Ho Chi Minh City University of Technology

Calculus Report

Semester: 212

Group: CC02

Team: 14

Topic: 2

Lecturer: Phan Thi Huong

Name	Student ID
Bùi Minh Đức	2052959
Lê Trọng Đức	2152523
Nguyễn Minh Đức	2052961
Tô Xuân Bình	2052885

Table of Contents

Exercise 1

Problems

Given the surface S: $x - 3y + 2z = 14$

(a) Sketch the surface S.

(b) Find the shortest distance from S to the point (0, 1, 1) and at which point ?

Manually Solutions

We have: $\mathrm{x} - 3\mathrm{y} + 2\mathrm{z} = 14 \Rightarrow \mathrm{z} = 7 - \frac{1}{2}\mathrm{x} + \frac{3}{2}\mathrm{y}$

Consider the point S'(x, y, z) is a point belong S, then the distance from S' to the point (0,1,1) is:

$$\mathrm{d} = \sqrt{\mathrm{x}^2 + (\mathrm{y}-1)^2 + (\mathrm{z}-1)^2} = \sqrt{\mathrm{x}^2 + (\mathrm{y}-1)^2 + \left(6 - \frac{1}{2}\mathrm{x} + \frac{3}{2}\mathrm{y}\right)^2}$$

Let:

$$\mathrm{f}(\mathrm{x},\mathrm{y}) = \mathrm{x}^2 + (\mathrm{y}-1)^2 + \left(6 - \frac{1}{2}\mathrm{x} + \frac{3}{2}\mathrm{y}\right)^2$$

$$\mathrm{f}_\mathrm{x} = \frac{5}{2}\mathrm{x} - \frac{3}{2}\mathrm{y} - 6 = 0 \Rightarrow \frac{5}{2}\mathrm{x} - \frac{3}{2}\mathrm{y} = 6$$

$$\mathrm{f}_\mathrm{y} = \frac{13}{2}\mathrm{y} - \frac{3}{2}\mathrm{x} + 16 = 0 \Rightarrow -\frac{3}{2}\mathrm{x} + \frac{13}{2}\mathrm{y} = -16$$

$$\Rightarrow \mathrm{x} = \frac{15}{14}, \mathrm{y} = -\frac{31}{14} \Rightarrow \mathrm{z} = \frac{22}{7} \Rightarrow \mathrm{P}\left(\frac{15}{14}, -\frac{31}{14}, \frac{22}{7}\right)$$

$$\mathrm{f}_{\mathrm{xx}} = \frac{5}{2} = \mathrm{A}, \mathrm{f}_{\mathrm{xy}} = -\frac{3}{2} = \mathrm{B}, \mathrm{f}_{\mathrm{yy}} = \frac{13}{2} = \mathrm{C} \Rightarrow \Delta = \mathrm{AC} - \mathrm{B}^2 = 14 > 0, \mathrm{f}_{\mathrm{xx}} > 0$$

Conclusion The shortest distance from S to the point (0,1,1) $\mathrm{d} = 4.0089$ at point $\mathrm{P}(\frac{15}{14}, -\frac{31}{14}, \frac{22}{7})$.

Python Solution

In this section, we solve the problem with a differrent idea. That is using the dot product to find the projection point, because the shortest distance is the distance between the point not belong and the projection point.

Now we will throughly explain the code.

Firtly, we will import the needed modules and the packages.

```
import math
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
```

Next, We will define the class Point, which will store the x, y, z coordinate of a point; and also some properties of a point, such as add together or minus a vector to reveive another point.

```
class Point:
    def __init__(self, x, y, z):
        self.x = x
        self.y = y
        self.z = z

    def __repr__(self):
        outstr = "Point( " + str(round(self.x,4))+ ", " + str(round(self.y,4))+ ", " +
str(round(self.z,4)) + " )"
        return outstr

    def __sub__(self, other):
        if isinstance(other, Vector):
            return Point(self.x - other.x,
                         self.y - other.y,
                         self.z - other.z)

        return Vector(self.x - other.x,
                      self.y - other.y,
                      self.z - other.z)

    def __add__(self, other):
        if isinstance(other, Vector):
            return Point(self.x + other.x,
                         self.y + other.y,
                         self.z + other.z)

        return Vector(self.x + other.x,
                      self.y + other.y,
                      self.z + other.z)

    def dis(self,other):
        d = math.sqrt((self.x - other.x)**2 + (self.y - other.y)**2 + (self.z -
other.z)**2)
        return d
```

