Peter Woodman
0127119
Amath 352 - B.Bale
Homework 6
"Happy Birthday Me" edition

1. (a) Because this gives us the top half of the ellipse- the part that does not intersect with the circle at all.

   (b) I created the following matlab function:

```
function f = tmp(x);
f = (x+1).^2 + (1 - sqrt(1 - 0.25.*x.^2)).^2 - .5;
```

   Then I entered the following commands:

```
>> fzero(@tmp,0)

ans =

  -2.9298e-01

>> fzero(@tmp,-1)

ans =

  -1.5883e+00
```

   This means that $x1 = -1.5883$ or -0.2930. Substituting back in for $x2 = -\sqrt{1 - 0.25x_1^2}$, we find the solutions to be

$$x_1 = -1.5833 \qquad x_2 = -0.6077$$
$$x_1 = -0.2930 \qquad x_2 = -0.9892$$

   (c)

$$f(x) = \begin{bmatrix} x_1^2 + 4x_2^2 - 4 \\ (x_1+1)^2 + (x_2+1)^2 - 0.5 \end{bmatrix}$$
$$J(x) = \begin{bmatrix} 2x_1 & 8x_2 \\ 2x_1 + 2 & 2x_2 + 2 \end{bmatrix}$$

   (d)

$$x_0 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad f(x_0) = \begin{bmatrix} -3 \\ 4.5 \end{bmatrix}$$

   A step of Newton's method for a system of more than one variable looks as follows-

$$x_n = x_{n-1} + \Delta x_{n-1}$$

   where

$$J_n \Delta_n = -f_n$$

   So first, we find J at x_0

$$J(x) = \begin{bmatrix} 2x_1 & 8x_2 \\ 2x_1 + 2 & 2x_2 + 2 \end{bmatrix} \qquad J(x_0) = \begin{bmatrix} 2 & 0 \\ 4 & 2 \end{bmatrix}$$

We then solve this system to find Δx-

$$\Delta x = \begin{bmatrix} -1.5 \\ 5.25 \end{bmatrix}$$

From here, we can compute x_1-

$$x_1 = x_0 - \Delta x_0 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} -3/2 \\ 15/2 \end{bmatrix} = \begin{bmatrix} 5/2 \\ 15/2 \end{bmatrix}$$

(e) Code is as follows:

```
function r = newtonsys(x0,maxit)
x = x0;    % initial guess
k = 0;     % initialize the iteration number
while k < maxit
  k = k+1;
  f = [x(1)^2 + 4*x(2)^2-4; (x(1)+1)^2+(x(2)+1)^2-0.5];
  J = [2*x(1) 8*x(2); 2*x(1)+2 2*x(2)+2];
  dx = J\f;
  x = x - dx;
end
r = x;
```

And the output:

```
>> newtonsys([-1 0]',8)

ans =

   -1.5883
   -0.6077
```

(f)
```
>> newtonsys([1 0]',8)

ans =

   -0.2930
   -0.9892
```

(g) In this case, the Jacobian evaluated at the initial guess is singular, meaning we cannot find a solution to $J_n \Delta x_n = -f_n$.

2. (a)

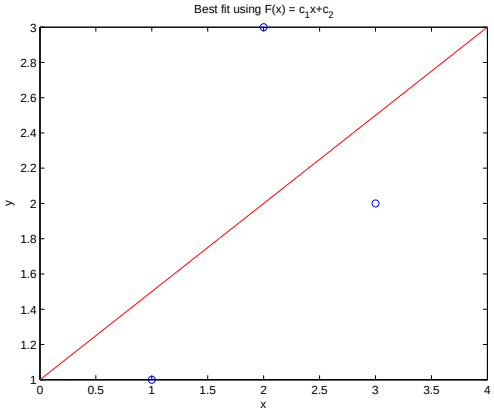
$$F(x) = c_1 x_1 + c_2 \tag{1}$$

$$Ac = y \qquad \begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 3 & 1 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \\ 2 \end{bmatrix} \tag{2}$$

$$(A^T A)c = (A^T y) \qquad \begin{bmatrix} 14 & 6 \\ 6 & 3 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} 13 \\ 6 \end{bmatrix} \tag{3}$$

```
>> x = [1 2 3]'; y = [1 3 2]';
>> A = [x ones(size(x))];
>> c = (A'*A)\(A'*y)

c =

    0.5000
```

```
    1.0000

>> c = A\y

c =

    0.5000
    1.0000
```



This doesn't really look the best, but it does seem to look like about the best one can get by drawing a straight line. Looks to be around the average value.

