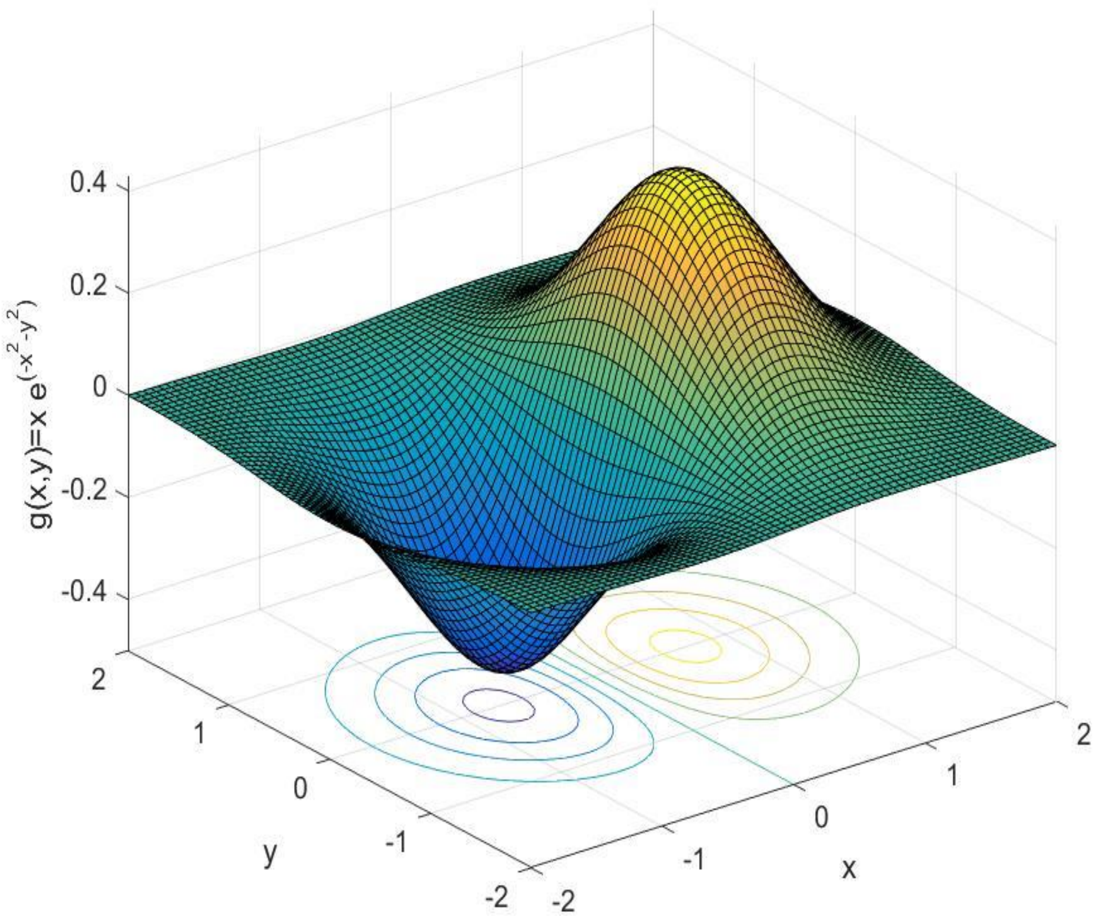
UNIVERSITY OF TORONTO
FACULTY OF APPLIED SCIENCE & ENGINEERING
First Year Program – Core 8 and TrackOne

FIRST YEAR PROGRAM
ENGINEERING PROBLEM SOLVING LABS

MAT188: Laboratory #9
Three-Dimensional Plots



THREE-DIMENSIONAL PLOTS

In this lab, you will learn how to create three-dimensional plots to visualize data and use them to solve engineering problems. Three-dimensional plots are a useful tool for understanding sophisticated problems, allowing you to take experimental data and create a complex visual representation.

Learning Outcomes

By the end of this lab students will…

1) Know how to create a three-dimensional plot in MATLAB using both numeric and symbolic computation techniques, and
2) Further develop the ability to use matrix indexing to better understand how a function behaves.

