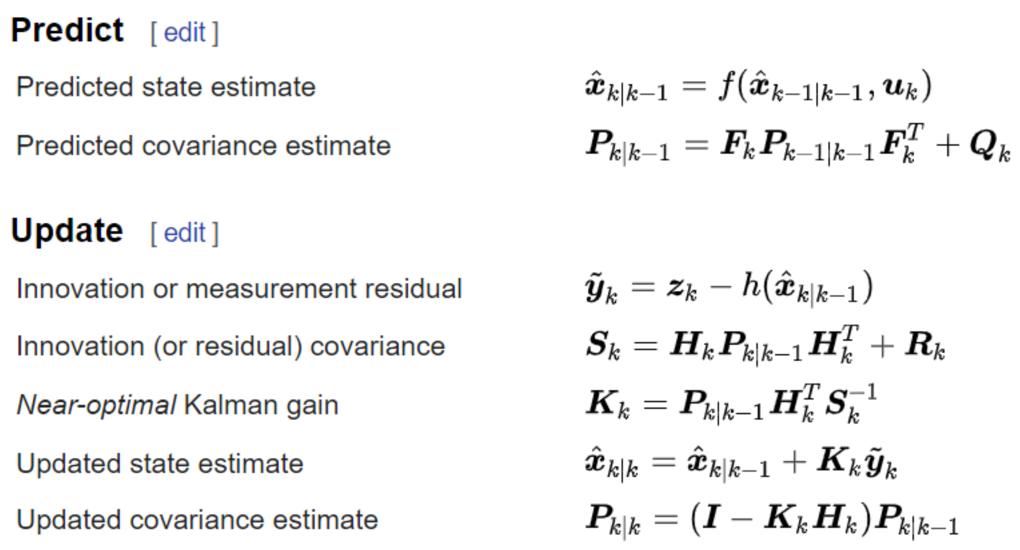
Problem 3

i) There are 6 states, and the corresponding equations are coded as shown in the code below:

```
syms ux uy phi r X Y real


F1=-dt*((p.f0+p.f1*ux+p.f2*ux^2)/(p.m))+ux;

F2= dt*(-((2*p.Cf+2*p.Cr)/(p.m*ux))*uy + (((2*p.Cr*p.lr-2*p.Cf*p.lf)/(p.m*ux))-
(ux))*r) + uy;

F3 = r*dt+phi;

F4= dt*(((2*p.Cr*p.lr-2*p.Cf*p.lf)/(p.Izz*ux))*uy -
((2*p.Cf*p.lf*p.lf+2*p.Cr*p.lr*p.lr)/(p.Izz*ux))*r)+r;

F5= dt*(ux*cos(phi) - uy*sin(phi))+X;

F6= dt*(ux*sin(phi)+uy*cos(phi))+Y;


%% The measurement matrix is shown below:

C = [1 0 0 0 0 0;... % backtrack from engine speed sensor [y1]
1 0 0 0 0 0;... % backtrack from wheel speed sensor [y2]
0 0 1 0 0 0;... % IMU for orientation [y3]
0 0 0 1 0 0;... % Yaw rate sensor [y4]
0 0 0 0 1 0;... % GPS X-pos [y5]
0 0 0 0 0 1;... % GPS Y-pos [y6]
0 0 0 0 1 0;... % Lidar X-pos [y7]
0 0 0 0 0 1;... % Lidar Y-pos [y8]
0 0 1 0 0 0]; % Lidar orientation [y9] (9x6 matrix)


% Calculting the Jacobian at initial operating point x0:

Fk_j = jacobian([F1;F2;F3;F4;F5;F6],[ux;uy;phi;r;X;Y]);

% substitute the initial operating point

Fk1 = double(subs(Fk_j,[ux uy phi r X Y],x0'));
% Checking observability at the initial operating point

rank(obsv(Fk1,C)) == 6;
```

The rank of the observability matrix is 6 which is full, hence the system is observable.

ii) Estimation of states using EKF:



Predict [edit]

