Perception in Robotics
Term 2, 2018. PS1

Gonzalo Ferrer
Skoltech

November 19, 2018

Task 1: Probability (25 points)

Part A

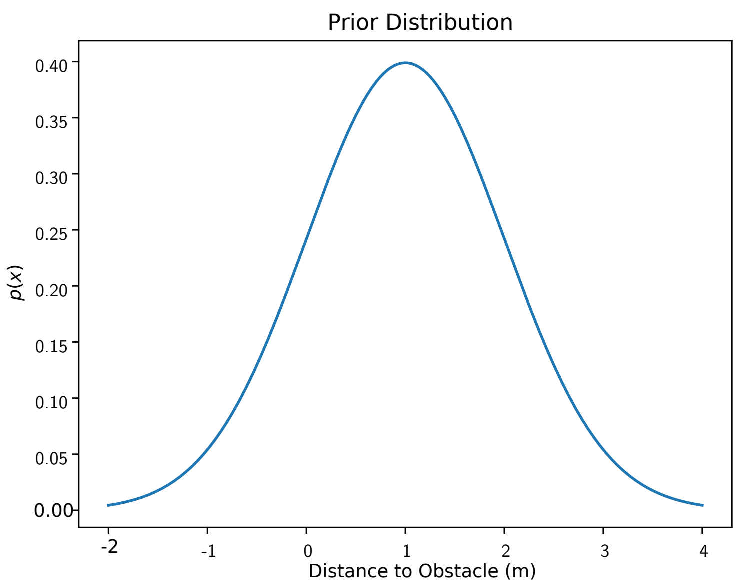


Figure 1: The prior distribution of the distance to obstacle from the robot.

Part B

The probability that we are actually colliding with the wall is $P(X \leq x) = \int_{-\infty}^{x} p(x)dx$. This can be computing using Python `norm.cdf()` which computes the standard normal CDF. Therefore, the probability that we are actually colliding with the wall, $P(X \leq 0) \approx 0.1587$.

Part C

The posterior distribution, $p(x|z)$ can be computed using the the Bayes Rule as illustrated in (1).

$$p(x|z) = \frac{p(z|x)p(x)}{p(z)} = \frac{p(z|x)p(x)}{\int_{x'} p(z|x')p(x')} \tag{1}$$

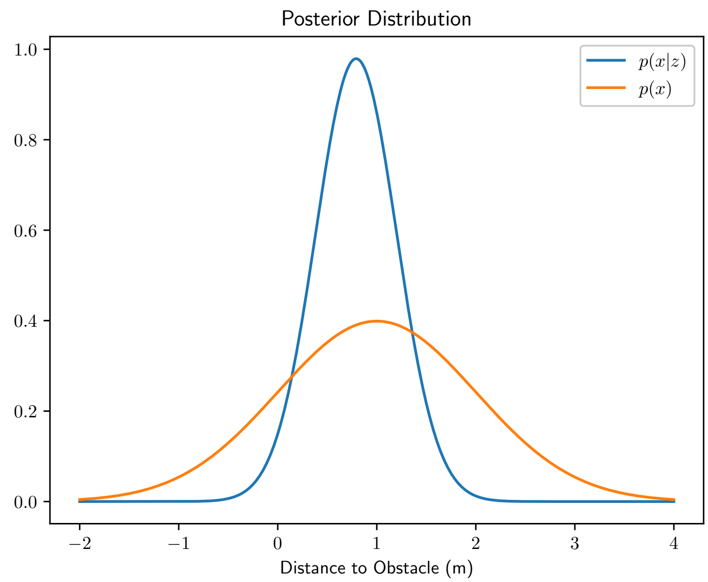


Figure 2: The posterior distribution of the distance to the obstacle given the observation $z = 0.75$.

Note that in case that we are using the normalized version of the Bayes' theorem, i.e., $p(x|z) = \eta p(z|x)p(x)$, we must normalize $p(x|z)$ such as $\int_{-\infty}^{\infty} p(x|z) = 1$.

