KU LEUVEN

MATLAB

Visualization 3D

3D visualisation

1. prepare your data	`z = peaks(20);`
2. select window and position plot region	`figure(1);` `subplot(1, 1, 1);`
3. call 3D graphing function	`h = surf(z)`

4. set colormap and shading algorithm	`colormap hot;`
	`shading interp;`
	`set(h, 'EdgeColor', 'k');`

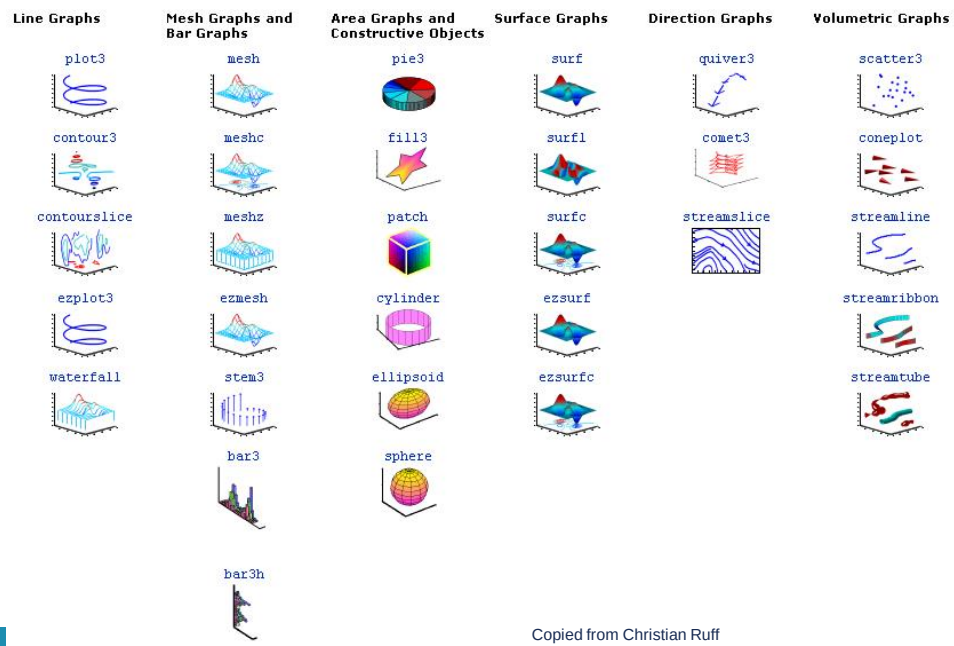
5. add lighting	`light('Position',[-2, 2, 20]);`
	`lighting phong material ([0.4,0.6,0.5,30])`
	`set (h, 'FaceColor', [0.7 0.7 0],'BackFaceLighting','lit')`

3D visualisation

6. set viewpoint	`view([30, 25]);` `set(gca, 'CameraViewAngleMode','Manual');`
7. set axis limits and tick marks	`axis([0 20 0 20 -10 10]);` `set (gca, 'Zticklabel', 'Negative || Positive')`
8. set aspect ratio	`set(gca, 'PlotBoxAspectRatio',[2.5 2.5 1]);`

9. annotate the graph	`xlabel('X Axis');`
	`ylabel('Y Axis');`
	`zlabel(' Function');`
	`title ('Peaks');`

file: first_plot3D.m



Copied from Christian Ruff

Domain generation: meshgrid

- 3D-plots: z=f(x,y)
- a surface is defined by the *z*-coordinates of points above a rectangular grid in the *x*-*y* plane.
- formed by joining adjacent points with straight lines.
- The first step in displaying a function of two variables, z = f(x,y), is to generate X and Y matrices consisting of repeated rows and columns, respectively, over the domain of the function.
- The **`meshgrid`** function transforms the domain specified by two vectors, x and y, into matrices, X and Y

meshgrid

- Compute z for each pair (x, y) by writing down a loop
- **`meshgrid`** provides the appropriate matrix
  - create a grid of uniformly sampled data points
-
```
X =[ -1 0 1]
y = [9 10 11 13]
[X, Y] = meshgrid(x, y)
```

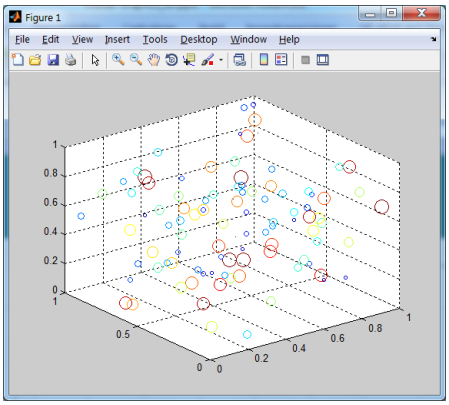
```
X =                          Y=
- 1   0   1                   9     9     9
- 1   0   1                  10    10    10
- 1   0   1                  11    11    11
- 1   0   1                  13    13    13
```

