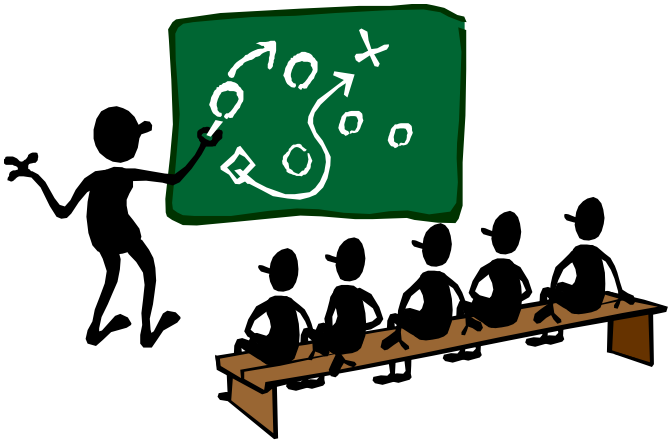
CSC 470 COMPUTER GRAPHICS

Fitting In

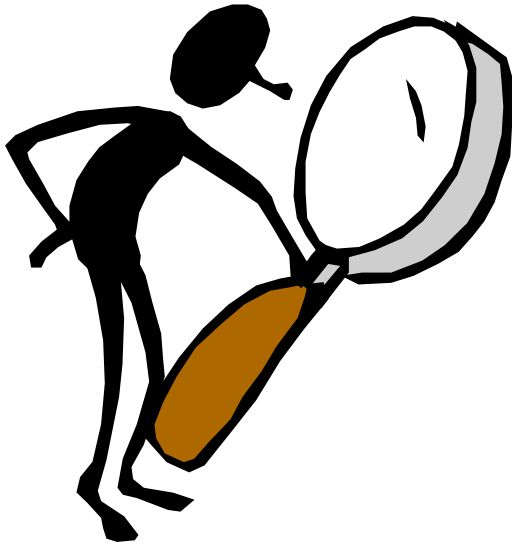
Fitting it in

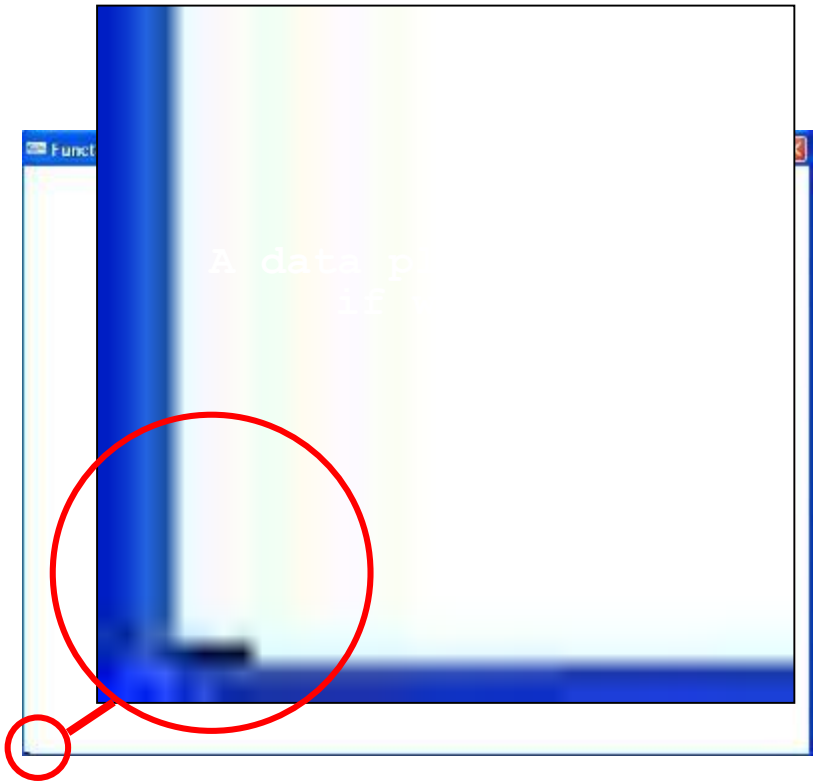
- Sometimes you may want to print out a plot of some data when the data ranges are unknown.
- However, you want them to appear in the window presented in a visually pleasing manner.