Part D

The conditional expectation is given by $E[x|z] = \int_{-\infty}^{\infty} xp(x|z)dx$. This can be computed numerically as well as from peeking at the plot of the posterior distribution. Moreover, the conditional expectation is the expected value of the posterior distribution: $E[x|z] \approx 0.79$.

Part E

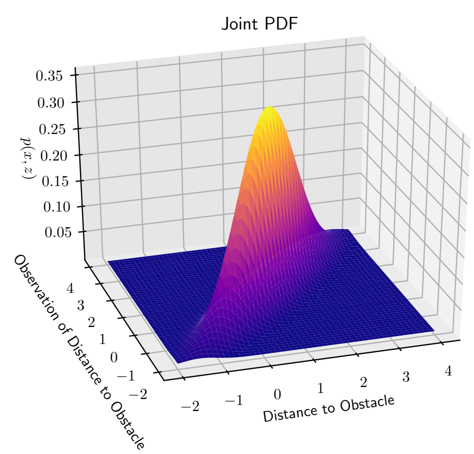
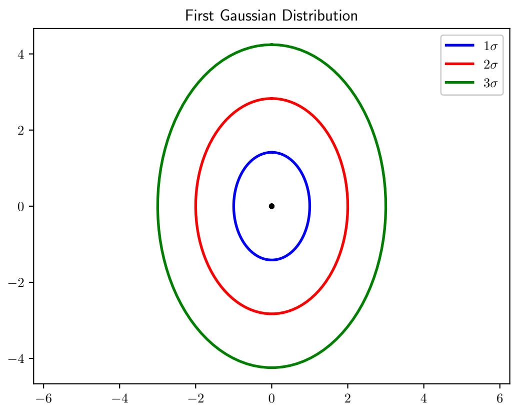


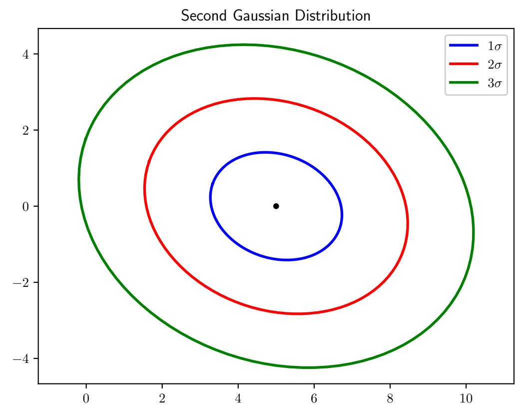
Figure 3: The joint distribution of distance to the obstacle and the corresponding observation.

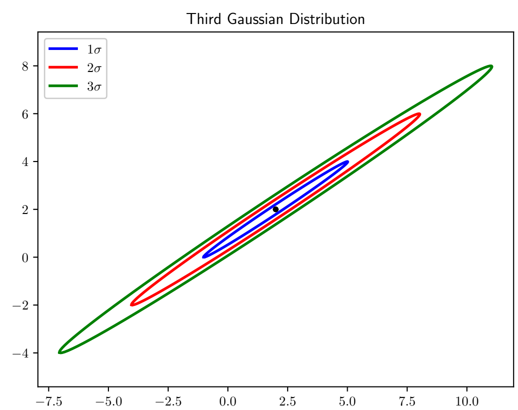
Task 2: Multivariate Gaussian (25 points)

Part A

Let an iso-contour be given by the distribution $\mathcal{N}(x;\mu,\Sigma)$. To plot the iso-contour, we extract points from a circle with radius k and project these points on to the iso-contour using a projection matrix. If $\Sigma = USV^T$ (a.k.a. the singular value decomposition), then the projection matrix is given by $P = U\sqrt{S}$. In addition, if $\Sigma = LL^T$ (the Cholesky decomposition with the lower triangular matrix, L), the projection matrix is given by $P = L$. In both the cases, the point $\vec{p_i^c} = (x_i^c, y_i^c)$ on the circle can be projected on to the ellipse by $\vec{p_i^e} = P\,\vec{p_i^c} + \mu$.