Next, we will define the class Vector, which will store the x, y, z coordinate of a vector; and also properties of a vector, such as add together or minus a vector to reveive another Vector.

```
class Vector:
    def __init__(self, x, y, z):
        self.x = x
        self.y = y
        self.z = z

    def __repr__(self):
        outstr = "Vector( " + str(round(self.x,4))+ ", " + str(round(self.y,4))+ ", " +
str(round(self.z,4)) + " )"
        return outstr

    def norm(self):
        d = math.sqrt(self.x**2 + self.y**2 + self.z**2)
        return Vector(self.x / d, self.y / d, self.z / d)

    def __mul__(self, other):
        if isinstance(other, Vector):
            return (self.x * other.x + self.y * other.y + self.z * other.z)

        return Vector(self.x * other, self.y * other, self.z * other)
```

Then we will start into our main program. At the beginning, we set the domain for x and y, and we also declare the surface function.

```
a = int(14)
x_data = np.linspace(-25, 25, 15)
y_data = np.linspace(-25, 25, 15)
x, y = np.meshgrid(x_data, y_data)
z = ((a - x + 3*y)/2)
```

Then, we set the label for x axis and y axis and we set the range limit for the graph. At here. We set the min value is -40 and the max value is 40, the data isn't too big so that the program will run faster.

```
ax = plt.axes(projection="3d")
ax.set_xlabel("x")
ax.set_ylabel("y")
ax.set_zlabel("z")
ax.set_xlim([-40, 40])
ax.set_ylim([-40, 40])
ax.set_zlim([-40, 40])
ax.plot_surface(x, y, z, rstride=1, cstride=1, cmap='viridis', edgecolor='none')
```

Next step, we find the projection coordinate.

```
s = Point(0, 1, 1)
plane = (
    Point(a, 0 ,0),
    Point(0, -a/3, 0),
    Point(0, 0, a/2)
)

n = Vector(1, -3, 2).norm()
s1 = s - plane[0]
re = (n*s1)/(n*n)
d = n*re
pp = s - d
ax.quiver(s.x,s.y,s.z,pp.x,pp.y,pp.z)
```

And for the last step, we print the output.

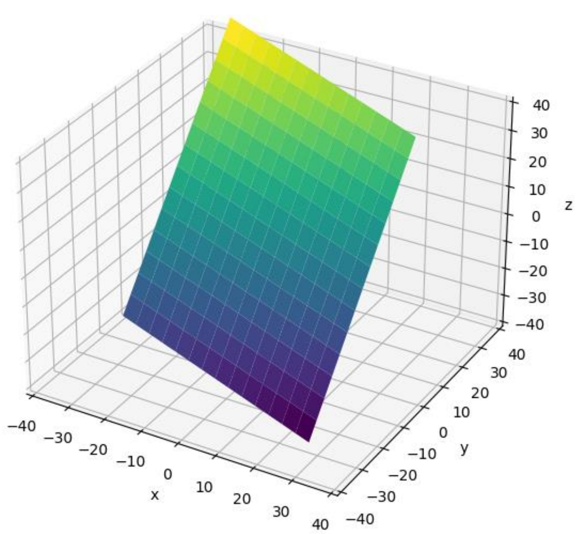
```
print("The shortest distance from S the point (0, 1, 1) is: " + str(round(s.dis(pp),4)))
print("And the point we need to find is " + str(pp))
plt.show()
```

Result

The point we need to find:

```
The shortest distance from S the point (0, 1, 1) is: 4.0089
And the point we need to find is Point( 1.0714, -2.2143, 3.1429 )
```