Fitting it in

- E.g. Let's plot $e^{-x}\cos(2\pi x)$
- Plotted with screen coordinates of 640 x 480 we will get:



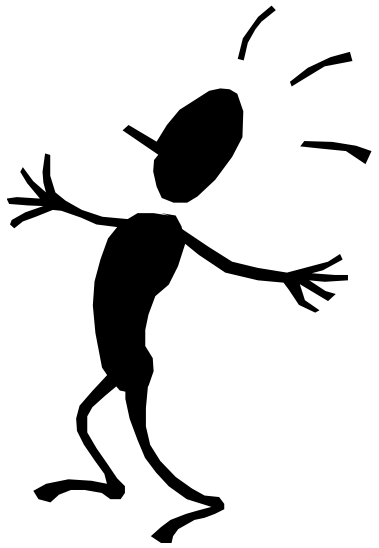


Fitting it in

How do we MAGNIFY the data?

Stretch out X

Stretch out Y



Simple “Dot Plots”

- Draw a function $f(x) = e^{-x} \cos(2\pi x)$

  for values of *x* between 0 and 4?

Steps:

- “sample” it at a collection of equispaced *x*-values;
- plot a dot at each coordinate pair (x_i, $f(x_i)$);
- choose some suitable increment, say 0.005, between consecutive *x*-values.

Simple “Dot Plots” cont’d

```
glBegin(GL_POINTS);
   for(GLdouble x = 0; x < 4.0 ; x += 0.005)
                glVertex2d(x, f(x));
glEnd();
glFlush();
```

Problems:

- The picture produced will be impossibly tiny because values of *x* between 0 and 4 map to the first four pixels at the bottom left of the screen window.
- Further, the negative values of *f*(.) will lie below the window and will not be seen at all.
- We therefore need to scale and position the values to be plotted so they cover the screen window area appropriately.

Fitting it in

- The X coordinates of the window range over 640 values (from 0 to 639).
- The X values for the data range from 0 to 4.
- We need to modify the data values so that data point 0 maps to window coordinate 0 and data point 4 maps to window coordinate 640.

Fitting it in

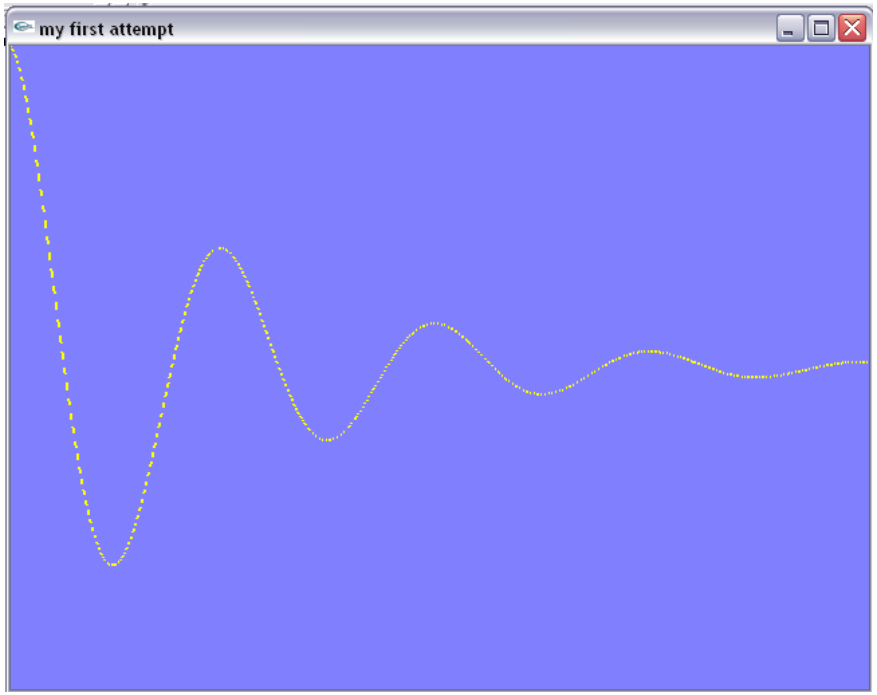
- In essence we want 4 to be plotted at 640.
- 4 * A = 640;
- A = 160; ….. or for all cases
  - A = SCREENWIDTH/x_{max}
- Therefore, x’ = x * A
  - …. if x ranges from 0 .. x_{max}

Fitting it in

- If x ranges from say 1 to 4 then we will want to stretch a range of 3 out instead of 4, therefore:
  - $A = SCREENWIDTH/(x_{max} - x_{min})$
- The same applies for the y coordinates:
  - $C = SCREENHEIGHT/(y_{max} - y_{min})$

