MatGeo Assignment 9.2.41

AI25BTECH11007

October 4, 2025

Question

Find the area of the region bounded by the curves $y^2 = 9x$, y = 3x.

Solution

$$\text{Given curves: } y^2 = 9x, \quad y = 3x \tag{1}$$

The conic can be written as

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

where

$$\mathbf{V} = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}, \quad \mathbf{u} = \begin{pmatrix} -\frac{9}{2} \\ 0 \end{pmatrix}, \quad f = 0 \tag{3}$$

The equation of the line is

$$\mathbf{x} = \mathbf{h} + \kappa \mathbf{m} \tag{4}$$

where

$$\mathbf{h} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad \mathbf{m} = \begin{pmatrix} 1 \\ 3 \end{pmatrix} \tag{5}$$

Substituting in $g(\mathbf{x}) = 0$:

$$(\mathbf{h}+\kappa\mathbf{m})^T\mathbf{V}(\mathbf{h}+\kappa\mathbf{m})+2\mathbf{u}^T(\mathbf{h}+\kappa\mathbf{m})+f=0 \tag{6}$$

Expanding,

$$\kappa^2(\mathbf{m}^T\mathbf{V}\mathbf{m})+2\kappa(\mathbf{m}^T(\mathbf{V}\mathbf{h}+\mathbf{u}))+(\mathbf{h}^T\mathbf{V}\mathbf{h}+2\mathbf{u}^T\mathbf{h}+f)=0 \tag{7}$$

Substituting the known values,

$$9\kappa^2-9\kappa=0 \tag{8}$$

$$\kappa_1=0, \quad \kappa_2=1$$

Finally, intersecting points are

$$\mathbf{x}_1=\begin{pmatrix}0\\0\end{pmatrix}+0\begin{pmatrix}1\\3\end{pmatrix}=\begin{pmatrix}0\\0\end{pmatrix} \tag{9}$$

$$\mathbf{x}_2=\begin{pmatrix}0\\0\end{pmatrix}+1\begin{pmatrix}1\\3\end{pmatrix}=\begin{pmatrix}1\\3\end{pmatrix} \tag{10}$$

Thus, the points of intersection are

$$A = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad B = \begin{pmatrix} 1 \\ 3 \end{pmatrix}$$

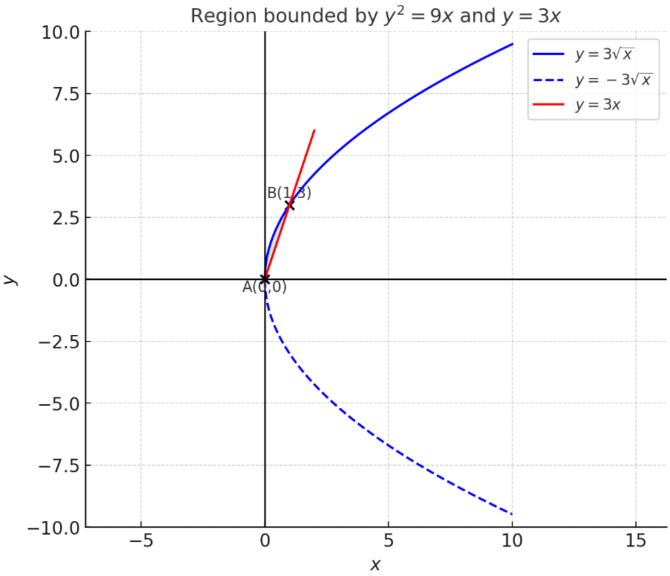
Hence, the required area is

$$A = \int_0^1 (3\sqrt{x} - 3x)\, dx \tag{11}$$

$$\tag{12}$$

$$\boxed{A = \tfrac{1}{2} \text{ square units}}$$

Plot



Region bounded by $y^2 = 9x$ and $y = 3x$
$y = 3\sqrt{x}$
$y = -3\sqrt{x}$
$y = 3x$
B(1,3)
A(0,0)
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−5
0
5
10
15
x
y

C code

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

// Function for the parabola y = 3*sqrt(x)
double parabola(double x) {
    return 3 * sqrt(x);
}

// Function for the line y = 3*x
double line(double x) {
    return 3 * x;
}

// Function to compute area using trapezoidal rule
double area_between_curves(double a, double b, int n) {
    double h = (b - a) / n;
    double area = 0.0;
    for(int i = 0; i <= n; i++) {
        double x = a + i*h;
```

C code

```c
        double y_diff = parabola(x) - line(x);
        if(i == 0 || i == n)
            area += y_diff / 2.0;
        else
            area += y_diff;
    }
    area *= h;
    return area;
}

int main() {
    double a = 0.0, b = 1.0; // limits of integration (
        intersection points)
    int n = 1000; // number of subintervals for numerical
        integration

    double area = area_between_curves(a, b, n);
    printf("Area enclosed between y^2 = 9x and y = 3x: %.6f
        square units\n", area);
```

C code

```c
        // Direct symbolic calculation
    double symbolic_area = 3.0*(2.0/3.0 - 0.5);
    printf("Area calculated symbolically: %.6f square units\n",
        symbolic_area);
    return 0;
}
```

Python code

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

# Define x range for the parabola (both branches)
x = np.linspace(0, 10, 400)
y_upper = 3 * np.sqrt(x)
y_lower = -3 * np.sqrt(x)

# Line y = 3x
x_line = np.linspace(0, 2, 200)
y_line = 3 * x_line

# Plot setup
plt.figure(figsize=(7,6))
plt.plot(x, y_upper, 'b', label=r'$y = 3\sqrt{x}$')
plt.plot(x, y_lower, 'b', linestyle='--', label=r'$y = -3\sqrt{x}
    $')
plt.plot(x_line, y_line, 'r', label=r'$y = 3x$')
```

Python code

```
x_fill = np.linspace(0, 1, 200)
plt.fill_between(x_fill, 3*x_fill, 3*np.sqrt(x_fill), color='
    lightblue', alpha=0.5)
# Mark intersection points
plt.scatter([0,1], [0,3], color='black')
plt.text(0, -0.5, 'A(0,0)', ha='center', fontsize=10)
plt.text(1, 3.3, 'B(1,3)', ha='center', fontsize=10)
plt.axhline(0, color='k', linewidth=1)
plt.axvline(0, color='k', linewidth=1)
plt.title(r'Region bounded by $y^2 = 9x$ and $y = 3x$', fontsize
    =13)
plt.xlabel(r'$x$')
plt.ylabel(r'$y$')
plt.legend(loc='upper right')
plt.grid(True, linestyle='--', alpha=0.6)
plt.axis('equal')
plt.xlim(-1, 10)
plt.ylim(-10, 10)
plt.show()
```