Part B

Let $\{x_i\}_{i=1}^N$ be the points from which to estimate the mean and covariance. The sample mean, $\vec{\mu}_x$, is given by (2), and the sample covariance, Σ_{xx}, is given by (3).

$$\vec{\mu}_x = E[\mathbf{x}] = \frac{1}{N}\sum_{i=1}^{N}\vec{x}_i \tag{2}$$

$$\Sigma_{xx} = E[(\mathbf{x} - E[\mathbf{x}])(\mathbf{x} - E[\mathbf{x}])^T] = \frac{1}{N-1}\sum_{i=1}^{N}(\vec{x}_i - \vec{\mu}_x)(\vec{x}_i - \vec{\mu}_x)^T \tag{3}$$

Part C

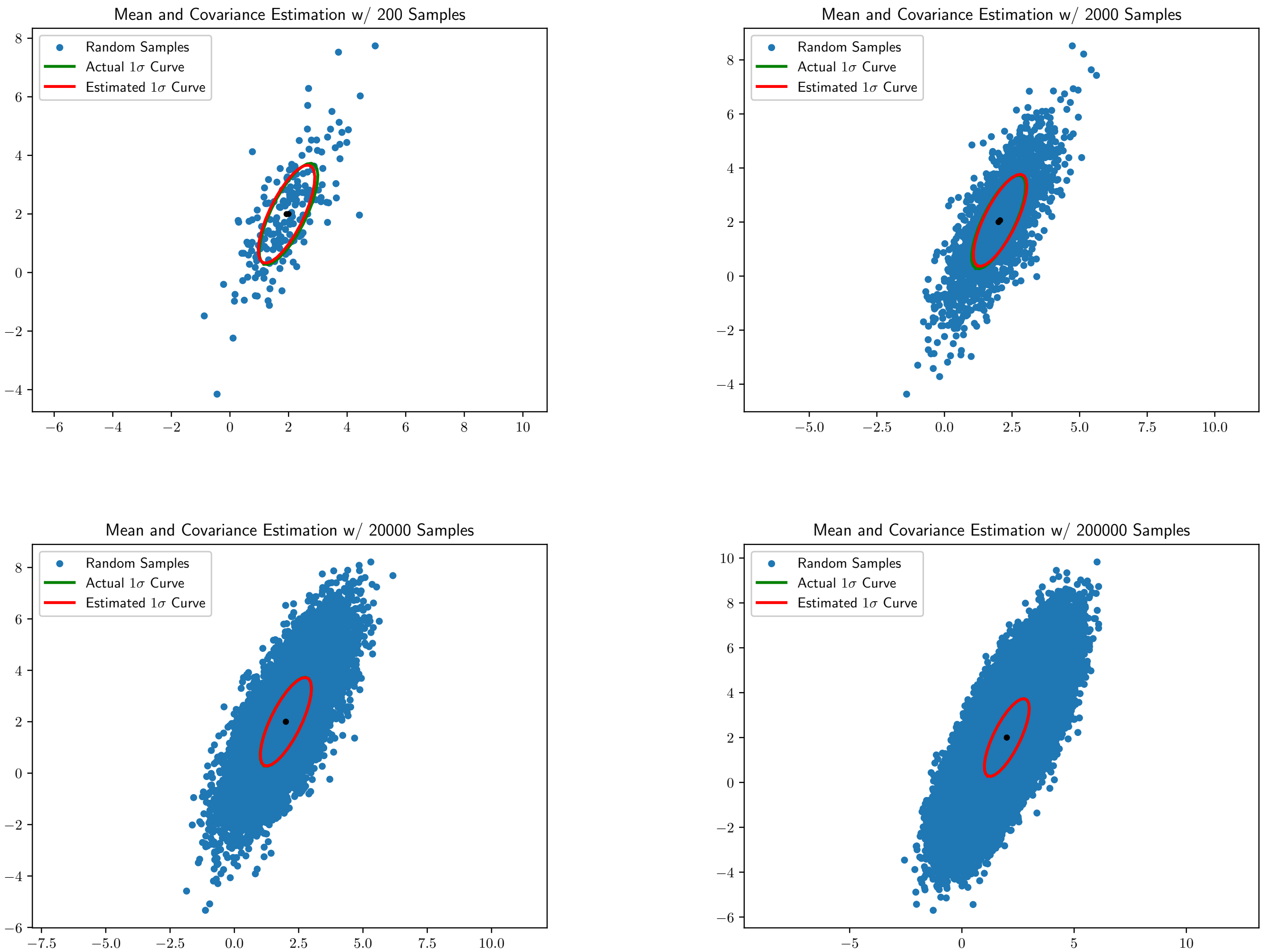


Figure 4: Actual and estiamated mean and covariance of 200, 2k, 20k, and 2M samples.