Dot Plots: scaling and shifting

Scaling x: `sx = x * screenWidth /4.0;`



Scaling and shifting y: `sy = (y + 1.0) * screenHeight / 2.0;`

Fitting In

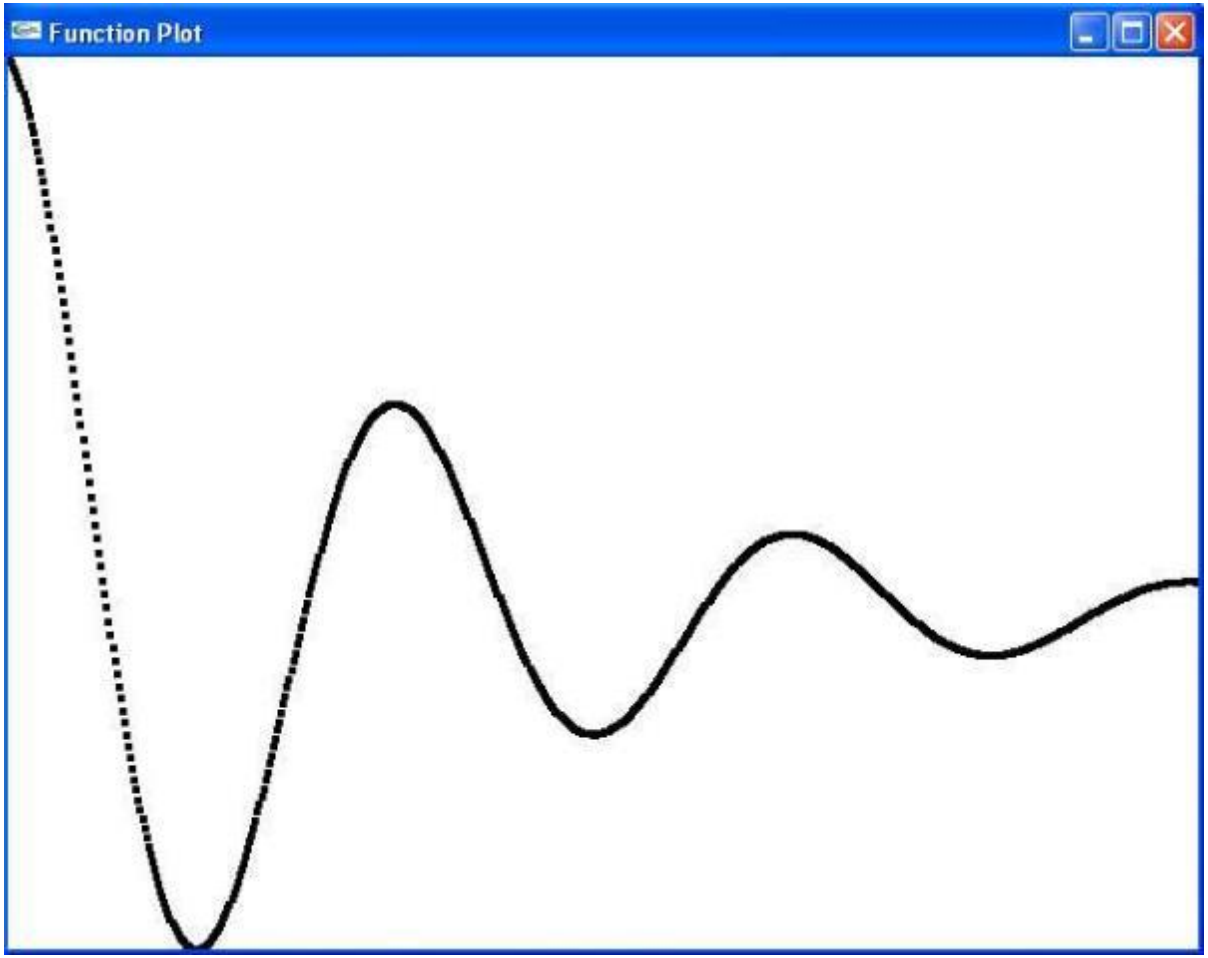
$$x' = Ax + B \text{ and } y' = Cy + D$$

```
A = SCREENWIDTH/(xmax-xmin)
B = -xmin*A
C = SCREENHEIGHT/(ymax-ymin)
D = |ymin|*B
```

