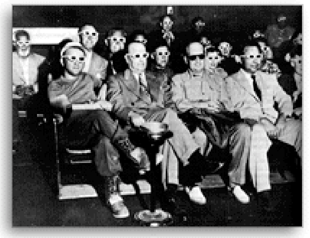
2D Routines in 3D

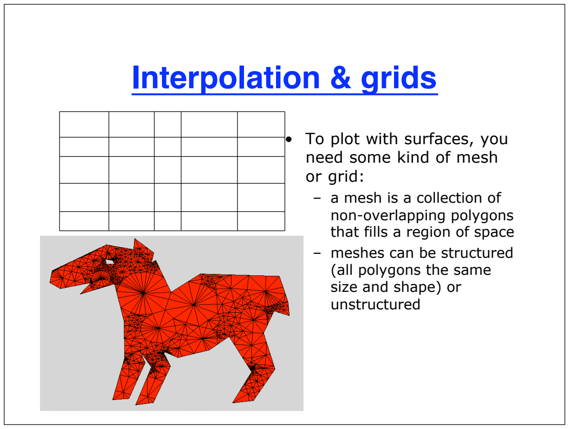


Outline

- Announcements
  - HW II--due Friday. 5PM
- HW1 & Cookie
- Grids & Meshes
- Representing f(x,y)
- Lines & Surfaces in 3D
- Survey

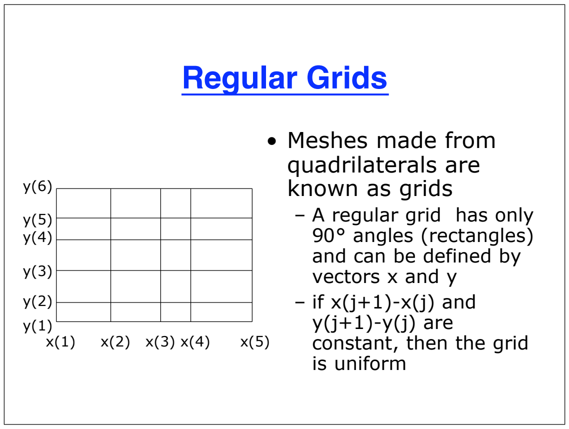
HW I

- No issues on the programs--most did well
  - sample solutions are on the web
  - Graphics functions should return handles!
- No problems figuring out colors or finding handles
  - if you don't understand a question, come find me!
- Problem 1
  - This was a bit of a trick question, but …
  - since you have to go to the computer to do the programming, you might as well try the problems
  - h=plot([1,2;1,2;1,2]',[3,4;3,2;2,4]','ro');
  - whos h will tell you h is length 3
  - Two additional objects: figure and axes



Interpolation & grids

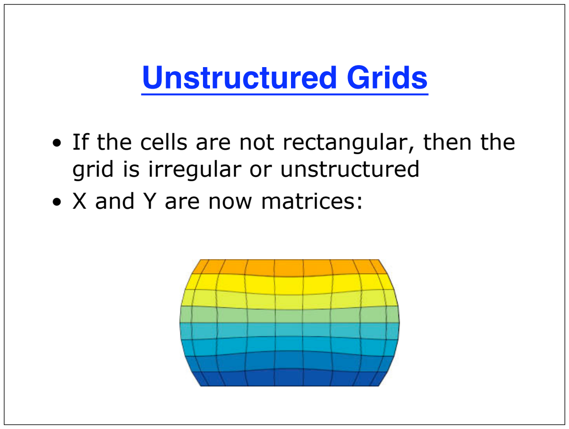
- To plot with surfaces, you need some kind of mesh or grid:
  - a mesh is a collection of non-overlapping polygons that fills a region of space
  - meshes can be structured (all polygons the same size and shape) or unstructured



Regular Grids

y(6)
y(5)
y(4)
y(3)
y(2)
y(1)
x(1)
x(2)
x(3)
x(4)
x(5)

- Meshes made from quadrilaterals are known as grids
  - A regular grid has only 90° angles (rectangles) and can be defined by vectors x and y
  - if x(j+1)-x(j) and y(j+1)-y(j) are constant, then the grid is uniform



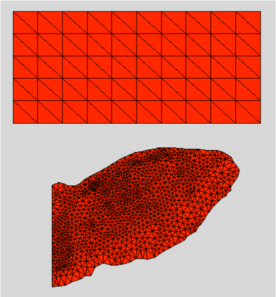
Unstructured Grids

- If the cells are not rectangular, then the grid is irregular or unstructured
- X and Y are now matrices:

