Assignment 4: Implement a Filter
Seth Kurtenbach

Description: In this assignment, I use data from A3-MeasurementData.bin to implement a filter, estimating 3D coordinates as measured by the sensor.

Deliverables:

1. First observation mean vector z and sqrt of its covariance R.

$$z1 = [\ 12.7785 \quad 130.0927 \quad 23.5293]$$

$$\text{sqrt}(R) = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1.4142 & 1.4142 \\ 1 & 1.4142 & 1.7321 \end{bmatrix}$$

However, in initializing my filter, I used the following initial input:

$$z0 = [0 \quad 0 \quad 0]$$

$$\text{sqrt}(R) = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 10 \end{bmatrix}$$

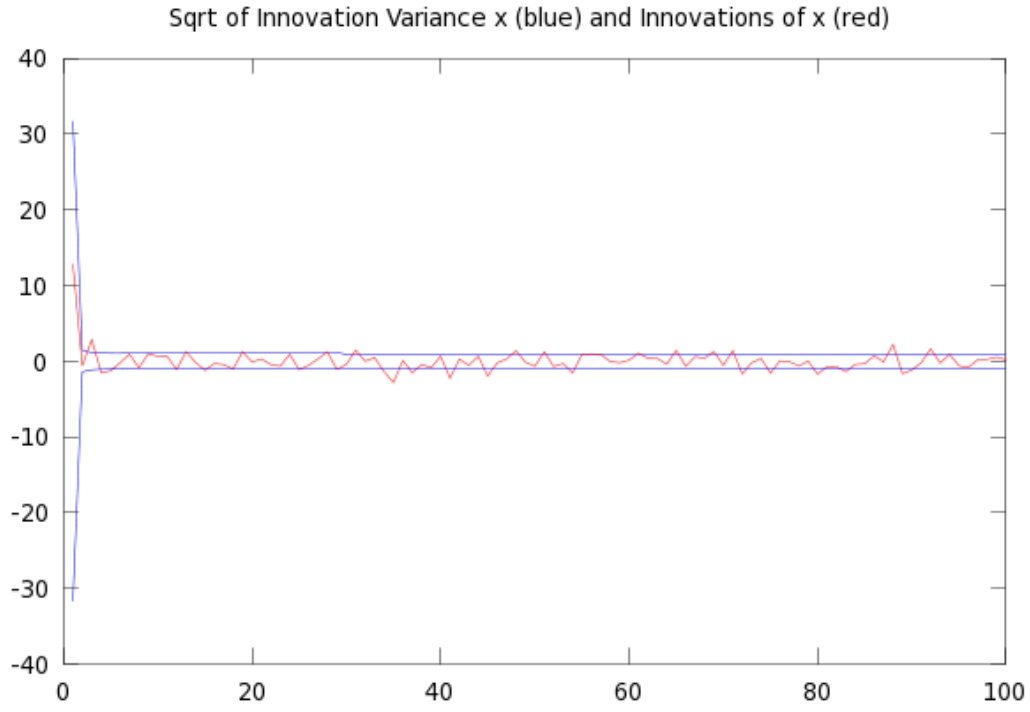
in order to represent an initial state of complete uncertainty.

