Theoretical Solution

Since  $\mathbf{C}$  is a real companion matrix of a monic quartic whose constant term is 1 ,

$$|\mathbf{C}| = (-1)^4 1 \quad (15)$$

Also,

$$|\mathbf{C}| = m_1 m_2 m_3 m_4 \quad (16)$$

$$\therefore m_1 m_2 m_3 m_4 = 1 \quad (17)$$

Hence Proved

# C Code - To find Solution of Circle and Hyperbola

```
#include <math.h>
double calc(double *X , double *Y , double c, double r)
{
    double temp1 , temp2 ,prod;
    temp1 = (r*r + sqrt(pow(r,4) - 4 * pow(c,4)))/2;
    temp2 = (r*r - sqrt(pow(r,4) - 4 * pow(c,4)))/2;
    X[0] = sqrt(temp1);
    X[1] = -sqrt(temp1);
    X[2] = sqrt(temp2);
    X[3] = -sqrt(temp2);
    prod = pow(sqrt(temp1),2) * pow(sqrt(temp2),2);
    for(int i = 0 ; i< 4 ; i++)
        Y[i] = c*c/X[i] ;
    return prod;
}
```

# C Code - Generate Circle

```
void circle_gen(double *X , double *Y , double *P, int n , double
    r)
{
    for (int i = 0 ; i < n ; i++ )
    {
        double theta = 2.0 * M_PI * i / n ;
        X[i] = P[0] + r * cos(theta);
        Y[i] = P[1] + r * sin (theta);
    }
}
```

# C Code - To generate Hyperbola

```
void points_gen (double *X , double a , double b , int n )
{
    double temp = (b - a )/(double)n ;
    for (int i = 0 ; i <= n ; i++ )
    {
        X[i] = a + temp * i ;
    }
}

void hyper_gen (double *X , double *Y , double c , int n )
{
    for(int i = 0; i < n ;i++)
    {
        Y[i] = c*c/X[i];
    }
}
```

# Python Code - 1

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

handc1 = ct.CDLL("./generator.so")

handc1.circle_gen.argtypes = [
    ct.POINTER(ct.c_double),
    ct.POINTER(ct.c_double),
    ct.POINTER(ct.c_double),
    ct.c_int,
    ct.c_double
]

handc1.circle_gen.restype = None
```

# Python Code - 1

```
0 = np.zeros(2 , dtype = np.float64).reshape(-1,1)
X_cic = np.zeros(200, dtype = np.float64).reshape(-1,1)
Y_cic = np.zeros(200, dtype = np.float64).reshape(-1,1)

handc1.circle_gen(
    X_cic.ctypes.data_as(ct.POINTER(ct.c_double)),
    Y_cic.ctypes.data_as(ct.POINTER(ct.c_double)),
    0.ctypes.data_as(ct.POINTER(ct.c_double)),
    200 , 3 )

handc1.points_gen.argtypes = [
    ct.POINTER(ct.c_double),
    ct.c_double,
    ct.c_double,
    ct.c_int]
```

# Python Code - 1

```
handc1.points_gen.restype = None

pt = 400
X_hyper_p = np.zeros(pt,dtype=np.float64).reshape(-1,1)
Y_hyper_p = np.zeros(pt,dtype=np.float64).reshape(-1,1)

handc1.points_gen(
    X_hyper_p.ctypes.data_as(ct.POINTER(ct.c_double)),
    0.1,
    30,
    pt
)
```



# Python Code - 1

```
X_hyper_n = np.zeros(pt,dtype=np.float64).reshape(-1,1)
Y_hyper_n = np.zeros(pt,dtype=np.float64).reshape(-1,1)

handc1.points_gen(
    X_hyper_n.ctypes.data_as(ct.POINTER(ct.c_double)),
    -30,-0.1,pt
)
handc1.hyper_gen.argtypes = [
    ct.POINTER(ct.c_double),
    ct.POINTER(ct.c_double),
    ct.c_double,
    ct.c_int
]
```

# Python Code - 1

```
handc1.hyper_gen.restype = None

handc1.hyper_gen(
    X_hyper_p.ctypes.data_as(ct.POINTER(ct.c_double)),
    Y_hyper_p.ctypes.data_as(ct.POINTER(ct.c_double)),
    1,pt
)

handc1.hyper_gen(
    X_hyper_n.ctypes.data_as(ct.POINTER(ct.c_double)),
    Y_hyper_n.ctypes.data_as(ct.POINTER(ct.c_double)),
    1,pt
)
```

# Python Code - 1