It's evident from Figure 4 that the more samples we draw from the distribution, the closer the estimated mean and covariance becomes to the actual mean and covariance, respectively.

Task 3: Covariance Projection (25 pts)

Part A

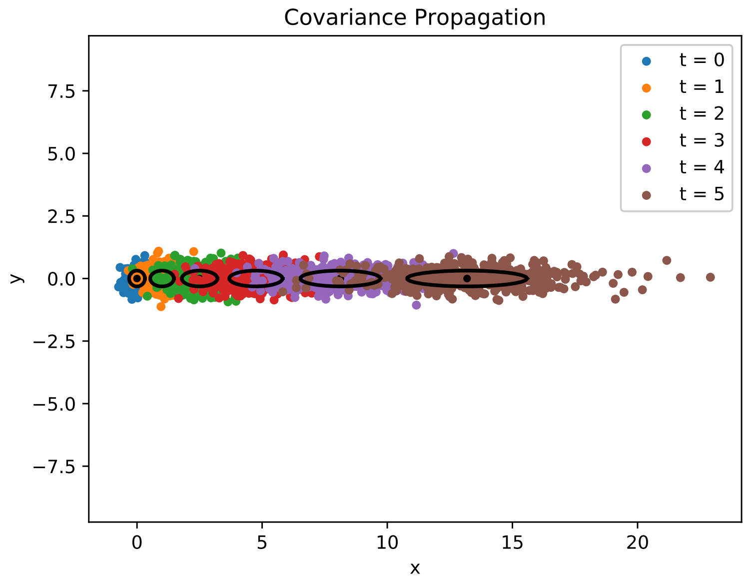


Figure 5: The state *pdf* propagation of the strangely moving robot.

Part B

The mean and covariance propagation for a singal iteration is given by (4) and (5), respectively. These equations can be placed inside a loop to be used for recursively propagating the state or in the case of particles, by following the sequence: sampling, estate estimation (sample mean, sample covariance) and then sampling from this estimated distribution on the next steps.

$$\mu_{\mathbf{t+1}} = A\mu_{\mathbf{t}} + Bu_t = \begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}_t + \begin{bmatrix} \Delta t & 0 \\ 0 & \Delta t \end{bmatrix} \begin{bmatrix} v_x \\ v_y \end{bmatrix} \tag{4}$$

$$\Sigma_{t+1} = A\Sigma_t A^T + R_t = \begin{bmatrix} \Sigma_{xx} & \Sigma_{xy} \\ \Sigma_{yx} & \Sigma_{yy} \end{bmatrix}_t + \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix} \tag{5}$$

Part C

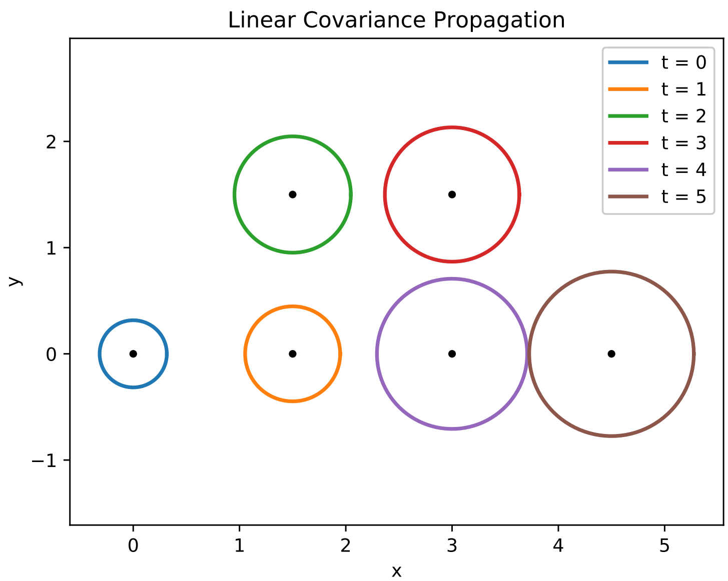


Figure 6: The linear state *pdf* propagation of the properly moving robot.