A and C are *scalars*

B and D are *translators*

Moving it around



Setting Values for A, B, C and D

- The conversions from ***x*** to ***sx***, and from ***y*** to ***sy***, have the form:

$$sx = A * x + B$$

$$sy = C * y + D$$

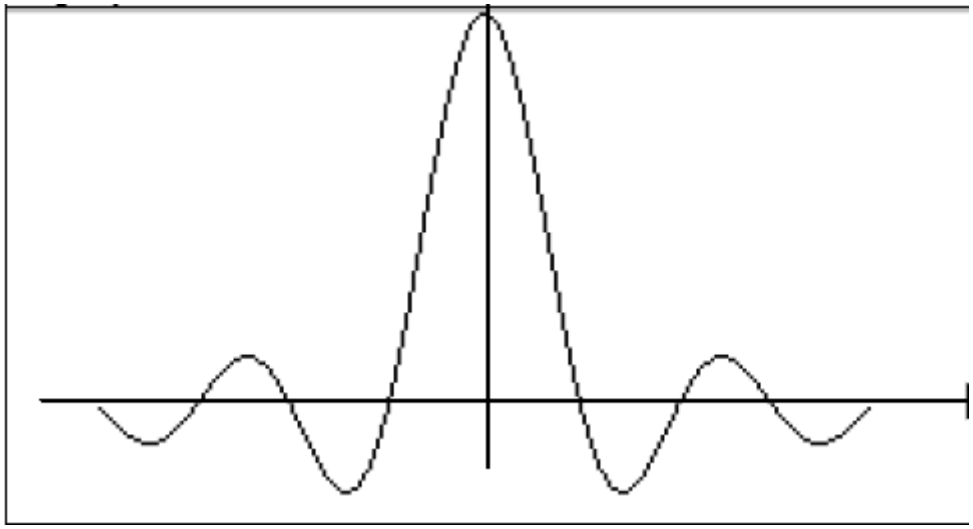
where

```
GLdouble A, B, C, D, x;
A = screenWidth / 4.0;
B = 0.0;
C = screenHeight / 2.0;
D = C;
glBegin(GL_POINTS);
for(x = 0; x < 4.0 ; x += 0.005)
    glVertex2d(A * x + B, C * f(x) + D);
glEnd();
glFlush();
```

```
#include <windows.h> // use proper includes for your system
#include <math.h>
#include <gl/Gl.h>
#include <gl/glut.h>
const int screenWidth = 640;      // width of screen window in pixels
const int screenHeight = 480;     // height of screen window in pixels
GLdouble A, B, C, D;  // values used for scaling and shifting
//<<<<<<<<<<<<<<<<<<<<<<< myInit >>>>>>>>>>>>>>>>>>>>>>>
 void myInit(void)
 {
    glClearColor(1.0,1.0,1.0,0.0);        // background color is white
    glColor3f(0.0f, 0.0f, 0.0f);          // drawing color is black
    glPointSize(2.0);                    // a 'dot' is 2 by 2 pixels
    glMatrixMode(GL_PROJECTION);        // set "camera shape"
    glLoadIdentity();
    gluOrtho2D(0.0, (GLdouble)screenWidth, 0.0, (GLdouble)screenHeight);
    A = screenWidth / 4.0; // set values used for scaling and shifting
    B = 0.0;
    C = D = screenHeight / 2.0;
}
//<<<<<<<<<<<<<<<<<<<<<<< myDisplay >>>>>>>>>>>>>>>>>>>>>>>
void myDisplay(void)
{
  glClear(GL_COLOR_BUFFER_BIT);      // clear the screen
  glBegin(GL_POINTS);
  for(GLdouble x = 0; x < 4.0 ; x += 0.005)
  {
     Gldouble func = exp(-x) * cos(2 * 3.14159265 * x);
     glVertex2d(A * x + B, C * func + D);
   }
  glEnd();
  glFlush();          // send all output to display
}
//<<<<<<<<<<<<<<<<<<<<<<< main >>>>>>>>>>>>>>>>>>>>>>>
void main(int argc, char** argv)
{
  glutInit(&argc, argv);            // initialize the toolkit
  glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB); // set display mode
  glutInitWindowSize(screenWidth, screenHeight); // set window size
```

Example

$$\mathrm{sinc}(x) = \frac{\sin(\pi x)}{\pi x}$$



```
void myDisplay(void)
{glBegin(GL_LINE_STRIP);
        for(GLfloat x = -4.0; x < 4.0; x += 0.1)
        {
                GLfloat y = sin(3.14159 * x) / (3.14159 * x);
                glVertex2f(x, y);
        }
glEnd();
glFlush();}
```