And the graph is:



Exercise 2

Problems

Let $W(s,t) = F(u(s,t), v(s,t))$, where F, u and v are differentiable, and

$u(1, 0) = 2$	$v(1, 0) = 3$
$u_s(1, 0) = -2$	$v_s(1, 0) = 5$
$u_t(1, 0) = 6$	$v_t(1, 0) = 4$
$F_u(2, 3) = -1$	$F_v(2, 3) = 14$

Find $W_s(1, 0)$ and $W_t(1, 0)$

Manually Solution

For this problem, we apply the chain rule for function of several variables.

Suppose that $z = f(u, v)$ is a differentiable function of u and v, where $u = u(x, y)$ and $v = v(x, y)$ are differentiable functions of x, y. Then

$$\boxed{\frac{\partial z}{\partial x} = \frac{\partial z}{\partial u}\frac{\partial u}{\partial x} + \frac{\partial z}{\partial v}\frac{\partial v}{\partial x}, \quad \frac{\partial z}{\partial y} = \frac{\partial z}{\partial u}\frac{\partial u}{\partial y} + \frac{\partial z}{\partial v}\frac{\partial v}{\partial y}} \qquad (1)$$

$W_s(1, 0) = F_u(2, 3).u_s(1, 0) + F_v(2, 3).v_s(1, 0)$ = (-1).(-2) + 14.5 = 72

$W_t(1, 0) = F_u(2, 3).u_t(1, 0) + F_v(2, 3).v_t(1, 0)$ = (-1).6 + 14.4 = 50

Python Solution

For the solution performed by Python, the way we apply still the same

```
a = int(14)
Ws = -2*-1 + a*5
Wt = -1*6 + a*4
```

```python
print("Ws(1, 0) is: " + str(round(Ws,4)))
print("Wt(1, 0) is: " + str(round(Wt,4)))
```

Result

The screen will print

```
Ws(1, 0) is: 72
Wt(1, 0) is: 50
```

Exercise 3

Problems

Let E be the tetrahedron bounded by the planes: $x = 0$, $y = 0$, $z = 0$, $x + y + z = 14$.

(a) Sketch the solid E.
(b) Given the density function $\rho(x, y, z) = 2y$. Find the mass of the solid E with the given density function ρ.

Manually Solution

For this problem. We use the normal triple integral to calculate the mass.

$$\boxed{\iiint_E f(x, y, z)\, dV = \int_a^b \int_{g_1(x)}^{g_2(x)} \int_{u_1(x, y)}^{u_2(x, y)} f(x, y, z)\, dz\, dy\, dx}$$

$$0 \le z \le 14 - x - y$$

$$0 \le y \le 14 - x$$

$$0 \le x \le 14$$

Mass of the solid: $E = \int_0^{14} dx \int_0^{14-x} dy \int_0^{14-x-y} 2y\, dz$

$$E = \int_0^{14} dx \int_0^{14-x} 2y(14 - x - y) dy$$

$$E = \int_0^{14} \frac{1}{3}(14 - x)^3 dx = \frac{9604}{3}$$

Matlab Solution

For the solution performed by Matlab

```
clc;
clf;
s = linspace(0, 14, 30);
s1 = meshgrid(s);
t1 = [];
```

Calculate the value of y

```
for i=1:length(s)
    tam = linspace(0, 14-s(i), 30);
    t1 = [t1 tam'];
end
x = s1; y = t1; z = 14-x-y; z1 = 0*x; x1 = 0*x; y1 = 0*x;
hold on
```

Draw the solid bounded by x, x_1, y, y_1, z, z_1

```
surf(x, y, z, 'FaceColor', 'g', 'FaceAlpha' , 0.3);
surf(x, y, z1, 'FaceColor', 'r', 'EdgeColor', 'none');
surf(x1, y, z, 'FaceColor', 'b', 'FaceAlpha' , 0.3);
surf(x, y1, z, 'FaceColor', 'y', 'FaceAlpha' , 0.3);
xlabel('x'); ylabel('y'); zlabel('z');
view(120, 12)
grid on
rotate3d on
```

