5.2.49

RAVULA SHASHANK REDDY - EE25BTECH11047

October 3, 2025

Question

Find the parameters of the conic

$$36x^2 + 4y^2 = 144.$$

Solution

$$g(\mathbf{x}) = \mathbf{x}^\top \mathbf{V}\mathbf{x} + 2\mathbf{u}^\top \mathbf{x} + f = 0 \tag{1}$$

$$\mathbf{V} = \begin{pmatrix} 36 & 0 \\ 0 & 4 \end{pmatrix}, \quad \mathbf{u} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad f = -144 \tag{2}$$

$$\lambda_1 = 36, \quad \lambda_2 = 4 \tag{3}$$

$$\tag{4}$$

Since $\lambda_1 > \lambda_2$, apply affine transformation

$$\mathbf{P} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \mathbf{x} = \mathbf{P}\mathbf{y} \tag{5}$$

Hence,

$$\lambda_1 = 4, \quad \lambda_2 = 36 \tag{6}$$

$$\mathbf{e}_1 = \mathbf{p}_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \quad \mathbf{e}_2 = \mathbf{p}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \tag{7}$$

Solution

$$f_0 = \mathbf{u}^\top \mathbf{V}^{-1}\mathbf{u} - f = 144 \tag{8}$$

$$e = \sqrt{1 - \frac{\lambda_1}{\lambda_2}} = \sqrt{1 - \frac{4}{36}} = \frac{2\sqrt{2}}{3} \tag{9}$$

$$\text{Major axis} = 2\sqrt{\frac{f_0}{\lambda_1}} = 12 \tag{10}$$

$$\text{Minor axis} = 2\sqrt{\frac{f_0}{\lambda_2}} = 4 \tag{11}$$

Normal vector of directrix:

$$\mathbf{n} = \sqrt{\lambda_2}\,\mathbf{p}_2 = \sqrt{36}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 6 \end{pmatrix} \tag{12}$$

$$\mathbf{p}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad \mathbf{p}_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{13}$$

Solution

$$c = \frac{e\,\mathbf{u}^\top\mathbf{n} \pm \sqrt{e^2\left(\mathbf{u}^\top\mathbf{n}\right)^2 - \lambda_2\left(e^2-1\right)\left(\|\mathbf{u}\|^2 - \lambda_2 f\right)}}{\lambda_2 e\left(e^2-1\right)} \tag{14}$$

$$c = \pm\frac{1}{e}\sqrt{\frac{\lambda_2 f_0}{\lambda_1}} = \pm\frac{1}{e}\sqrt{\frac{36\cdot 144}{4}} = \pm 27\sqrt{2} \tag{15}$$

Foci:

$$\mathbf{F} = \frac{ce^2\mathbf{n} - \mathbf{u}}{\lambda_2} = \pm 4\sqrt{2}\mathbf{e}_2 \tag{16}$$

Directrices:

$$\mathbf{n}^\top\mathbf{x} = \pm 27\sqrt{2} \tag{17}$$

$$6\mathbf{e_2}^\top\mathbf{x} = \pm 27\sqrt{2} \tag{18}$$

$$\mathbf{e_2}^\top\mathbf{x} = \pm\frac{9\sqrt{2}}{2} \tag{19}$$

Solution

Latus rectum:

$$l = \frac{2\sqrt{|f_0 \lambda_1|}}{\lambda_2} = \frac{4}{3} \tag{20}$$

Parameter	Value
$\mathbf{V},\ \mathbf{u},\ f$	$\begin{pmatrix} 36 & 0 \\ 0 & 4 \end{pmatrix},\ \mathbf{0},\ -144$
λ_1, λ_2	$4,\ 36$
f_0	144
e	$\frac{2\sqrt{2}}{3}$
Major axis length	12
Minor axis length	4
Foci	$\mathbf{F} = \pm 4\sqrt{2}\,\mathbf{e}_2$
Directrices	$\mathbf{e}_2^\top \mathbf{x} = \pm \frac{9\sqrt{2}}{2}$
Latus rectum	$\frac{4}{3}$

C Code

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

int main() {
    // Conic: 36x^2 + 4y^2 = 144
    double V[2][2] = {{36,0},{0,4}};
    double u[2] = {0,0};
    double f = -144;

    // Step 1: eigenvalues (already diagonal since V is diagonal)
    double lam1 = 4; // smaller eigenvalue (after swap)
    double lam2 = 36; // larger eigenvalue

    // Step 2: compute f0
    double f0 = -(f); // since u=0, f0 = -f
    printf(f0 = %.2f\n, f0);
```

C Code

```c
    // Step 3: eccentricity
    double e = sqrt(1 - lam1/lam2);
    printf(Eccentricity e = %.5f\n, e);

    // Step 4: semi-axes
    double a = sqrt(f0/lam1);
    double b = sqrt(f0/lam2);
    printf(Semi-major axis a = %.2f\n, a);
    printf(Semi-minor axis b = %.2f\n, b);

    // Step 5: vertices
    printf(Vertices (major): (%.2f,0), (%.2f,0)\n, a, -a);
    printf(Vertices (minor): (0,%.2f), (0,%.2f)\n, b, -b);
```

```c
    // Step 6: directrix constant c (matrix formula simplified)
    double c = (1/e) * sqrt((lam2 * f0)/lam1);
    printf(Directrix constant c = %.2f\n, c);

    // Directrix equations (for n = (0,6))
    printf(Directrices: y = %.2f\n, c/6.0);