Visualizing Grids

- Matlab’s surface-based functions want grids:
  - pcolor
  - contour
  - surf
  - mesh

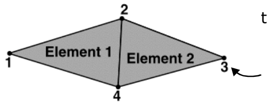
The World is not Square



- Meshes of triangles are common, especially in finite element modlling
- Triangular meshes can also be structured or unstructured
  - unstructured are more common

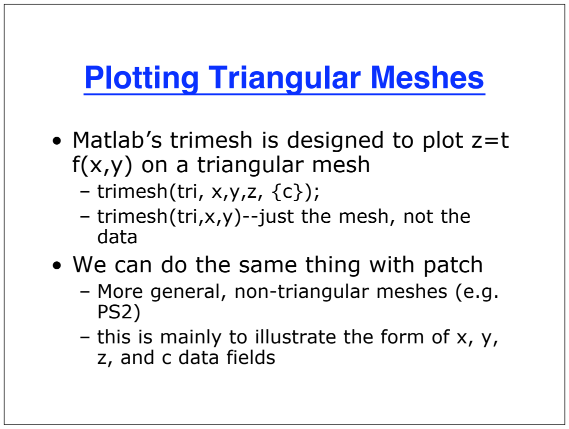
Triangular Meshes

- Matrices are rectangular, so it is hard to “fit” a triangular mesh into a matrix
- Typically, triangular meshes require 3 arrays:
  - vectors x and y contain the location of the vectors (in no particular order)
  - array tri defines how the vertices are connected
    - Each row contains indexes the three vertices forming a triangle



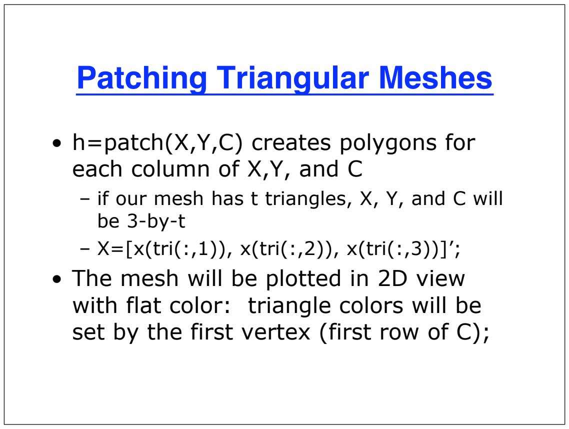
```
tri=[1 4 2;
     2 4 3];
```

(x(3),y(3))



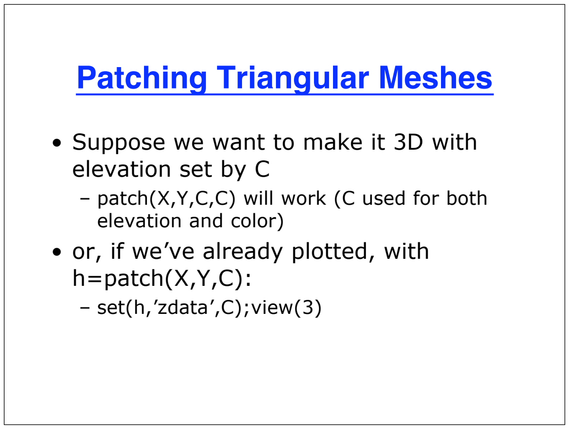
Plotting Triangular Meshes

- Matlab’s trimesh is designed to plot z=t f(x,y) on a triangular mesh
  - trimesh(tri, x,y,z, {c});
  - trimesh(tri,x,y)--just the mesh, not the data
- We can do the same thing with patch
  - More general, non-triangular meshes (e.g. PS2)
  - this is mainly to illustrate the form of x, y, z, and c data fields



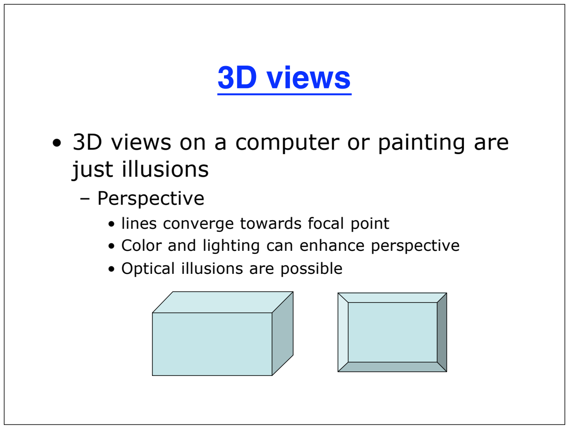
Patching Triangular Meshes

- h=patch(X,Y,C) creates polygons for each column of X,Y, and C
  - if our mesh has t triangles, X, Y, and C will be 3-by-t
  - X=[x(tri(:,1)), x(tri(:,2)), x(tri(:,3))]’;
- The mesh will be plotted in 2D view with flat color: triangle colors will be set by the first vertex (first row of C);



