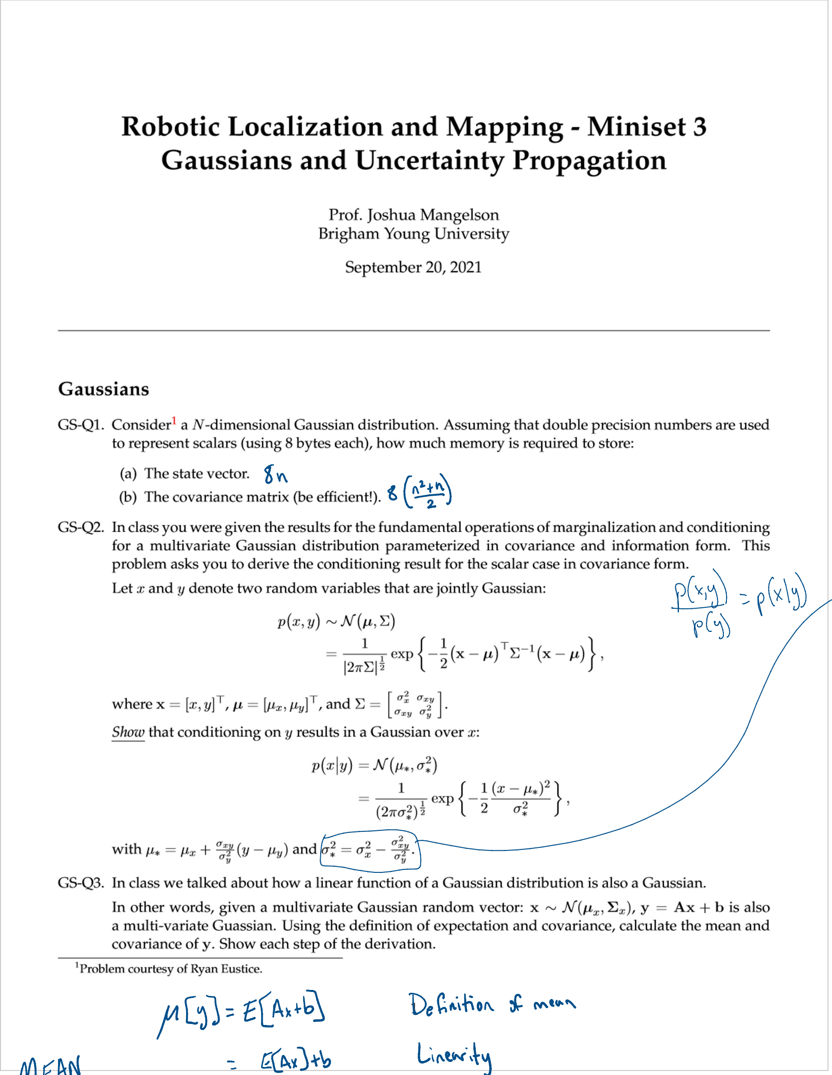
Miniset 3

Monday, September 26, 2022 1:32 PM



Robotic Localization and Mapping - Miniset 3
Gaussians and Uncertainty Propagation

Prof. Joshua Mangelson
Brigham Young University

September 20, 2021

Gaussians

GS-Q1. Consider[1] a N-dimensional Gaussian distribution. Assuming that double precision numbers are used to represent scalars (using 8 bytes each), how much memory is required to store:

(a) The state vector. $8n$

(b) The covariance matrix (be efficient!). $8\left(\frac{n^2+n}{2}\right)$

GS-Q2. In class you were given the results for the fundamental operations of marginalization and conditioning for a multivariate Gaussian distribution parameterized in covariance and information form. This problem asks you to derive the conditioning result for the scalar case in covariance form.