Predicted state estimate	$\hat{\boldsymbol{x}}_{k\|k-1} = f(\hat{\boldsymbol{x}}_{k-1\|k-1}, \boldsymbol{u}_k)$
Predicted covariance estimate	$\boldsymbol{P}_{k\|k-1} = \boldsymbol{F}_k \boldsymbol{P}_{k-1\|k-1} \boldsymbol{F}_k^T + \boldsymbol{Q}_k$

Update [edit]

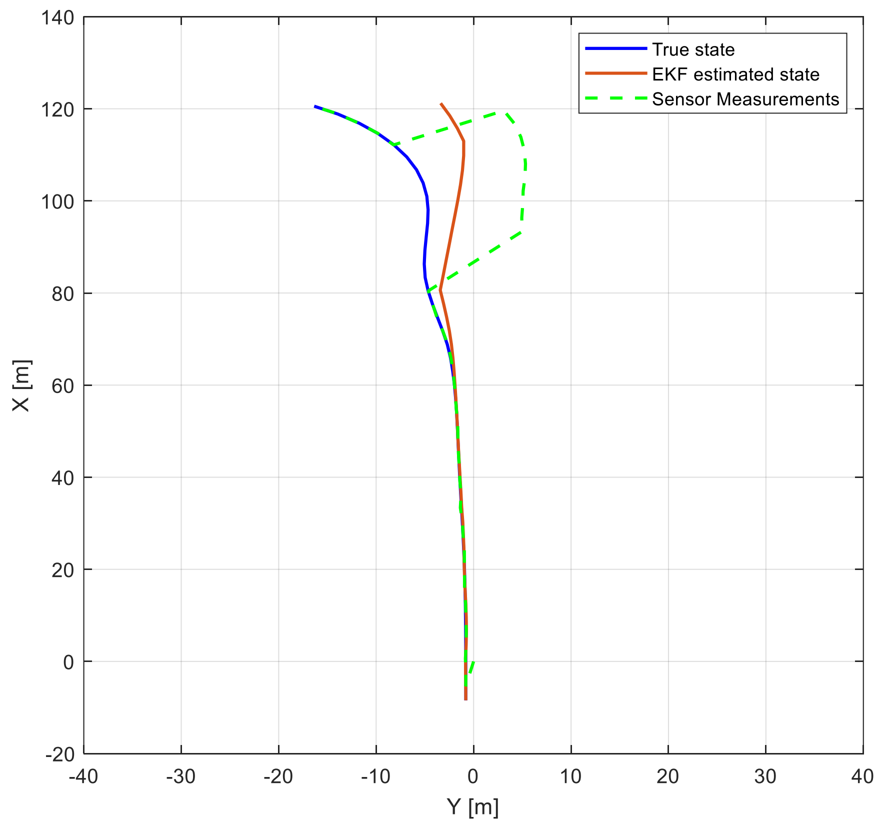
Innovation or measurement residual	$\tilde{\boldsymbol{y}}_k = \boldsymbol{z}_k - h(\hat{\boldsymbol{x}}_{k\|k-1})$
Innovation (or residual) covariance	$\boldsymbol{S}_k = \boldsymbol{H}_k \boldsymbol{P}_{k\|k-1} \boldsymbol{H}_k^T + \boldsymbol{R}_k$
Near-optimal Kalman gain	$\boldsymbol{K}_k = \boldsymbol{P}_{k\|k-1} \boldsymbol{H}_k^T \boldsymbol{S}_k^{-1}$
Updated state estimate	$\hat{\boldsymbol{x}}_{k\|k} = \hat{\boldsymbol{x}}_{k\|k-1} + \boldsymbol{K}_k \tilde{\boldsymbol{y}}_k$
Updated covariance estimate	$\boldsymbol{P}_{k\|k} = (\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k) \boldsymbol{P}_{k\|k-1}$