Preparation *(Required to do* ***before*** *you come to the laboratory session)*

1. Read through this lab document.
2. Watch the posted ***Lab #9 Video Introduction, Part 1 and Part 2.***
3. Come up with two single variable functions that you would like to plot. You could make these up or perhaps you could find a function that is used to represent a physical phenomenon (e.g., Newton's law of gravitation, Einstein's (really Lorentz's) time-dilation formula, the normal distribution, etc.):

   $y_1(x) =$ ______________________________

   $y_2(x) =$ ______________________________

Related Reference Materials *(Not required, but may be a helpful resource)*

http://www.mathworks.com/help/matlab/examples/creating-3-d-plots.html
http://www.mathworks.com/help/matlab/visualize/representing-a-matrix-as-a-surface.html

Review – Labs 1-8

In previous labs, you have learned about how to work with matrices in MATLAB, and specifically how to:

1) *Define matrices* using semi-colons to separate the rows:

   $A = \begin{bmatrix} -2 & 0 & 1 \\ 4 & -8 & 3 \end{bmatrix}$ can be defined using: `>> A = [-2 0 1;4 -8 3];`

   $B = \begin{bmatrix} 0 & 8 \\ -1 & 5 \\ 0 & 4 \end{bmatrix}$ can be defined using: `>> B = [0 8;-1 5;0 4];`

2) *Doing basic matrix calculations and operations* including:
   Scaling (`3*A` or `A+5`), Addition (`A+3*A`), subtraction (5*`A-A`), matrix multiplication (`A*B`), element-by-element multiplication (`C.*C`), and transposes (`A'`).

3) *Index certain parts of a matrix* using the colon operator (:), such as:

```
>> A(2,1)
ans =
       4
>> A(2,2:3)
ans =
   -8   3
```

```
>> B(:,1)
ans =
        0
       -1
        0
```

4) *Assign values to specific parts of a matrix* through commands such as:

```
>> A(2,:) = x.*exp(-2*x);
```

MATLAB Skills and Knowledge: Review and Maximum and Minimum Values

Defining Vectors and Matrices to Numerically and Graphically Represent Functions

For the two arbitrary functions that you identified above in your preparation, $y_1(x)$ and $y_2(x)$ write a short script that will:

(a) Create a single 2 x n matrix, `Y`, that contains the values of these two functions over a set domain for the independent variable x (with n data points over this domain),

(b) Plot these two functions in a single figure over this domain through the use of different line types. This plot should be properly labeled and titled, and have a legend.

(c) Use the MATLAB commands `max` and `min` to find the maximum and minimum values of these two functions, $y_1(x)$ and $y_2(x)$ within the domain that you have chosen. ***Hint***: Review the online documentation for these functions at `max` and `min`, and the transpose operator, `transpose` may be useful to you. How is `max(Y)` different from `max(transpose(Y))`?

(d) Now use proper indexing of the matrix to plot these functions over half the domain.

MATLAB Skills and Knowledge: Three-Dimensional Plots

MATLAB can create many different kinds of 3D plots, but we will focus on the basic plots in this lab.

First, let's create a surface plot of the two-variable function $f(x, y) = z = y^2 - x^2$ over the domain $-3 \leq x \leq 3$ and $-3 \leq y \leq 3$.

We will start by defining the domains of the input variables (x and y) over the same values, but with different step sizes (the reason for this will be obvious soon):

```
>> domain_x = [-3:1:3]
>> domain_y = [-3:0.5:3]
```

In earlier labs, when you created 2D plots with one input variable, the input was a one-dimensional array (one vector). Now that we are creating a 3D plot with two input variables, each input must be a matrix (i.e., two-dimensional arrays).

The vectors `domain_x` and `domain_y` specify the domain of function f. Since this is a two-variable function, it applies to a set of (x, y) points in which x and y belong to `domain_x` and `domain_y` respectively.

The `meshgrid` function will allow you easily create a set of all pairs of x and y values, i.e., (x, y) points on the grid defined by domain $-3 \leq x \leq 3$ and $-3 \leq y \leq 3$. Notice how the function returns two matrices as outputs, which are assigned to the two variables `X` and `Y`.

```
>> [X,Y] = meshgrid(domain_x, domain_y)
```

What are the sizes of `X` *and* `Y`*?*

How do these compare to the sizes of `domain_x` *and* `domain_y`*?*
Look at the elements of `X` *and* `Y` *compared to* `domain_x` *and* `domain_y`*. Can you explain the differences?*

Now create the matrix `Z` with the value the function $f(x,y) = z$ over this domain. Notice how the dot operators are used for element-by-element operations to ensure that the function z is evaluated at each point within the (x, y) set.

```
>> Z = Y.^2-X.^2
```

Note: ***We have to use the matrices*** `X` and `Y` created through `meshgrid`, ***not the vectors*** `domain_x` and `domain_y`.