Calculate the mass of the solid by using integral function

```
syms x y z
mass = int(int(int(2*y,z,0,14-x-y),y,0,14-x),x,0,14);
```

Print the result

```
disp('The mass of the solid E is ');
disp(mass);
```

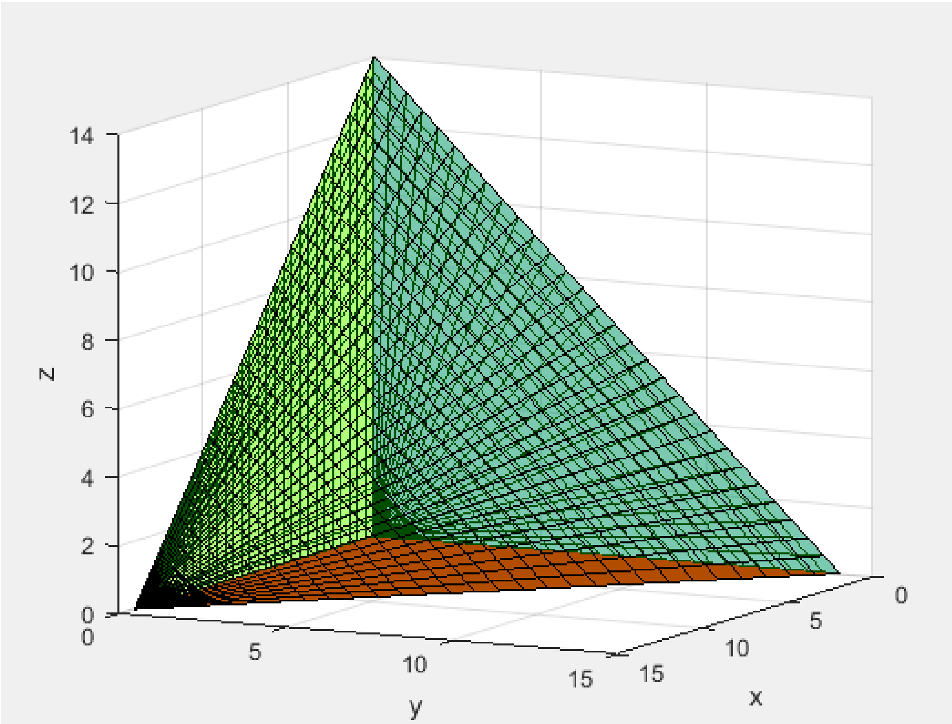
Result

The command window will print:

Command Window

```
The mass of the solid E is
9604/3
```

And the figure:



Exercise 4

<u>Problems</u>

Given S be the surface of the paraboloid $x^2 + y^2 + z = 14$ that lies below the plane $z = 1$, with upward orientation.

(a) Sketch the surface S.
(b) Given the vector field $F(x, y, z) = z\, tan^{-1}(y^2)\mathbf{i} + z^3\, ln(x^2 + 1)\mathbf{j} + z\mathbf{k}$. Find the flux of F across the surface S.

<u>Manually Solution:</u>

$\boxed{8}$ **Definition** If **F** is a continuous vector field defined on an oriented surface S with unit normal vector **n**, then the **surface integral of F over *S*** is

$$\iint_S \mathbf{F} \cdot d\mathbf{S} = \iint_S \mathbf{F} \cdot \mathbf{n}\, dS$$

This integral is also called the **flux** of **F** across S.

