Advanced topics in data modelling

Assignment 3: Tracking with Bayesian Filtering

Mariuta Nicolae (rqt629) June 8, 2015

# Question 1

For the implementations of filter I did, unfortunately, not make functions with parameters to use them for each type of model, and I have written only one code that also includes the application of chosen models and plots for the pendulum data.

For the implementation of Kalman filter I start with defining the parameters needed for the algorithm: the initial state, measurement and observation noise, parameters A, B and C for the update functions of dynamical and observation model, and also matrices to store the results.

The main implementation of the algorithm is inside the for-loop where I follow the algorithm exactly, by doing the necessary matrix multiplication for prediction and correction (as they are also described in the course slides). The operation use matrices, so the code can be adapted for m-dimensional observations and n-dimensional state spaces.

The estimate of pendulum movement obtained by the filter is stored in *Q\_estimate* matrix and then it is plotted separately the last 100 values for coordinates *x* and *y*.

# Question 2

For the implementation of particle filter I have also followed the algorithm exactly as described in the documentation for the course: it starts with initial sampling with chosen variance,

For each pair of coordinates for the pendulum, the particles are moved according to the model function and weights are calculated for these particles. After the normalization is applied to form a probability distribution, the resampling with replacement is performed.

The code contains many comments that explain every part of the algorithm that is implemented.

# Question 3&4

## Kalman filter

For the application of Kalman filter to the pendulum data, I have chosen the following dynamical model:

In the dynamical model, *x* and *y* are the coordinates of pendulum, x’ is the velocity on the x coordinate and y’ is the velocity on the y coordinate. T is the time stamp, u is the acceleration of pendulum (same for each coordinate) and Ex is the error in measurement. I have chosen the model so that, by doing the matrix multiplication, to obtain the next coordinates of the points according to laws of movement from physics.

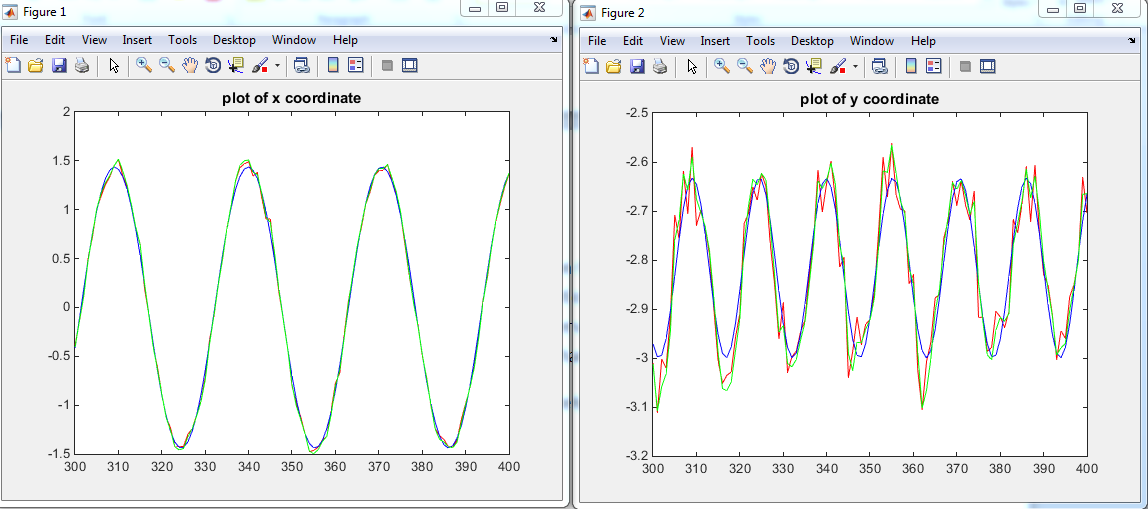
The observational model that I have chosen to use is:

Where I take the output of the observational model, multiply with unit matrix so I obtain the two coordinates of observed pendulum and the observation noise is also added.

Overall, I have chosen to model the coordinates of pendulum and velocity on both directions for measurement and consider only the coordinate of pendulum for the observation.

I have a separate piece of code where I apply the same algorithm but I also do grid search to check how I obtain the lowest RMS for different values of acceleration, the sampling rate, error for the measurement and error for the observation. I have used the data in true pendulum because I understood that it is allowed from the posts on the forum. I have concluded that the best results are obtained for sampling rate 0.5, acceleration 0.001, measurement noise 31 for the x axis, measurement noise 91 for the y axis and observation noise 31. For these values, I obtained RMS = (0.0461, 0.0422) that is slightly better than the RMS of the noisy data.

I have also plotted the last 100 values of x and y separately because like that is was easier to track the differences:



The blue color represents the movement of the true pendulum, the red line represents the noisy pendulum and green shows the estimated movement of the pendulum.