Observe how the different *__rows__* of `X`, `Y`, `Z` correspond to different y values, while the different *__columns__* correspond to different x values.

This means that as you move across the columns within a row, y is constant while x changes. As you move down the rows within a column, x is constant while y changes.

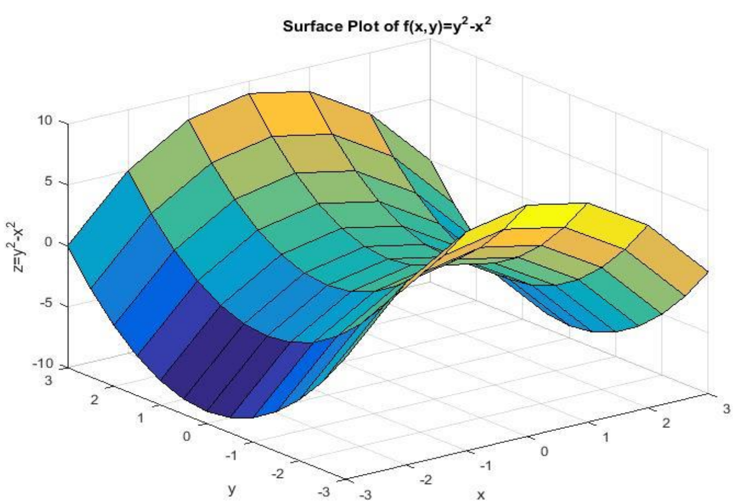
Now that we defined the function's values at these discrete points in the grid, we can plot it as a 3D figure. The two most common MATLAB functions to create 3D surface plots are `surf` and `mesh`, which have the same input syntax. As described in the MATLAB Primer document (Page 1-25):

> "`surf` displays both the connecting lines and the faces of the surface in color. `mesh` produces wireframe surfaces that color only the lines connecting the defining points."

Again note: For these plot commands, ***we have to use the matrices*** `X` and `Y` created through `meshgrid`, ***not the vectors*** `domain_x` and `domain_y`.

Let's use the `surf` (surface) plot first:

```
>> surf(X, Y, Z);
>> xlabel('x');ylabel('y');zlabel('z');
```



Experiment with the "Rotate 3D" button, ,in the toolbar to view the figure from different angles.

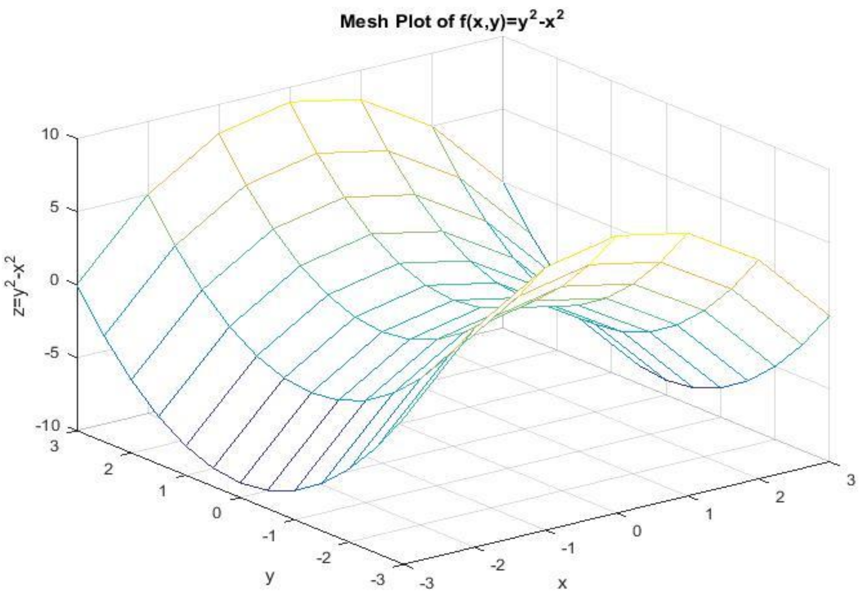
Observe how many "pieces", or segments there are with respect to the *x* and *y* domains.
How does this relate to the sizes of `domain_x` *and* `domain_y`*?*

There are many ways to change how the plot is displayed, and we will explore some of them. ***Try each of the following commands*** and use `help` or search MATLAB documentation to find out what each does.