For this problem, we apply the surface integral formula to calculate the flux

$$1 \le z \le 14 - x^2 - y^2$$

The flux of $F = + \iiint dxdydz = \iint_{dxy} dA \int_1^{14-x^2-y^2} dz = \iint_{dxy} 13 - x^2 - y^2 \, dA$

The projection of S on xy plane : $x^2 + y^2 = 13$

$$x = r\cos\varphi$$

$$y = r\sin\varphi$$

$$0 \le r \le \sqrt{13}$$

$$0 \le \varphi \le 2\pi$$

$$\Rightarrow \text{The flux } F = \int_0^{2\pi} d\varphi \int_0^{\sqrt{13}} (13 - r^2\cos\varphi^2 - r^2\sin\varphi^2).rdr$$

$$= \int_0^{2\pi} d\varphi \int_0^{\sqrt{13}} (13 - r^2).rdr$$

$$= \int_0^{2\pi} \frac{13}{2}.(\sqrt{13})^2 - \frac{(\sqrt{13})^4}{4} d\varphi = \frac{169}{2}\pi$$

Matlab Solution

For the solution performed by Matlab

```
clc;
clf;
hold on;
```

Output spaced values of phi and r

```
phi = linspace(0,2*pi,30); r = linspace(0,sqrt(13),30);
```

Transforms the domain specified by vectors r and phi into arrays r and phi

```
[r, phi] = meshgrid(r,phi);
x = r.*cos(phi); y = r.*sin(phi);
z = 14- x.^2- y.^2; z1 = cos(phi).^2+sin(phi).^2;
```

Draw the plane bounded by x, y, z

```
surf(x,y,z,'FaceColor','g','FaceAlpha',0.3);
```

Draw the plane bounded by x, y, z_1

```
surf(x,y,z1,'FaceColor','b','FaceAlpha',0.3);
phi=linspace(0,2*pi,30); z2=linspace(0,1,30);
```

Transfoms the domain specified by vectors z_2 and phi

```
[z2, phi] = meshgrid(z2,phi);
x1=sqrt(13).*cos(phi); y1=sqrt(13).*sin(phi);
```

Draw the plane bounded by x_1, y_1, z_2.

```
surf(x1,y1,z2,'FaceColor','r','FaceAlpha',0.3);
```

Labeling x, y, z axis

```
xlabel('x'); ylabel('y'); zlabel('z');
view (13,28)
grid on
rotate3d on
```

Calculating the flux

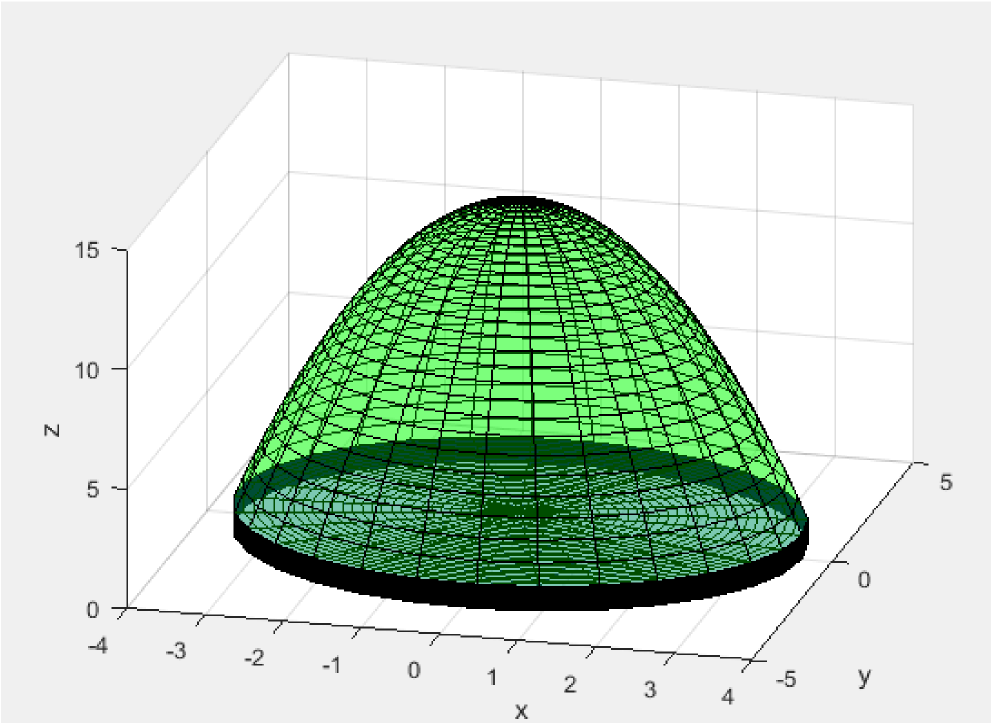
```
fun = @(a,b,c) 1 + 0.*a;
xmin = -sqrt(13);
xmax = sqrt(13);
ymin = @(a)-sqrt(13 - a.^2);
ymax = @(a) sqrt(13 - a.^2);
zmin = @(a,b) 1 + 0.*a;
zmax = @(a,b) 14 - a.^2 - b.^2;
q = integral3(fun,xmin,xmax,ymin,ymax,zmin,zmax,'Method','tiled');
disp(['The flux F is ' , num2str(q)]);
```