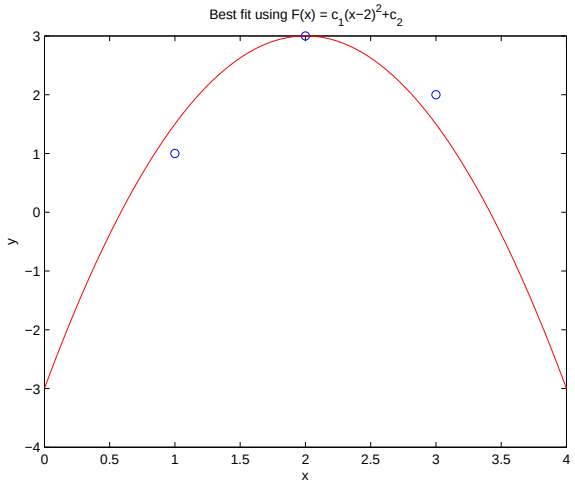
(b)

$$F(x) = c_1(x-2)^2 + c_2$$

$$Ac = y \qquad \begin{bmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \\ 2 \end{bmatrix}$$

$$(A^T A)c = (A^T y) \qquad \begin{bmatrix} 2 & 2 \\ 2 & 3 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \\ 2 \end{bmatrix}$$

```
>> x = [1 0 1]'; y = [1 3 2]';
>> A = [x ones(size(x))];
>> c = (A'*A)\(A'*y)

c =

   -1.5000
    3.0000

>> c = A\y

c =

   -1.5000
    3.0000
```



Aha! This looks much better!

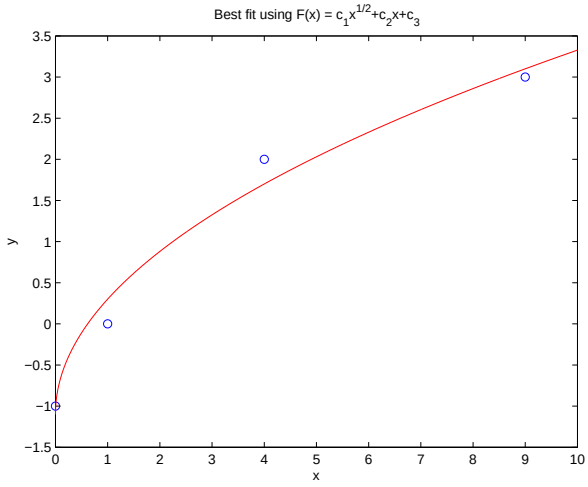
(c)

$$F(x) = c_1\sqrt{x} + c_2 x + c_3$$

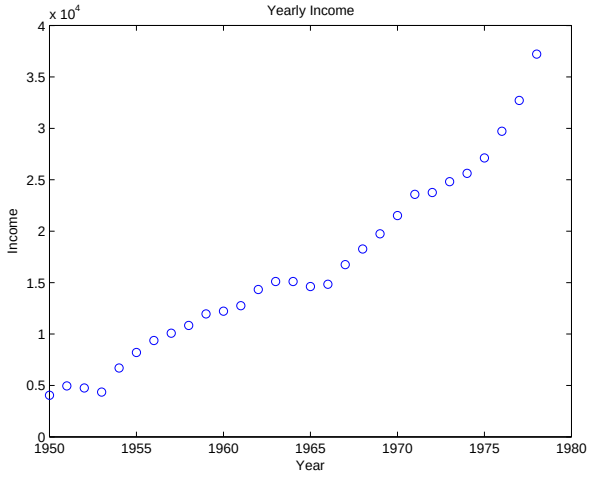
$$Ac = y \qquad \begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \\ 2 & 4 & 1 \\ 3 & 9 & 1 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \\ 2 \\ 3 \end{bmatrix}$$

$$(A^T A)c = (A^T y) \qquad \begin{bmatrix} 14 & 36 & 6 \\ 36 & 98 & 14 \\ 6 & 14 & 4 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \\ 2 \\ 3 \end{bmatrix}$$

```
>> x = [0 1 4 9]'; y = [-1 0 2 3]';
>> A = [sqrt(x) x ones(size(x))];
>> c = (A'*A)\(A'*y)

c =

    1.4000
   -0.0000
   -1.1000

>> c = A\y

c =

    1.4000
   -0.0000
   -1.1000
```



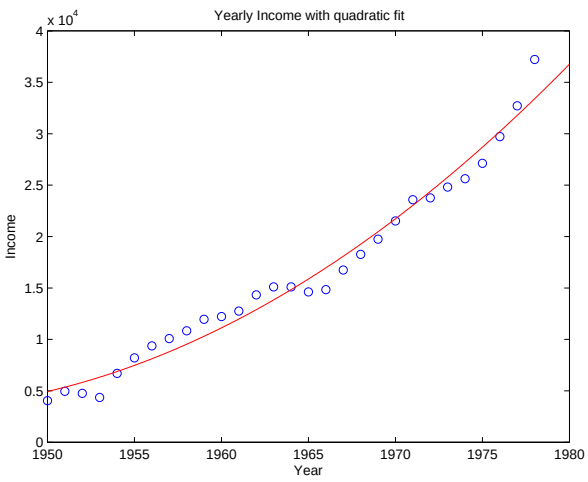
3. (a) Graph!



This data *might* be a quadratic fit, but if so, a very slight quadratic, and given that this is income data that we're talking about, I'm going to guess it's going to be best described by a linear function.