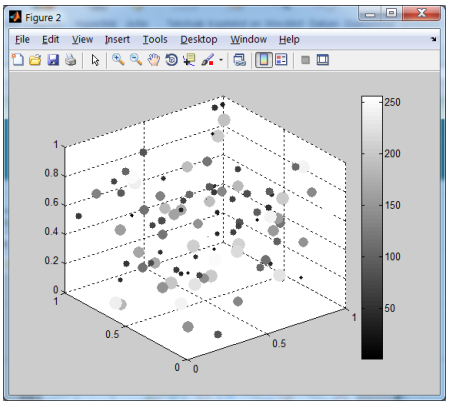
- *file: plot3D_meshgrid.m*

scatter3

- `scatter3(X,Y,Z,S,C)`
  - Will produce a scatter plot with marker size **S** and color **C**
  - **C** can be a vector of values
  - MATLAB uses current colormap and **C** to determine color of each marker
  - **C** needs to be the same length as **X, Y** if it is a vector
- Changing the size S and the color C can be used to draw 4D, 5D graphs
- *plot3D_scatter3.m*

scatter3



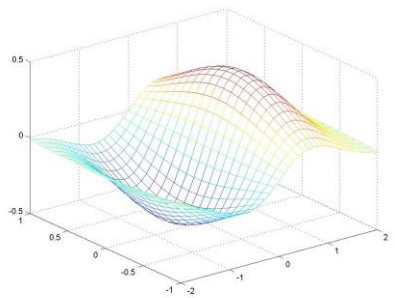


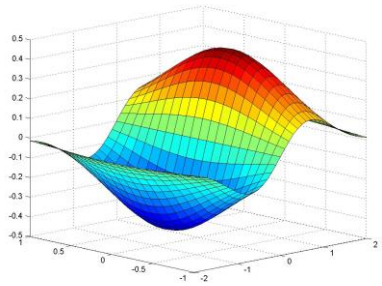
mesh / surf

- The **`mesh`** and **`surf`** commands create 3D surface plots of matrix data.
  Z(i,j) define the height of a surface over an underlying (i,j) grid, then:
  - **`mesh(Z)`** generates a colored, wire-frame view of the surface and displays it in a 3-D view.
  - **`surf(Z)`** generates a colored, faceted view of the surface and displays it in a 3-D view
    - the facets are quadrilaterals, each a constant color, outlined with black mesh lines. The shading command allows you to eliminate the mesh lines (**`shading flat`**) or to select interpolated shading across the facet (**`shading interp`**).
    - Lighting can add more realism.

mesh / surf

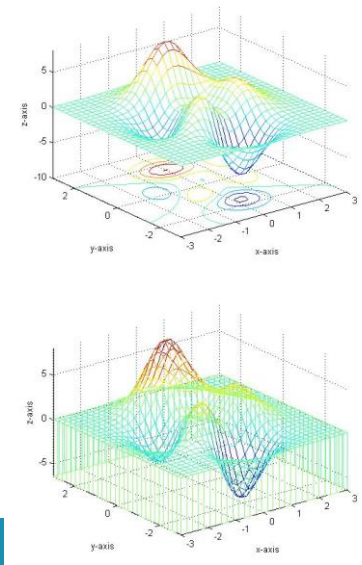
- **`mesh(X,Y,Z)`** draws a wireframe mesh with color determined by Z, so color is proportional to surface height.
- **`surf(X,Y,Z)`** draws a 3-D shaded surface plot





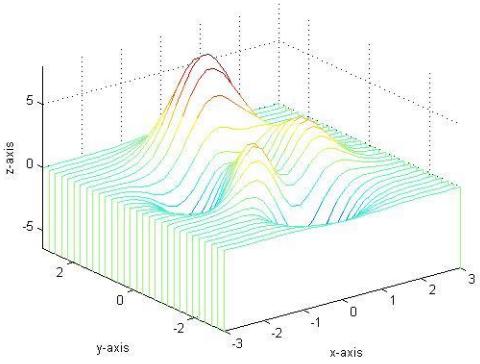
meshc / meshz

- `meshc` adds a contour plot directly below the mesh
  - helps visualize the contours
  - can locate the peaks and dips
- `meshz`: allows to emphasize the zero plane in the mesh plot
- *file: plot3D_meshc.m*



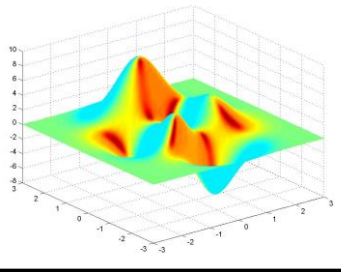
waterfall

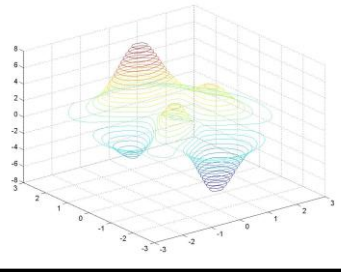
- variation on the mesh plot and can also be useful in some special cases



surfl / contour3

- **`surfl`** displays a shaded surface based on a combination of ambient, diffuse, and specular lighting models.
- **`surfc`** acts much like meshc with a contour plot drawn below the surface
- **`contour3`** creates a three-dimensional contour plot of a surface defined on a rectangular grid.





lighting

- **Flat** lighting produces uniform color across each of the faces of the object. Select this method to view faceted objects.
- **Gouraud** lighting calculates the colors at the vertices and then interpolates colors across the faces. Select this method to view curved surfaces.
- **Phong** lighting interpolates the vertex normals across each face and calculates the reflectance at each pixel. Select this choice to view curved surfaces.
  Phong lighting generally produces better results than Gouraud lighting, but takes longer to render.
- *file: plot3D_lighting.m*

More…

- Check MathWorks blogs
- http://blogs.mathworks.com/videos/2009/10/23/basics-volume-visualization-19-defining-scalar-and-vector-fields/
- Check the documentation