`shading flat`
`shading interp`
`shading faceted` (the default)

Now let's try the `mesh` plot.

```
>> mesh(X, Y, Z);
```



Variations and Symbolic Plots

There are many other types of commands for creating 3D plots in MATLAB. These include:

Numerical methods: `surfc(X,Y,Z)`, `meshc(X,Y,Z)`, and `meshz(X,Y,Z)`

Symbolic methods:
`fsurf(f,[xmin xmax ymin ymax])`,
`fmesh(f,[xmin xmax ymin ymax])`, and
`fplot3(xt,yt,zt)` (particularly useful for plotting parametric curves in 3D)

Quickly edit your earlier commands to try out some of these so you can see the differences.

Pick One of the Two exercises and submit by Wednesday March 21st 2018 11:59pm:

Submission must be individual work and no more than 2 pages.

1) **Function Plotting:** *Evaluate the behavior of* $g(x,y) = xe^{-x^2-y^2}$ *with a 3D plot and 2D plots using proper indexing, as specified below:*

Consider the function $g(x,y) = xe^{-x^2-y^2}$

a) Create a 3D `surfc` plot of this function for $-2 \leq x \leq 2$ and $-2 \leq y \leq 2$.

b) Create the 2D plots of following functions and save them in one figure.

i. $g\,(x\,,-2)$

ii. $g\,(x,0)$

iii. $g\,(-1,y)$

You will have to first identify how to properly index the matrix `Z` to create this figure. For example, to do the first plot you should use the command:

```
plot(domain_x,Z(1,:),'b-');
```

Since the first row of `Z` corresponds to the constant value of $y = -2$ with x changing.

Observe how the ROWS of Z correspond to different y values, and the COLUMNS of Z correspond to different x values. This means that as you move across the columns for a fixed row, your y is constant and your x changes.

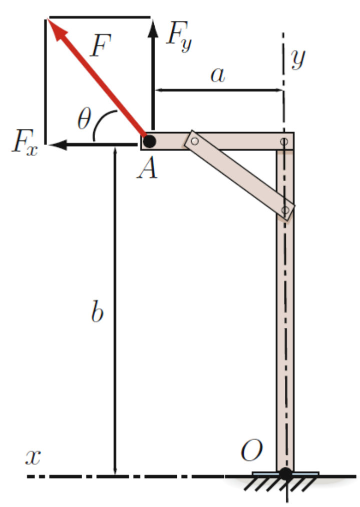
c) Use the `min` and `max` commands to find the maximum and minimum values of $g(x,y)$within this domain.

d) Explain: How do the 2D plots relate to the main 3D plot of f (x, y)? Are the maximum and minimum values what you would expect?

e) Provide a brief description about why your plots make sense

f) Provide a single sentence description, in your own words, of what the function "meshgrid" does.

2) **Moment Calculation:** *Consider the situation shown in the figure to the right. Plot the variation of the moment about the base point O caused by the force F as a function of length a and angle* θ*. It is known that* $F = 500\ N$, $b = 5\ m$ *and* $a \leq 10\ m$*.*



$$M_O(a,\theta) = 500(a\sin\theta - 5\cos\theta)$$

a) Use a 3D plot to visualize how this moment at O changes with respect to a and θ.

b) Generate three 2D plots that represent the following cases

i. $M_O\,(a\,,0^{\circ})$

ii. $M_O\,(5\text{ m},\theta)$

iii. $M_O\,(10\text{ m},\theta)$

c) Use the `min` and `max` commands to find the maximum and minimum values of $M_O(a,\theta)$.

d) BONUS (Not required): Is it possible for $M_O = 0$ for any values of a and θ? If so, identify all such cases?

e) Provide a brief description about why your plots make sense

f) Provide a single sentence description, in your own words, of what the function "meshgrid" does.