Part D

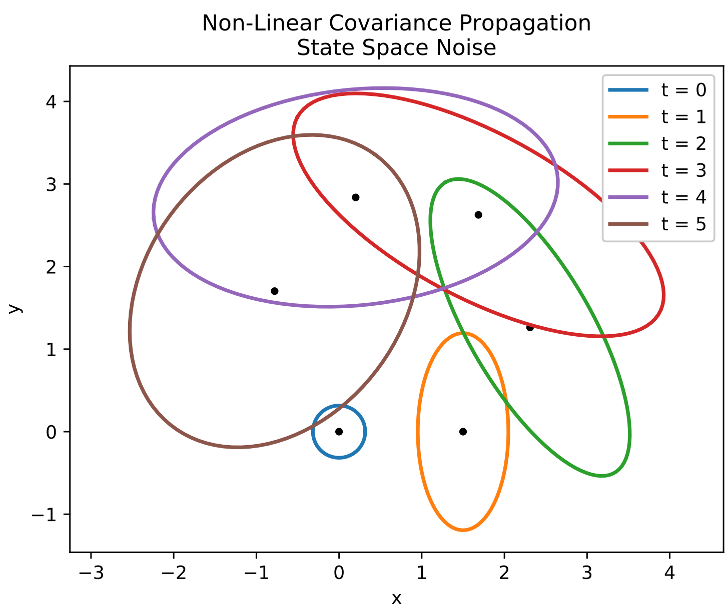


Figure 7: Non-linear state *pdf* propagation with noise given in the state space.

Part E

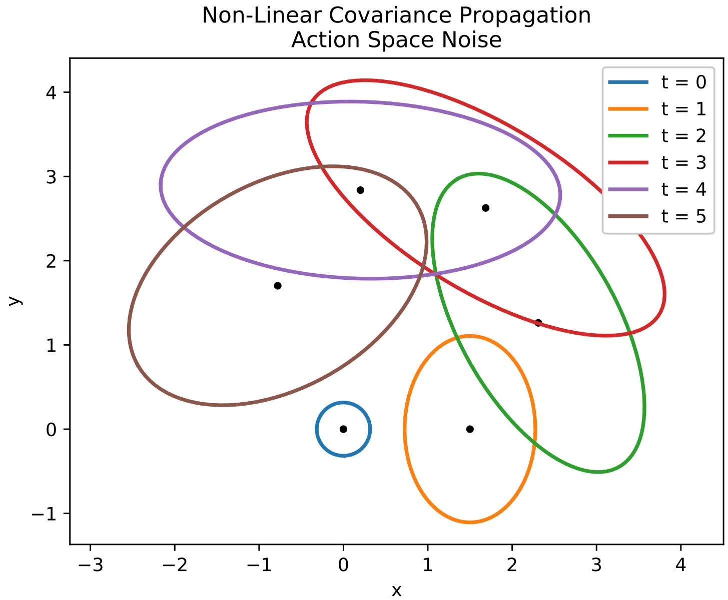


Figure 8: Non-linear state *pdf* propagation with noise given in the action space.