The EKF is implemented according to the above equations in the MATLAB code shown below, The Jacobian matrix should be calculated at each operating point inside the loop to implement EKF.

```
while iter < Tfinal/dt


       %% PART 2: Full state estimate considering all measurements using an EKF


       Fk=double(subs(Fk_j,[ux uy phi r X Y],xhat(:,iter-1)'));

       % Priori update
       x_prior(:,iter) = Fk*xhat(:,iter-1);
       P_prior = Fk*P*Fk';

       % Measurement update
       Sk=C*P_prior*C'+R;
       K = P_prior*C'*(eye(size(Sk))*inv(Sk));
       xhat(:,iter)=x_prior(:,iter)+K*(y(:,iter) - C*x_prior(:,iter));

       P= (eye(6)-K*C)*P_prior;
```

All the EKF estimates can be logged from the MATLAB code attached.

iii) The trajectory comparison for true value, sensor measurement and EKF estimate is shown in the figure below:



The EKF calculates the gain based on both the sensor measurements and the actual mode. Here since the GPS measurements are faulty, the EKF estimate does not exactly track the true trajectory. The EKF measurements are somewhere between the true and the sensor measurements as shown in the figure above.

iv) Decomposing the output space:

According the GOS framework, each observer input should have all the sensor outputs except the corresponding sensor output.

$$y^1 = C^1 X$$

For observer 1, the C^1 matrix is shown below: It will be an (8x6) matrix since Observer 1 will have all the sensor measurements except sensor 1.

```
C1 = [  1 0 0 0 0 0;
          0 0 1 0 0 0;
           0 0 0 1 0 0;
            0 0 0 0 1 0;
          0 0 0 0 0 1;
            0 0 0 0 1 0;
           0 0 0 0 0 1;
          0 0 1 0 0 0]
```

Similarly, the $C^2, C^3, C^4, C^5, C^6, C^7, C^8 and\ C^9$ matrices can be constructed.

The rank of observability matrix of each $(A, C^1), (A, C^2), \dots\dots (A, C^9)$ is 6 which is full. Hence each pair is still observable.

v) The observer design is implemented according to the equation below.

$$\dot{\hat{\mathbf{x}}}^{(1)} = \mathbf{A}\hat{\mathbf{x}}^{(1)} + \mathbf{B}\mathbf{u} + \mathbf{L}_1(\hat{\mathbf{y}}^{(1)} - \mathbf{y}^{(1)})$$

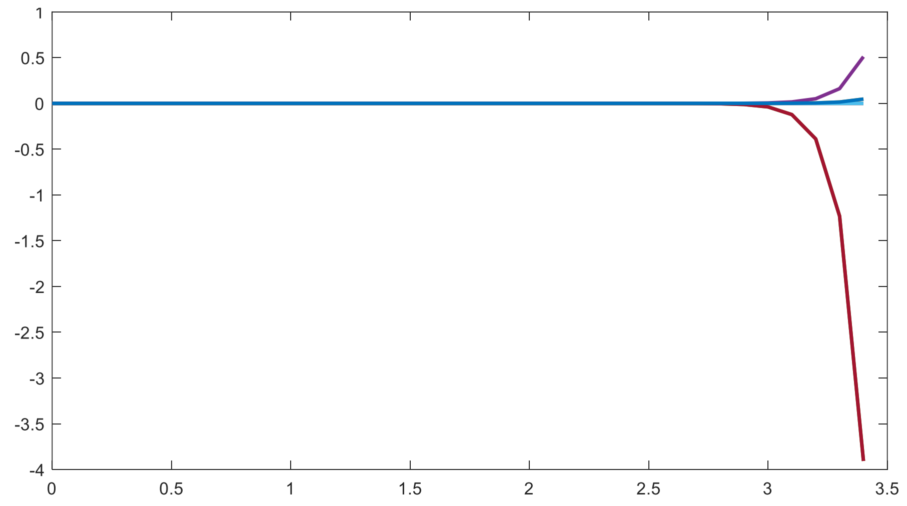
Since there are nine sensor measurements, there are nine observers. All the observers can be designed according to the code shown below:

```
% Loop to estimate observer states for each observer

for i = 1:45

   Xhat1(:,i+1) = dt*(A*Xhat1(:,i) + B*u_cl(:,i) + L1*((C1*Xhat1(:,i)) - y1(:,i))) + Xhat1(:,i);
   Xhat2(:,i+1) = dt*(A*Xhat2(:,i) + B*u_cl(:,i) + L2*((C2*Xhat2(:,i)) - y2(:,i))) + Xhat2(:,i);
   Xhat3(:,i+1) = dt*(A*Xhat3(:,i) + B*u_cl(:,i) + L3*((C3*Xhat2(:,i)) - y3(:,i))) + Xhat3(:,i);
   Xhat4(:,i+1) = dt*(A*Xhat4(:,i) + B*u_cl(:,i) + L4*((C4*Xhat2(:,i)) - y4(:,i))) + Xhat4(:,i);
   Xhat5(:,i+1) = dt*(A*Xhat5(:,i) + B*u_cl(:,i) + L5*((C5*Xhat2(:,i)) - y5(:,i))) + Xhat5(:,i);
   Xhat6(:,i+1) = dt*(A*Xhat6(:,i) + B*u_cl(:,i) + L6*((C6*Xhat2(:,i)) - y6(:,i))) + Xhat6(:,i);
   Xhat7(:,i+1) = dt*(A*Xhat7(:,i) + B*u_cl(:,i) + L7*((C7*Xhat2(:,i)) - y7(:,i))) + Xhat7(:,i);
   Xhat8(:,i+1) = dt*(A*Xhat8(:,i) + B*u_cl(:,i) + L8*((C8*Xhat2(:,i)) - y8(:,i))) + Xhat8(:,i);
   Xhat9(:,i+1) = dt*(A*Xhat9(:,i) + B*u_cl(:,i) + L9*((C9*Xhat1(:,i)) - y9(:,i))) + Xhat9(:,i);
end
```