<u>Result</u>

The command window will print:

```
Command Window
  The flux F is 265.4646
fx >>
```

And the figure:



Exercise 5

Problems

Find the work done by the force field in moving an object along a part of the circle $x^2 + y^2 = 14$ from $x = 1$ to $x = 0$, where:

$$F(x, y) = 2y^{3/2}\mathbf{i} + 3x\sqrt{y}\mathbf{j}.$$

Manually Solution

For solving this solution, we calculating the work done by using line integral.

13 Definition Let **F** be a continuous vector field defined on a smooth curve C given by a vector function $\mathbf{r}(t)$, $a \leqslant t \leqslant b$. Then the **line integral of F along *C*** is

$$\int_C \mathbf{F} \cdot d\mathbf{r} = \int_a^b \mathbf{F}(\mathbf{r}(t)) \cdot \mathbf{r}'(t)\, dt = \int_C \mathbf{F} \cdot \mathbf{T}\, ds$$

$$x = \sqrt{14}\, cos\, t$$

$$y = \sqrt{14}\, sin\, t$$

$$\cos^{-1}(\frac{1}{\sqrt{14}}) \leq \mathrm{t} \leq \frac{\pi}{2}$$

The work done:

$$W = \int_{\cos^{-1}(\frac{1}{\sqrt{14}})}^{\frac{\pi}{2}} (2.(\sqrt{14}\, sin\, t)^{3/2}.(-\sqrt{14}\, sin\, t) + 3.\sqrt{14}\, cos\, t\, .(\sqrt{14}\, sin\, t)^{1/2}.\sqrt{14}\, cos\, t)\, dt$$

$$= -13.6927(J)$$

Python Solution

First we will import the needed packages and modules.

```
from math import cos
from math import sin
from math import acos
from math import pi
from math import sqrt
import os
import scipy.integrate as integrate
os.system('cls')
a = sqrt(14)
```

Next we will define the parameter function of x and y according to t.

```
def fun_x(x):
    return sqrt(14)*cos(x)

def fun_y(x):
    return sqrt(14)*sin(x)
```

Then we define a function to calculate differential equation

```
def d_fun(x, val):
    h = 1e-5
    if val=='x':
        return (fun_x(x+h)-fun_x(x-h))/(2*h)
    elif val=='y':
        return (fun_y(x+h)-fun_y(x-h))/(2*h)
```

Last step, we calculate the work done

```
result = integrate.quad(lambda t: 2*(a*sin(t))**1.5*d_fun(t,'x') +
3*a*cos(t)*(a*sin(t))**0.5*d_fun(t,'y'), acos(1/a), pi/2)[0]
print("The work done is %.4f"%result + " (J)")
```

Result

The command window will print:

```
The work done is -13.6927 (J)
```

This also mark the end of our report!

Thanks for reading uptill here!

Source and reference

- Mutivariable Calculus 7E by James Stewart
- https://www.mathworks.com/
- https://docs.python.org/3/library/numeric.html

<< If you want to access the full code, you can access our repository on our Github via this link: The full code is here >>