In the plot is visible that there is more noise in observation of x coordinate than it is for the y coordinate, perhaps because of the position of sensor and that is why I obtained better results for higher value of measurement noise on y coordinate in the application of Kalman algorithm for this data.

Perhaps an improvement could have been to also consider the acceleration on both x and y coordinates for the creation of dynamical model which could lead to slightly improved results but I did not have enough time for making more experiments.

## Particle filter

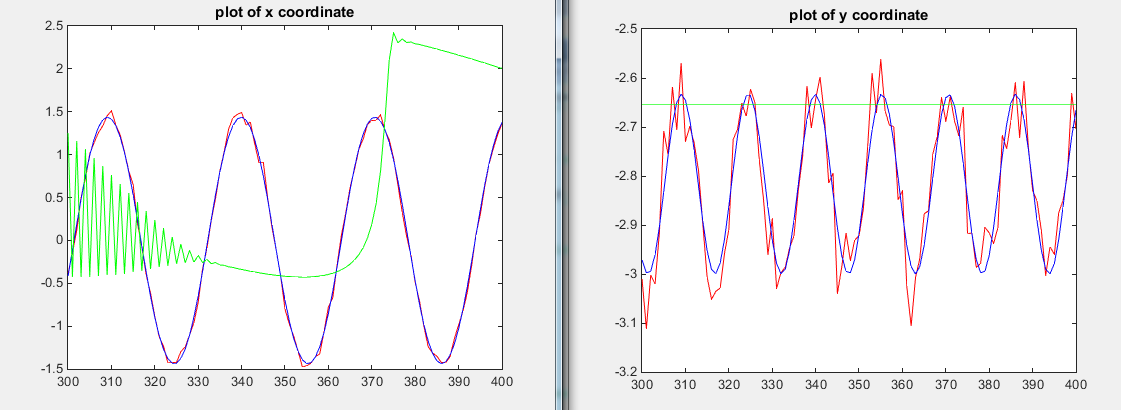
For the application of particle filter on the input data, I unfortunately did not obtain good results even if I tried I have tried two different models.

First of all, I tried to use the observation from the Kalman filter that the variation of coordinates x and y is sinusoidal and I used a sinusoidal function for the dynamical model:

For this equation, C is the mean obtained from the data of pendulum; I approximated the amplitude A by looking at plots and calculated the frequency in the same way. For the phase (x) I considered the values of initial samples that are obtained in the first stage of the algorithm.

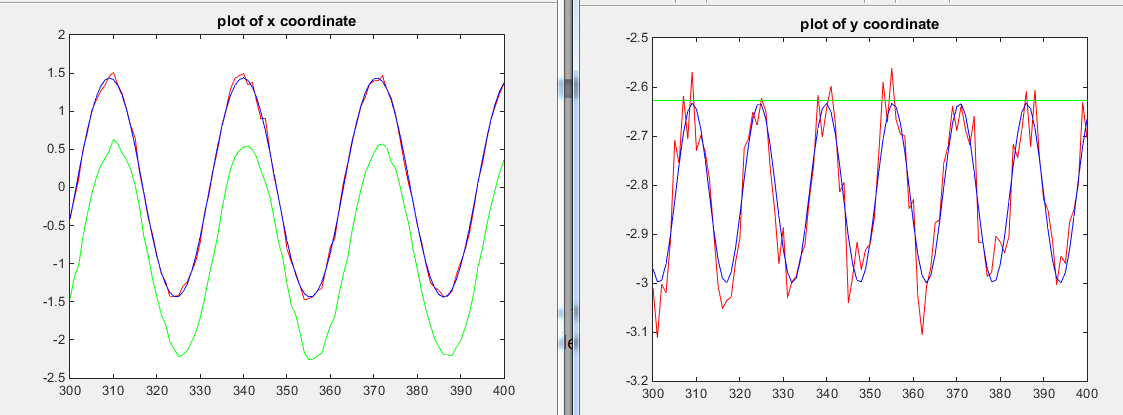
For the observation model, I just took the value obtained through measurement and added observation noise.

Unfortunately, it can be seen in the plots that the results obtained were very bad and the RMS is very far from a good value.



I think this happened because, perhaps, the sinusoid is not constant in time and because of decreasing energy, the amplitude of oscillation for pendulum is decreasing over time.

In a second attempt, I used the same model from Kalman filter by modeling coordinates and velocity at the dynamical model and only the coordinates for the observation model. Unfortunately, once again, the results are not anywhere close to the value of the true pendulum, no matter what values I tried.



I am pretty sure that the choice of model is responsible for the bad results that I obtained by using the particle filter. I thought about looking and the physics laws involved in pendulum movement and calculate its angle but I did not understand very well that math and I could not build a model to describe that behavior.