(b) Here is the graph we get from the linear fit-

According to the best fit line, a person's income in 1980 would be $32,647.
Just out of curiosity, the following is the quadratic fit-

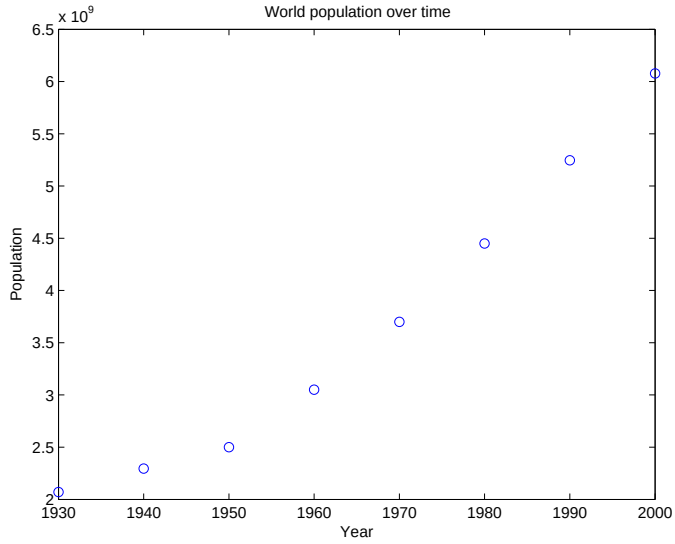


Wow. That sure does look a lot better. Let's ask matlab to compute the norm of the residuals-

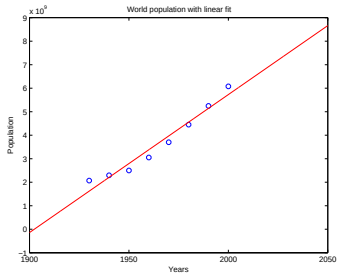
```
>> A1 = [x ones(size(x))];
>> c1 = A1\y

c1 =

   1.0e+06 *

    0.0010
   -1.9803

>> A2 = [x.^2 x ones(size(x))];
>> c2 = A2\y

c2 =

   1.0e+07 *

    0.0000
   -0.0086
    8.3290

>> fx = linspace(1950,1980,1000);
>> fy1 = c1(1).*fx + c1(2);
>> fy2 = c2(1).*fx.^2 + c2(2).*fx + c2(3);
>> r1 = y - A1*c1;
>> r2 = y - A2*c2;
>> [norm(r1,2) norm(r2,2)]

ans =

   1.0e+04 *

    1.0334    0.7171
```

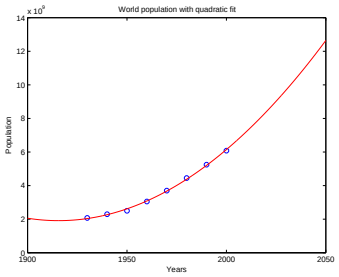
Well. That plainly shows that the quadratic fit is a significantly better fit to this data (by a factor of about .3). With this fit, our man makes $36,759 in 1980.

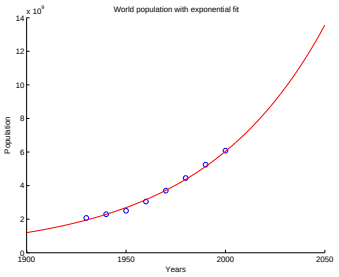
4. Here is the initial graph of the data-



Here is the data with fits-







Interestingly enough, the linear fit's figure of 8,663,000,000 people for the year 2050 is the closes number to the UN prediction of 9 billion. This sort of makes sense- in the initial plot of the data set alone, it does appear as if the last 4 data points are linear- obviously this is how the UN sees it, too. The quadratic fit predicts 12,642,000,000 people, and the pseudoexponential fit predicts 13,565,000,000 people in the year 2050, both overshooting the UN estimate by billions.

When you look at the other end of the scale, though, in 1900, the linear fit makes the nonsensical estimate of -142,000,000 people. Negative people. The quadratic fit is a better guess at 2,055,200,000 people, but if you look at the graph, you'll also note that at this point in time, the fit predicts that the poulation is *decreasing* from this number, and we can only predict that if we were to look further back in time, the population would increase. The exponential fit makes the guess of 1,196,600,000, and shows proper growth behavior. This is definitely a better fit for this end of the graph.

CODE SNIPPETS *Diary for solution of problem 4*

```
>> load worldpop.dat
>> x = worldpop(:,1);
>> y = worldpop(:,2);
>> A1 = [x ones(size(x))];
>> A2 = [x.^2 x ones(size(x))];
>> A3 = [x ones(size(x))];
>> v = log(y);
>> c1 = A1\y;
>> c2 = A2\y;
>> ct = A3\v;
>> c3 = [exp(ct(2)); ct(1)];
>> fx = linspace(1900,2050,2000);
>> fy1 = c1(1).*fx + c1(2);
>> fy2 = c2(1).*fx.^2 + c2(2).*fx + c2(3);
>> fy3 = c3(1).*exp(c3(2).*fx);
>> [fy1(1) fy2(1) fy3(1)]

ans =

   1.0e+09 *

   -0.1421    2.0552    1.1966

>> [fy1(2000) fy2(2000) fy3(2000)]

ans =

   1.0e+10 *

    0.8663    1.2642    1.3565
```