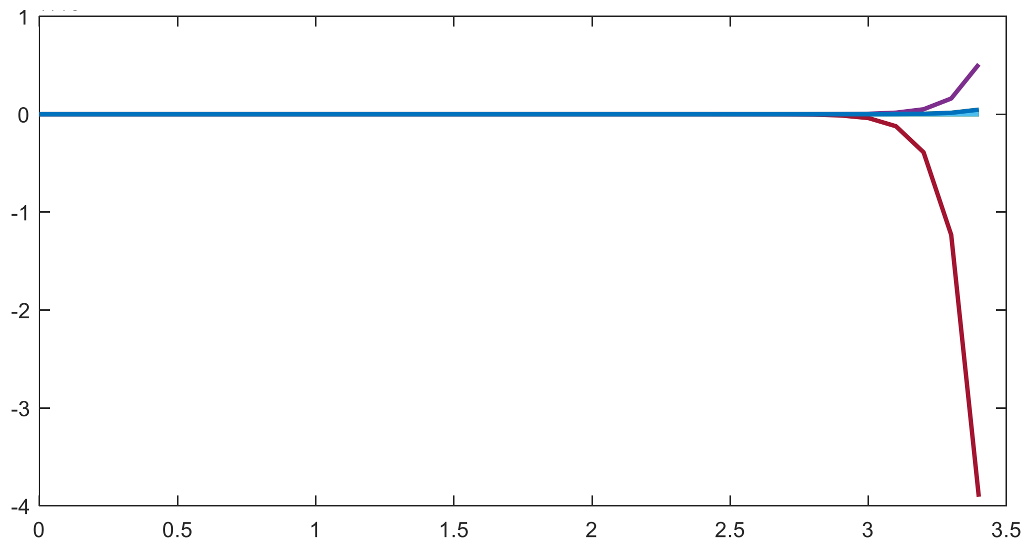
Since there is a fault in sensor 5 and 6 (GPS measurements in X and Y positions), According to the GOS detection logic, the residuals corresponding to sensors 5 and 6 should be zero and the rest of the residuals should be non zero.

vi) *Residual Plots:*

The residual plot for observer 1 is shown below: Notice that the residual is nonzero for all the states after 3 seconds. This is due to a malfunction in the GPS sensors between 3 and 4 seconds.



Similarly, the residual plot for observer 2 is shown below:



Hence except residuals for observer 5 and 6, all other residuals are non-zero. This implies that there exists a malfunction or fault in the sensor 5 and 6.