Task 4: Kalman Filter (25 pts)

Part A

Given that the propagation model is $x_t = x_{t-1} + \Delta t(u_t + \epsilon_t)$, the prediction step in the Kalman Filter, for this problem, is given by:

$$\bar{\mu}_t = \mu_{t-1} + \Delta t u_t \tag{6}$$

$$\bar{\Sigma}_t = \Sigma_{t-1} + (\Delta t)^2 M \tag{7}$$

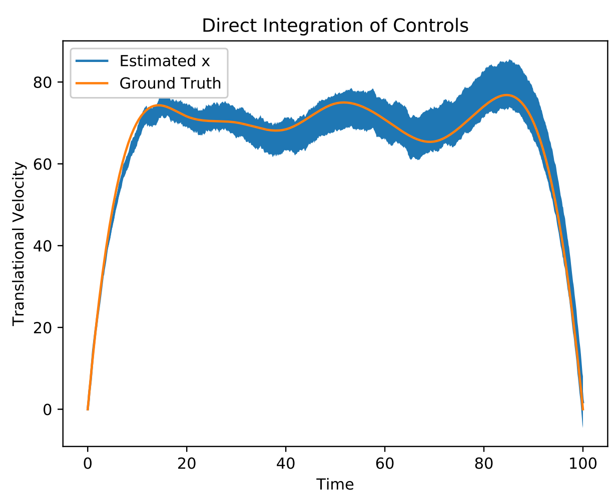


Figure 9: The Kalman Filter with only prediction.

From Figure 9, it's evident that the uncertainty in the translational velocity grows with time without the update step (i.e. incorporating the measurements to correct the predictions).

Part B

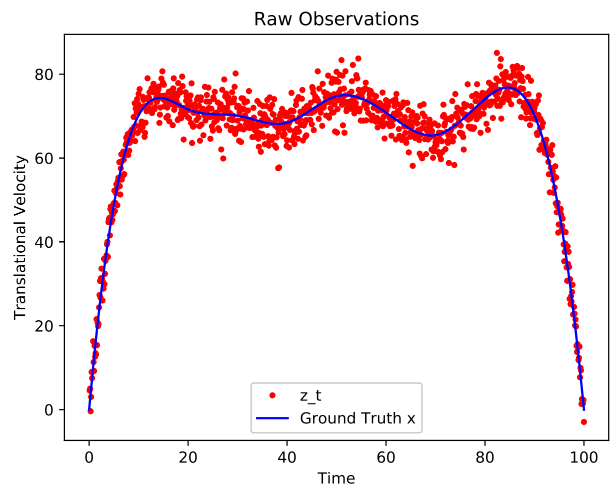


Figure 10: The raw observations of the translational velocity accumulated by the accelerometer.

Part C

The Kalman Filter's update step is given as follows:

$$A = 1$$
$$B = \Delta t = 0.1$$
$$C = 1$$
$$K_t = \frac{\bar{\Sigma}_t}{\bar{\Sigma}_t + 10}$$
$$\mu_t = \bar{\mu}_t + K_t(z_t - \bar{\mu}_t)$$
$$\Sigma_t = (1 - K_t)\bar{\Sigma}_t$$

Part D

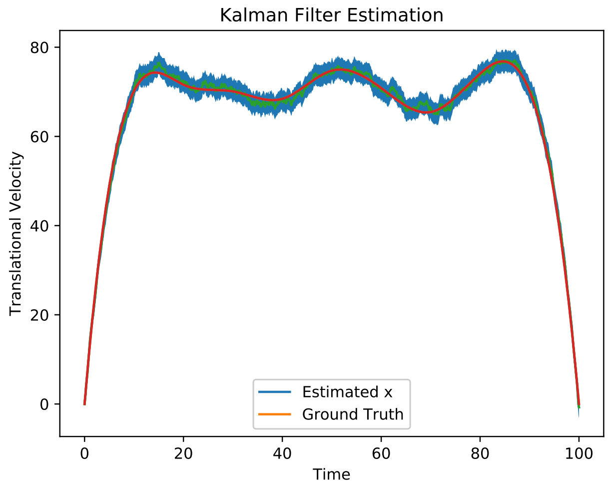


Figure 11: The complete Kalman Filter implementation with the prediction and update steps.

Part E

From comparing the two plots, it's clear that with just the prediction step, even the 1σ bounds are relatively large and diverge through time. However, with both the prediction and update steps, both the 1σ and 3σ curves are relatively smaller, indicating the better certainty in the state estimation.