```
plt.figure()
plt.plot(X_cic,Y_cic,"blue",label= "Circle")
plt.plot(X_hyper_p,Y_hyper_p,"red",label="Rectangular Hyperbola")
plt.plot(X_hyper_n,Y_hyper_n,"red")

handc2 = ct.CDLL("./func.so")

handc2.calc.argtypes = [
    ct.POINTER(ct.c_double),
    ct.POINTER(ct.c_double),
    ct.c_double,
    ct.c_double
]
```

# Python Code - 1

```
handc2.calc.restype = ct.c_double

x = np.zeros(4,dtype=np.float64)
y = np.zeros(4,dtype=np.float64)

prod = handc2.calc(
    x.ctypes.data_as(ct.POINTER(ct.c_double)),
    y.ctypes.data_as(ct.POINTER(ct.c_double)),
    1 , 3
)
print("m_1m_2m_3m_4 = ",prod)
plt.scatter(x,y)
vert_labels = [r'$m_1$',r'$m_2$',r'$m_3$',r'$m_4$']
```

# Python Code - 1

```
for i, txt in enumerate(vert_labels):
    plt.annotate(f'({txt},1/{txt})',
                 (x[i], y[i]),
                 textcoords="offset points",
                 xytext=(10,-15),
                 ha='center')

plt.xlabel('$x$')
plt.ylabel('$y$')
```

# Python Code - 1

```
plt.legend(loc='best')
plt.grid()
plt.axis('equal')
plt.xlim([-10/2,10/2])
plt.ylim([-10/2,10/2])
plt.title("7.4.39")
plt.axhline(0, color='black', linewidth=0.7)
plt.axvline(0, color='black', linewidth=0.7)
plt.savefig("../figs/graph1.png")
plt.show()
```

## Python Code - 2

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

from conics.funcs import *
```

## Python Code - 2

```
def intersect_hyperbola_circle(c, r):
    #Find intersection points of  $xy = c^2$  and  $x^2 + y^2 = r^2$ 
    # Coefficients for quadratic in  $X = x^2$ 
    #  $X^2 - r^2X + c^4 = 0$ 
    coeffs = [1, -r**2, c**4]
    roots = np.roots(coeffs)

    points = []
    for X in roots:
        if np.isreal(X) and X >= 0: # valid  $x^2$ 
            x_vals = [np.sqrt(X).real, -np.sqrt(X).real]
            for x in x_vals:
                if x != 0: # avoid division by zero
                    y = c**2 / x
                    points.append((x, y))
    return points
```



## Python Code - 2

```
lt = intersect_hyperbola_circle(1,3)
inter1 = np.array([lt[0][0] , lt[0][1]]).reshape(-1,1)
inter2 = np.array([lt[1][0] , lt[1][1]]).reshape(-1,1)
inter3 = np.array([lt[2][0] , lt[2][1]]).reshape(-1,1)
inter4 = np.array([lt[3][0] , lt[3][1]]).reshape(-1,1)
prod = lt[0][0] * lt[1][0] * lt[2][0] * lt[3][0]
print('m_1m_2m_3m_4 = ',prod)
# xy = c^2
y_n = np.linspace(-30, -0.1 , 400)
y_n = y_n[y_n!=0]
x_n = 1**2/y_n

y_p = np.linspace(0.1, 30 , 400)
y_p = y_p[y_p!=0]
x_p = 1**2/y_p
```

```
plt.figure()
plt.plot(x_p,y_p,"r-",label="Rectangular Hyperbola")
plt.plot(x_n,y_n,"r-")
0 = np.zeros(2,dtype=np.float64).reshape(-1,1)
x_circ = circ_gen(0,3)
plt.plot(x_circ[0,:],x_circ[1:], "blue",label="Circle")

coords = np.block([[inter1,inter2,inter3,inter4]])
plt.scatter(coords[0,:],coords[1,:])
vert_labels = [r'$m_1$',r'$m_2$',r'$m_3$',r'$m_4$']
```

```
for i, txt in enumerate(vert_labels):
    plt.annotate(f'({txt},1/{txt})',(coords[0,i], coords[1,i]),
        textcoords="offset points",xytext=(10,-15),ha='center')

plt.xlabel('$x$')
plt.ylabel('$y$')
plt.legend(loc='best')
plt.grid()
plt.axis('equal')
plt.xlim([-10/2,10/2])
plt.ylim([-10/2,10/2])
plt.title("7.4.39")
plt.axhline(0, color='black', linewidth=0.7)
plt.axvline(0, color='black', linewidth=0.7)
plt.savefig("../figs/graph2.png")
plt.show()
```