    // Step 7: foci
    double Fy = (c * e * e / lam2) * 6.0; // factor 6 from n =(0,6)
    printf(Foci: (0,%.2f), (0,%.2f)\n, Fy, -Fy);

    // Step 8: latus rectum length
    double l = (2 * sqrt(f0 * lam1)) / lam2;
    printf(Latus rectum length = %.2f\n, l);

    return 0;
}
```

Python Direct

```
import numpy as np
import matplotlib.pyplot as plt
from numpy import linalg as LA

# local imports
from libs.line.funcs import *
from libs.conics.funcs import *

# Matrix form: x^T V x + 2u^T x + f = 0
V = np.array(([36,0],[0,4]))
u = np.array(([0,0])).reshape(-1,1)
f = -144

# Get parameters
n,c,F,O,lam,P,e = conic_param(V,u,f)
ab = ellipse_param(V,u,f)
```

Python Direct

```
print(Eigenvalues =, lam)
print(Rotation matrix P =\n, P)
print(Semi-axes a,b =, ab)

# Compute eccentricity from matrix formula
lam1, lam2 = min(lam), max(lam)
f0 = u.T @ LA.inv(V) @ u - f
f0 = f0.item()
ecc = np.sqrt(1 - lam1/lam2)
print(f0 =, f0, eccentricity =, ecc)

# Compute c using big formula
c_val = np.array([
    (ecc*u.T@n + np.sqrt(
        ecc**2*(u.T@n)**2 - lam2*(ecc**2-1)*(LA.norm(u)**2 - lam2
            *f)
    )) / (lam2*ecc*(ecc**2-1)),
```

Python Direct

```
    (ecc*u.T@n - np.sqrt(
        ecc**2*(u.T@n)**2 - lam2*(ecc**2-1)*(LA.norm(u)**2 - lam2
            *f)
    )) / (lam2*ecc*(ecc**2-1))
], dtype=float).flatten()

print(Directrix constants c =, c_val)

# Generate ellipse points
xStandard = ellipse_gen_num(ab[0], ab[1], 100)

# Directrix lines
k1, k2 = -1, 1
x_A = line_norm(n,c_val[0],k1,k2)
x_C = line_norm(n,c_val[1],k1,k2)

# Plot
plt.plot(xStandard[0,:], xStandard[1,:], label='Ellipse')
plt.plot(x_A[0,:], x_A[1,:], label='Directrix 1')
```

Python Direct

```
plt.plot(x_C[0,:], x_C[1,:], label='Directrix 2')

# Plot center, foci
plt.scatter(O[0],O[1], c='r', label='Center O')
plt.scatter(F[0,:],F[1,:], c='g', label='Foci')

# Annotate
plt.annotate(O, (O[0],O[1]), xytext=(5,5), textcoords=offset
    points)
for i in range(F.shape[1]):
    x_coord = round(float(F[0, i]), 2)
    y_coord = round(float(F[1, i]), 2)
    coord_text = fF{i+1} ({x_coord}, {y_coord})
    plt.annotate(coord_text,
```

```
                (x_coord, y_coord),
                xytext=(5, -15), textcoords=offset points)

# Axes setup
ax = plt.gca()
ax.spines['top'].set_color('none')
ax.spines['right'].set_color('none')
ax.spines['left'].set_position('zero')
ax.spines['bottom'].set_position('zero')

plt.legend()
plt.axis('equal')
plt.grid()
plt.show()
```

Python Shared

```python
        import ctypes
import os

# Load shared library
lib = ctypes.CDLL(os.path.abspath(./libellipse.so))

# Define the struct (must match C struct)
class EllipseResult(ctypes.Structure):
    _fields_ = [
        (f0, ctypes.c_double),
        (eccentricity, ctypes.c_double),
        (a, ctypes.c_double),
        (b, ctypes.c_double),
        (c_directrix, ctypes.c_double),
        (Fy, ctypes.c_double),
        (latus_rectum, ctypes.c_double)
    ]
```

Python Shared

```python
# Set return type of the function
lib.compute_ellipse.restype = EllipseResult

# Call the function
res = lib.compute_ellipse()

print(ff0 = {res.f0})
print(fEccentricity = {res.eccentricity})
print(fSemi-major a = {res.a}, Semi-minor b = {res.b})
print(fDirectrix constant c = {res.c_directrix})
print(fFoci y = {res.Fy})
print(fLatus rectum = {res.latus_rectum})
```

Plot

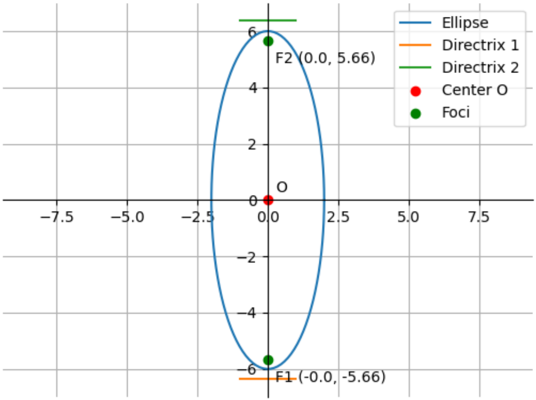
Ellipse
Directrix 1
Directrix 2
Center O
Foci
F2 (0.0, 5.66)
F1 (-0.0, -5.66)
O
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
6
4
2
0
−2
−4
−6

Figure: