Graphs of functions of two variables

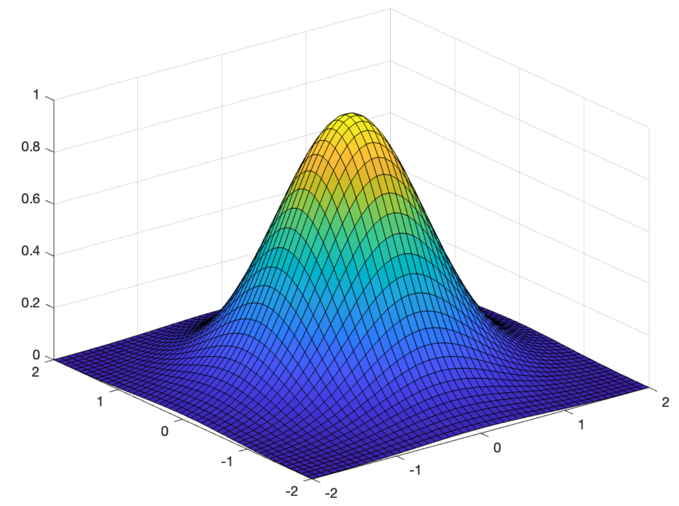
Consider the function

$$f(x, y) = 10^{-(x^2+y^2)}.$$

We'd like to graph this function over the region $-2 \leq x \leq 2$ and $-2 \leq y \leq 2$.

Here are the instructions in MATLAB needed to accomplish this.

```
>> x = linspace(-2,2,50);
>> y = linspace(-2,2,50);
>> [XX,YY] = meshgrid(x,y);
>> surf(XX,YY,exp(-(XX.^2+YY.^2)));
```



It's a fair amount of typing, but the tradeoff is extra power as we'll see below. You can always put the commands all together in a script to generate the plot, which gives you more power still to make edits and see the results of your changes.

Let's walk through these commands one by one to see what they do. The first is `linspace`, which is used to create a list of evenly spaced numbers. For example, to create a list of 5 evenly spaced numbers starting at 0 and ending at 4 we could do the following.

```
>> linspace(0,4,5)

ans =

     0     1     2     3     4
```

Similarly, to create 11 evenly spaced numbers between 0 and 1 we have:

```
>> linspace(0,1,11)

ans =

  Columns 1 through 8

         0    0.1000    0.2000    0.3000    0.4000    0.5000    0.6000    0.7000

  Columns 9 through 11

    0.8000    0.9000    1.0000
```

Notice that we are specifying the number of numbers, not the number of interval or steps. There are eleven numbers in the list with spacing of 0.1 and a total of 10 steps.

We use `linspace` here to generate a list of x and y-coordinates where we will be generating data for the plot.

```
>> x = linspace(-2,2,50);
>> y = linspace(-2,2,50);
```

The semicolons at the ends of the lines suppress output. We don't need to see all these numbers! You can think of these commands as generating the list of x-coordinates and the list of y-coordinates that we'll use to make a grid. Each of these lists is one dimensional, but a grid is two dimensional.

The `meshgrid` command generates the a grid from the lists of x- and y-coordinates. This step probably feels a little superfluous or maybe just annoying at this point. But it's the key to generating images of surfaces, even surfaces that aren't nicely represented as graphs of fuctions. Anyway, here's a tiny example of using `meshgrid`. Suppose we have set `x = [0, 1, 3]` and `y=[4 5 6]`. Let's look at what `meshgrid` does to these inputs.

```
>> [XX,YY] = meshgrid(x,y)

XX =

     0     1     3
     0     1     3
     0     1     3


YY =

     4     4     4
     5     5     5
     6     6     6
```

The result of `meshgrid` is two matrices of numbers, and in order to access both of them we need to store them in named variables. The notation `[XX,YY] = meshgrid(...)` is MATLAB's way of saying "I'm expecting to get two things back from `meshgrid`. Put the first of these in the variable `XX` and the second in `YY`". Fair enough. But what are these two matrics? The first has columns generated from the entries of `x` and the second has rows generated from the entries of `y`. Together these two matrices give all the combinations of choices of both x and y-coordinates from the original lists. For example, the first row and second column corresponds to $x = 1$ and $y = 4$. If you want, we now have something like this

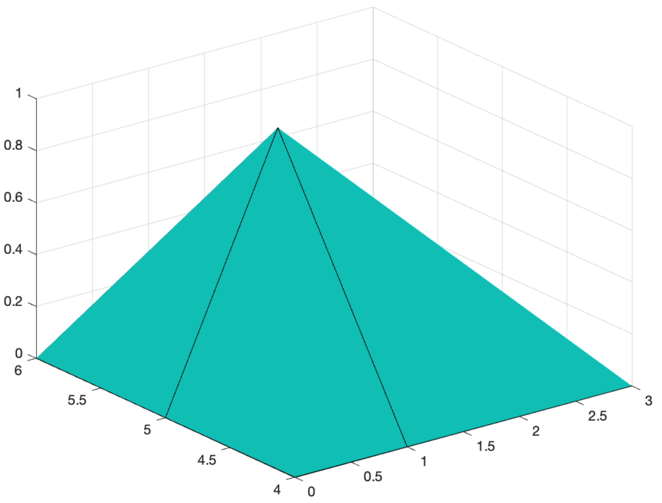
$$\begin{pmatrix} (0,4) & (1,4) & (2,4) \\ (0,5) & (1,5) & (3,5) \\ (0,6) & (1,6) & (3,6) \end{pmatrix} \tag{1}$$

which looks like a grid of coordinate pairs. The only difference is that its represented in MATLAB with two arrays of numbers rather than one array of pairs of numbers. In order to make a graph we just need z-coordinates corresponding to all these points. For example we could set

```
ZZ  = [ 0 0 0; 0 1 0; 0 0 0]
```

and make a surface plot:

```
surf(XX,YY,ZZ)
```



Once the grid of x- and y-coordinates has been generated, you can make the corresponding z-coordinates by evaluating your function at the grid locations. This was the expression `exp(-(XX.^2+YY.^2)` in the original example above. Because the variables `XX` and `YY` are matrices, you need to be sure to use dots whenever they are warranted (multiplication and exponentiation).

The graph itself is generated with a surface plot using the `surf` command which takes as its arguments three arrays of numbers. The arrays have to have the same shape (the same number of rows and columns) and describe the x-, y- and z-coordinate of a mesh of points in three dimensional space.

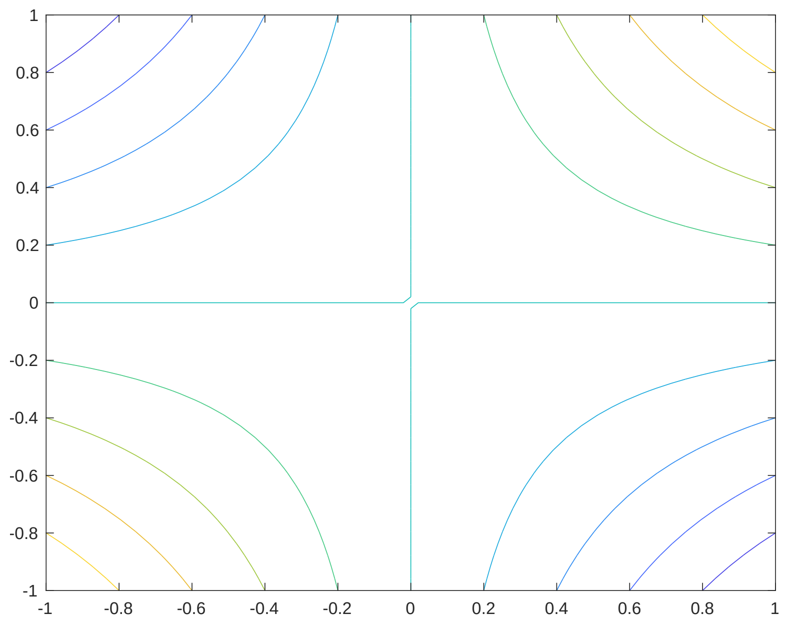
In summary, to graph a function of two variables:

1. Make lists of x- and y-coordinates that will be used to lay out a grid.
2. Generate a grid of points where the function will be sampled using `meshgrid`.
3. Compute the z-coordinates by evaluating the function at the x- and y-values created by `meshgrid`.
4. Finally, make the plot with the `surf` command.

Contour plots of functions of two variables

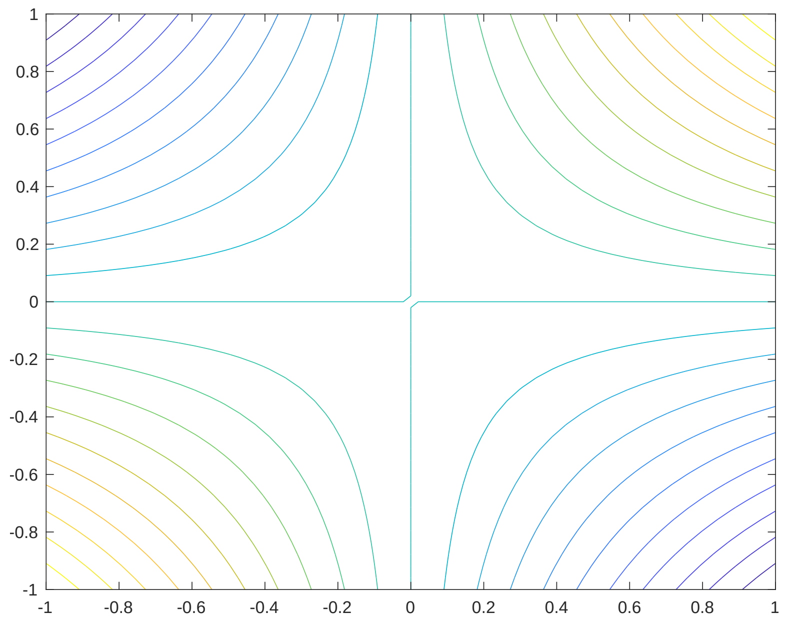
MATLAB can generate contour plots from the same data used to make surface plots. The main difference is that instead of using `surf` we use `contour`.

```
x = linspace(-1,1,50);
y = linspace(-1,1,50);
[XX,YY] = meshgrid(x,y);
contour(XX,YY, XX.*YY);
```



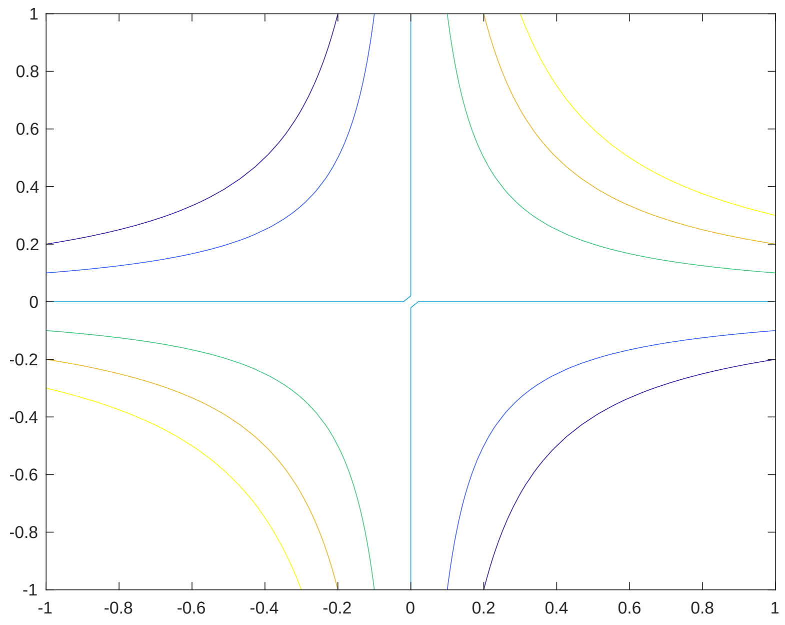
MATLAB chooses a selection of function values for the contours, but you can overide this default. The easiest control is simply to indicate the number of contours.

```
contour(XX,YY, XX.*YY, 20);
```



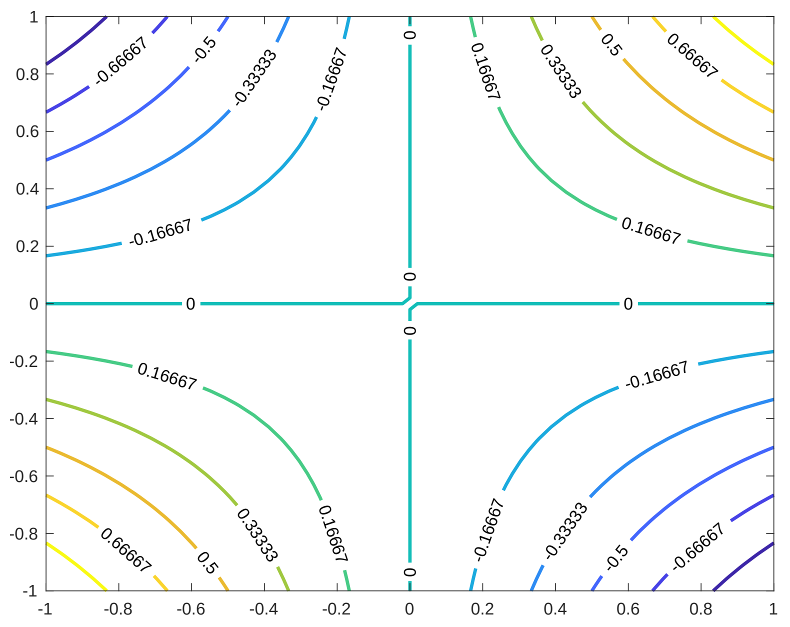
Alternatively you can list the function values where you want to see contours

```
contour(XX,YY, XX.*YY, [-0.2, -0.1, 0, 0.1, 0.2, 0.3]);
```



There are a number of options that can be specified to adjust the appearance of the plot. The following example has thicker lines and labels the function value along each contour.

```
contour(XX,YY, XX.*YY, 11, 'LineWidth',2,'ShowText','on');
```


-0.66667
-0.5
-0.33333
-0.16667
0
0.16667
0.33333
0.5
0.66667
-1
-0.8
-0.6
-0.4
-0.2
0.2
0.4
0.6
0.8
1