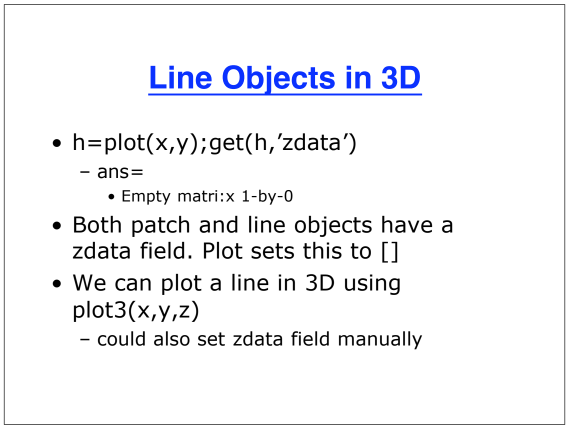
Patching Triangular Meshes

- Suppose we want to make it 3D with elevation set by C
  - patch(X,Y,C,C) will work (C used for both elevation and color)
- or, if we’ve already plotted, with h=patch(X,Y,C):
  - set(h,’zdata’,C);view(3)



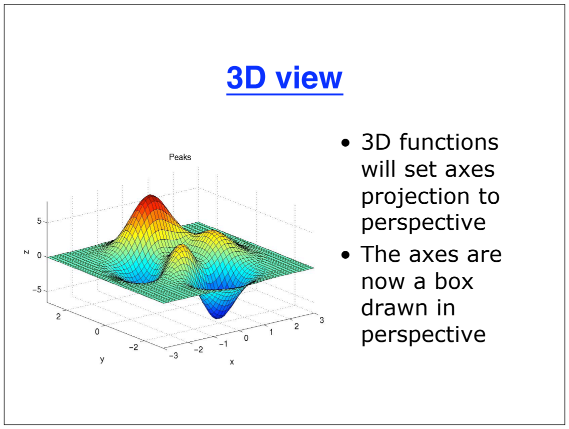
3D views

- 3D views on a computer or painting are just illusions
  - Perspective
    - lines converge towards focal point
    - Color and lighting can enhance perspective
    - Optical illusions are possible



Line Objects in 3D

- h=plot(x,y);get(h,’zdata’)
  - ans=
    - Empty matri:x 1-by-0
- Both patch and line objects have a zdata field. Plot sets this to []
- We can plot a line in 3D using plot3(x,y,z)
  - could also set zdata field manually



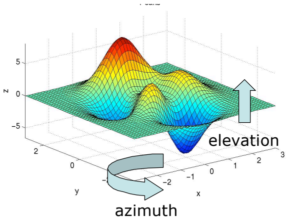
3D view

- 3D functions will set axes projection to perspective
- The axes are now a box drawn in perspective

Controlling the 3D view

- We can control the size of the axes (limits) and the way they are drawn (view)
  - set(gca,’xlim’,[minimum, maximum])--also for y and z
  - Can also set scale to log or reverse direction (must be done manually)
- Clicking on the circle button allows you to rotate the axes in 3D

Controlling the 3D view



- Can also control the view from the command line through view:
  - view(2) or view (3) gets default 2D or 3D views
  - view([az,el]) sets the azimuth=az (rotates about z) and elevation=el(rotates about line in x-y- plane)

Surfaces in 3D

- Like lines, patch and surface objects have zdata fields.
- surf(X,Y,Z) creates a surface with vertices defined by X,Y, and Z
  - color is proportional to Z
  - facecolor=flat
- mesh(X,Y,Z) is similar, but doesn’t fill polygons
  - edgecolor=flat

Comparing surf and pcolor

- pcolor is a special form of surf

field	pcolor(x,y,Z)	surf(x,y,Z)
xdata	x	x
ydata	y	y
zdata	*0*Z*	*Z*
cdata	*Z*	*Z*
facecolor	‘faceted’	‘faceted’
projection	*orthographic*	*perspective*
View	*[0 90]*	*[-37.5 30]*

- How can we change cdata?