MATGEO 9.9.2.30

EE24BTECH11032 - JOHN BOBBY

Question

Calculate the area under the curve $y = 2\sqrt{x}$ included with the lines $x = 1$ and $x = 0$.

Variable	Description	Value
$\mathbf{m_1}$	direction vector of L_1	$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$
$\mathbf{m_2}$	direction vector of L_2	$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$
$\mathbf{h_1}$	vector passing through L_1	$\begin{pmatrix} 0 \\ 0 \end{pmatrix}$
$\mathbf{h_2}$	vector passing through L_2	$\begin{pmatrix} 1 \\ 0 \end{pmatrix}$
$\mathbf{V}$	Conic parameter	$\begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$
$\mathbf{u}$	Conic parameter	$\begin{pmatrix} -2 \\ 0 \end{pmatrix}$
f	Conic parameter	0

Theory

For a line $\mathbf{x} = \mathbf{h} + k\mathbf{m}$,the intersection of the line with a conic with parameters $\mathbf{V}, \mathbf{u}, \mathbf{h}, \mathbf{m}$ and h is given by $\mathbf{x} = \mathbf{h} + k_i\mathbf{m}$
$\mathbf{V} = \|x\|^2\mathbf{I} - e^2\mathbf{n}\mathbf{n}^\top$,
$\mathbf{u} = ce^2\mathbf{n} - \|n\|^2\mathbf{F}$,
$f = \|x\|^2\|F\|^2 - c^2e^2$
$L_1 : x = 0$
$L_2 : x = 1$

$$k_i = \frac{1}{\mathbf{m}^\top\mathbf{V}\mathbf{m}}\left(-\mathbf{m}^\top\left(\mathbf{V}\mathbf{h} + \mathbf{u}\right) + \sqrt{\left[\mathbf{m}^\top\left(\mathbf{V}\mathbf{h} + \mathbf{u}\right)\right]^2 - g\left(\mathbf{h}\right)\left(\mathbf{m}^\top\mathbf{V}\mathbf{m}\right)}\right) \tag{1}$$

on solving for L_1 and L_2

$$k_1 = 0, k_2 = 2 \tag{2}$$

$$\mathbf{A} = \mathbf{h_1} + k_1\mathbf{m_1} = \begin{pmatrix}0\\0\end{pmatrix} + 0\begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}0\\0\end{pmatrix} \tag{3}$$

$$\mathbf{B} = \mathbf{h_2} + k_2\mathbf{m_2} = \begin{pmatrix}1\\0\end{pmatrix} + 2\begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}1\\2\end{pmatrix} \tag{4}$$

Thus the area under the curve included with the lines $x = 1$ and $x = 0$ is given by

Plot

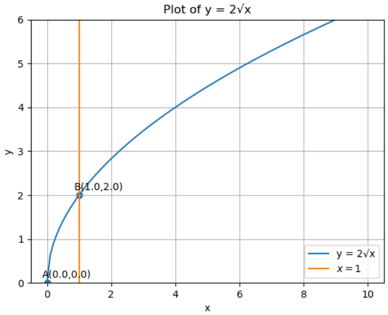

Figure: Plot of the parabola

Code(parabola.c)

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

void compute_values(double* x, double* y, int n) {
    for (int i = 0; i < n; i++) {
        if (x[i] < 0) {
            y[i] = NAN;
        } else {
            y[i] = 2 * sqrt(x[i]);
        }
    }
}
double compute_value(double x){
        return 2*sqrt(x);
}
```

Code(area.c)

```
#include <stdio.h>
#include <math.h>
double func(double x){
  return 2*sqrt(x);
}
double computeArea(double a, double b, int n) {
    double delta_x = (b - a) / n; // Step size
    double area = 0.0;
    for (int i = 0; i < n; i++) {
        double x = a + i * delta_x;
        double y = func(x);
        area += y* delta_x; // Add the area of each slice
    }
    return area;
}
```

Code(area.c)

```
int main(){
  int a=0;
  int b=1;
  FILE *file;
  file =fopen("area.txt","w");
  if (file == NULL) {
        printf("Error opening file!\n");
        return 1; //
    }
  fprintf(file, "The area enclosed by the parabola between the lines
  ↪  x=0 and x=1 is %lf", computeArea(a,b,1000));
  fclose(file);
  return 0;
```

Output(area.txt)

The area enclosed by the parabola between the lines x=0 and x=1 is 1.332320

Code(plot.py)

```
import sys
sys.path.insert(0, '/home/john-bobby/MyRepos/matgeo/codes/CoordGeo')
import numpy as np
import matplotlib.pyplot as plt
import ctypes
from line.funcs import *
from triangle.funcs import *
from conics.funcs import *
math_functions = ctypes.CDLL('./parabola.so')
math_functions.compute_values.argtypes = (ctypes.POINTER(ctypes.c_double),
↪  ctypes.POINTER(ctypes.c_double), ctypes.c_int)
math_functions.compute_value.argtypes = [ctypes.c_double]
math_functions.compute_value.restype = ctypes.c_double
x = np.linspace(0, 10, 100)
y = np.zeros_like(x)
A = np.array(([0, math_functions.compute_value(0)])).reshape(-1, 1)
B = np.array(([1, math_functions.compute_value(1)])).reshape(-1, 1)
C = np.array(([1, -1])).reshape(-1, 1)
D = np.array(([1, 7])).reshape(-1, 1)
x_1 = line_gen(C, D)
```

Code(plot.py)

```
math_functions.compute_values(x.ctypes.data_as(ctypes.POINTER(ctypes.c_double))
,y.ctypes.data_as(ctypes.POINTER(ctypes.c_double)), len(x))
points=np.block([[A,B]])
plt.ylim([0, 6])
plt.plot(x, y, label='y = 2x')
plt.plot(x_1[0, :], x_1[1, :], label='$x=1$')
plt.annotate(f"A(0.0,{math_functions.compute_value(0)})", (0,
↪  math_functions.compute_value(0)), textcoords="offset points", xytext=(20,
↪  5), ha='center')
plt.annotate(f"B(1.0,{math_functions.compute_value(1)})", (1,
↪  math_functions.compute_value(1)), textcoords="offset points", xytext=(20,
↪  5), ha='center')
plt.scatter(tri_coords[0,:], tri_coords[1,:])
plt.title('Plot of y = 2x')
plt.xlabel('x')
plt.ylabel('y')
plt.legend()
plt.grid()
plt.savefig("/home/john-bobby/MyRepos/EE1030/Assignment5/Figs/Fig1.png")
```