Exercise Round 1

The deadline of this exercise round is **Wednesday January 15, 2020**. The solutions will be gone through during the exercise session in room T2 in Konemiehentie 2 (CS) on that day starting at 14:15.

The problems should be *solved before the exercise session*, and during the session those who have completed the exercises may be asked to present their solutions on the board/screen.

Exercise 1. (Mean as the Minimum Mean Square Estimator)

Prove that mean of distribution $p(\theta)$ minimizes the expected value of the loss function

$$\mathrm{E}[(\theta - a)^2] = \int (\theta - a)^2 \, p(\theta) \; \mathrm{d}\theta. \tag{1}$$

Exercise 2. (Linear Least Squares Estimation)

Assume that we have obtained T measurement pairs (x_k, y_k) from the linear regression model

$$y_k = \theta_1 \, x_k + \theta_2, \qquad k = 1, 2, \ldots, T. \tag{2}$$

The purpose is now to derive estimates of the parameters θ_1 and θ_2 such that the following error is minimized (least squares estimate):

$$E(\theta_1, \theta_2) = \sum_{k=1}^{T} (y_k - \theta_1 \, x_k - \theta_2)^2. \tag{3}$$

(a) Define $\mathbf{y} = (y_1 \; \ldots \; y_T)^\mathsf{T}$ and $\boldsymbol{\theta} = (\theta_1 \; \theta_2)^\mathsf{T}$. Show that the set of Equations (2) can be written in matrix form

$$\mathbf{y} = \mathbf{X} \, \boldsymbol{\theta},$$

with a suitably defined matrix $\mathbf{X}$.

(b) Write the error function in Equation (3) in matrix form in terms of $\mathbf{y}$, $\mathbf{X}$ and $\boldsymbol{\theta}$.

(c) Compute the gradient of the matrix form error function and solve the least squares estimate of the parameter $\boldsymbol{\theta}$ by finding the point where the gradient is zero.

Exercise 3. (Kalman filtering with the EKF/UKF Toolbox)

You are also allowed to do all the steps below using a suitable Python library or R library if you for some reason don't have an access to Matlab.

(a) Download and install the EKF/UKF toolbox to some Matlab computer from the web page:

`https://github.com/EEA-sensors/ekfukf`

Run the following demonstrations:

```
demos/kf_sine_demo/kf_sine_demo.m
demos/kf_cwpa_demo/kf_cwpa_demo.m
```

After running them, read the contents of these files and try to understand how they have been implemented. Also read the documentations of functions `kf_predict` and `kf_update` (type *e.g.* "`help kf_predict`" in Matlab).

(b) Consider the following state space model:

$$\begin{aligned} \mathbf{x}_k &= \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \mathbf{x}_{k-1} + \mathbf{w}_{k-1} \\ y_k &= \begin{pmatrix} 1 & 0 \end{pmatrix} \mathbf{x}_k + v_k \end{aligned} \tag{4}$$

where $\mathbf{x}_k = (x_k \; \dot{x}_k)^\mathsf{T}$ is the state, y_k is the measurement, and $\mathbf{w}_k \sim \mathrm{N}(\mathbf{0}, \mathrm{diag}(1/10^2, 1^2))$ and $v_k \sim \mathrm{N}(0, 10^2)$ are white Gaussian noise processes.

Simulate a 100 step state sequence from the model and plot the signal x_k, signal derivative $\dot{x}_k$ and the simulated measurements y_k. Start from an initial state drawn from a zero-mean 2d-Gaussian distribution with identity covariance.

(c) Use the Kalman filter for computing the state estimates $\mathbf{m}_k$ using the following kind of Matlab-code:

```
m = [0;0];  % Initial mean
P = eye(2); % Initial covariance
for k = 1:100
   [m,P] = kf_predict(m,P,A,Q);
   [m,P] = kf_update(m,P,y(k),H,R);
   % Store the estimate m of state x_k here
end
```

(d) Plot the state estimates $\mathbf{m}_k$, the true states $\mathbf{x}_k$ and measurements y_k. Compute the RMSE (root mean square error) of using the first components of vectors $\mathbf{m}_k$ as the estimates of first components of states $\mathbf{x}_k$. Also compute the RMSE error that we would have if we used the measurements as the estimates.