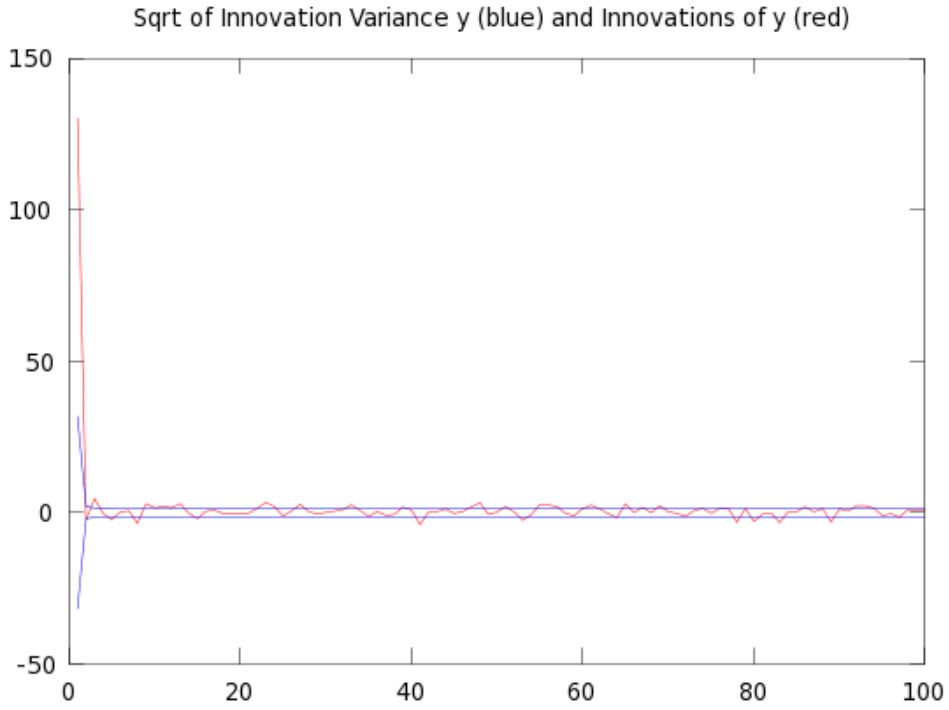
2. Final mean x and sqrt of covariance matrix P.

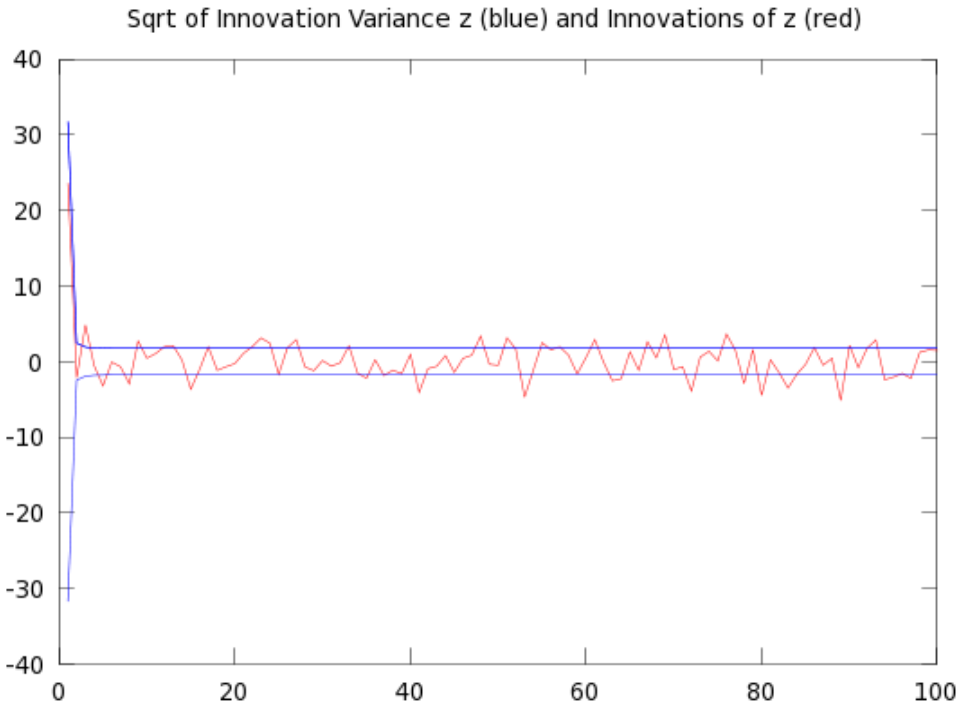
$$x = [\ 12.893 \quad 130.271 \quad 23.489\ ]$$

$$\text{sqrt}(P) = \begin{bmatrix} 0.0015811 & 0 & 0 \\ 0 & 0.0011180 & 0 \\ 0 & 0 & 0.0018257 \end{bmatrix}$$

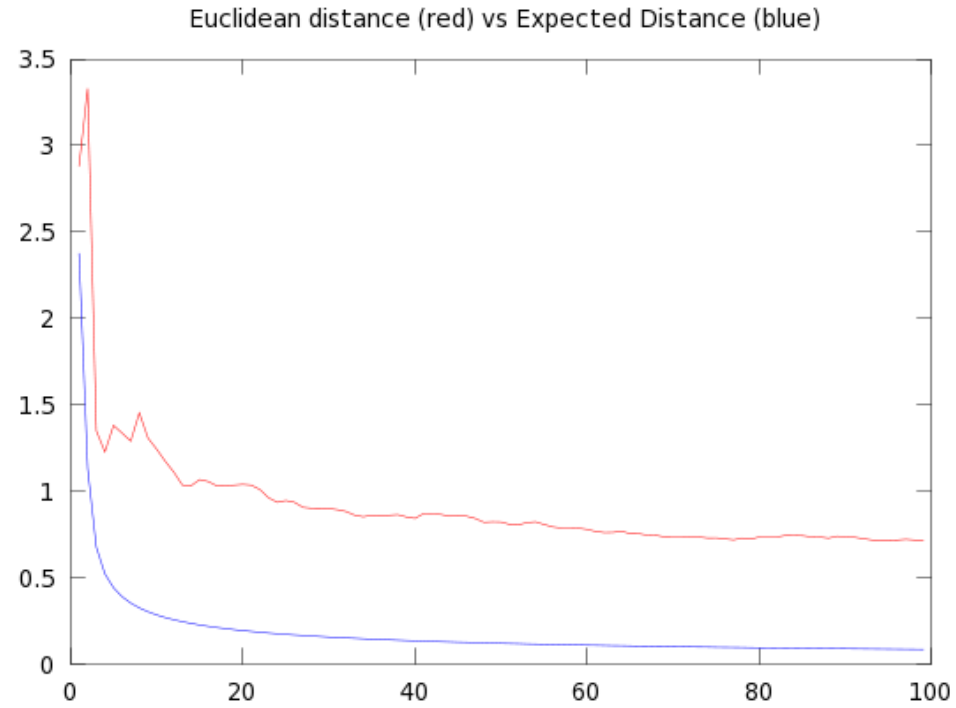
3. Three plots showing innovations in x,y,z with sqrts of respective invariances.







4. Plot of Euclidean distance vs Expected distance



Code:

```
# Assignment 4: Implement a filter to process 3D sensor observations.
# Seth Kurtenbach

actual = [12.9, 130.4, 23.5];

R = [1,1,1;
     1,2,2;
     1,2,3];

fm = fopen("A3-MeasurementData(1).bin");

meas = fread(fm, [3, 100000], "float");
meas = meas';

## Initialized Estimate (x, P) ##
x = [0,0,0;
       0,0,0;
       0,0,0];
P = [100,0,0;
        0,100,0;
        0,0,100];


for i = 1:100000

    z = [meas(i,1),meas(i,2),meas(i,3)];
    S = P .+ R;
    W = P .* inv(S);

    innovX(i) = z(1) - x(1,1);
    innVarX(i) = sqrt(S(1,1));
    innVarXmin(i) = -(innVarX(i));

    innovY(i) = z(2) - x(2,2);
    innVarY(i) = sqrt(S(2,2));
    innVarYmin(i) = - (innVarY(i));

    innovZ(i) = z(3) - x(3,3);
    innVarZ(i) = sqrt(S(3,3));
    innVarZmin(i) = - (innVarZ(i));

    newX(1) = x(1,1);
    newX(2) = x(2,2);
    newX(3) = x(3,3);

    euc(i) = norm(newX - actual, 2);
    expDist(i) = sqrt(sum(eig(P)));
```

```
    P = P .- (W .* S .* W');
    x = x .+ (W .* (z .- x));

end

finalX(1) = x(1,1);
finalX(2) = x(2,2);
finalX(3) = x(3,3);
```