Let x and y denote two random variables that are jointly Gaussian:

$$\frac{p(x,y)}{p(y)} = p(x|y)$$

$$\begin{aligned} p(x,y) &\sim \mathcal{N}(\boldsymbol{\mu}, \Sigma) \\ &= \frac{1}{|2\pi\Sigma|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}, \end{aligned}$$

where $\mathbf{x} = [x, y]^\top$, $\boldsymbol{\mu} = [\mu_x, \mu_y]^\top$, and $\Sigma = \begin{bmatrix} \sigma_x^2 & \sigma_{xy} \\ \sigma_{xy} & \sigma_y^2 \end{bmatrix}$.

Show that conditioning on y results in a Gaussian over x:

$$\begin{aligned} p(x|y) &= \mathcal{N}(\mu_*, \sigma_*^2) \\ &= \frac{1}{(2\pi\sigma_*^2)^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}\frac{(x-\mu_*)^2}{\sigma_*^2}\right\}, \end{aligned}$$

with $\mu_* = \mu_x + \frac{\sigma_{xy}}{\sigma_y^2}(y - \mu_y)$ and $\sigma_*^2 = \sigma_x^2 - \frac{\sigma_{xy}^2}{\sigma_y^2}$.

$$\mu' = \mu_x + \Sigma_{xy}\Sigma_{yy}^{-1}(y-\mu_y) = \mu_x + \frac{\sigma_{xy}}{\sigma_{yy}}(y-\mu_y)$$ Schur complement

$$\Sigma' = \Sigma_{xx} - \Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{yx}$$ Schur complement

$$p(x|y) = \frac{1}{|2\pi\Sigma'|^{1/2}} \exp\left\{-\frac{1}{2}(x-\mu')^T\Sigma'^{-1}(x-\mu')\right\}$$

$$= \frac{1}{2\pi\sigma_*^2} \exp\left\{-\frac{1}{2}(x-\mu')^T\Sigma'^{-1}(x-\mu')\right\}$$ Sub in Schur complement

$$= \frac{1}{2\pi\sigma_*^2} \exp\left\{-\frac{1}{2}\frac{(x-\mu')^2}{\sigma_*^2}\right\}$$

GS-Q3. In class we talked about how a linear function of a Gaussian distribution is also a Gaussian.

In other words, given a multivariate Gaussian random vector: $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x)$, $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{b}$ is also a multi-variate Guassian. Using the definition of expectation and covariance, calculate the mean and covariance of $\mathbf{y}$. Show each step of the derivation.

[1] Problem courtesy of Ryan Eustice.

MEAN

$$\mu[y] = E[Ax+b]$$ Definition of mean

$$= E[Ax]+b$$ Linearity

$$\boxed{= A\,E[x]+b}$$

COVARIANCE

$$\Sigma_{yy} = E\left[(y-E(y))(y-E(y))^T\right]$$

$$= E\left[(Ax+b-E(Ax+b))(Ax+b-E(Ax+b))^T\right]$$ Sub for y

$$= E\left[(Ax+\cancel{b}-\cancel{b}-AE(x))(Ax+\cancel{b}-\cancel{b}-AE(x))^T\right]$$ Linear Function

$$= E\left[A(x-E(x))(x-E(x))^T A^T\right]$$ Simplify and pull out A

$$= A\,E\left[(x-E(x))(x-E(x))^T\right]A^T$$ Linearity

$$\boxed{\Sigma_{yy} = A\Sigma_{xx}A^T}$$

$\sigma_{range} = 0.5m$

$\sigma_{bearing} = 0.25\ rad$

Uncertainty Propagation

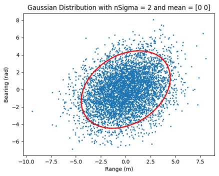
In[2] most SLAM or localization approaches, the noise characteristics of individual observations are linearized, resulting in a Gaussian covariance matrix. In this task, we explore the effects of the error introduced via linearization.

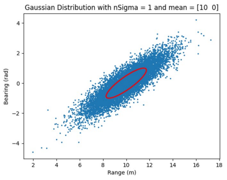
Consider a robot sensor that observes range and bearing to nearby landmarks. In this case, the range error is relatively small, but the bearing error is large. We are interested in determining the (x, y) position of the beacon based on observations obtained by the robot at the origin $(0, 0)$. For this problem, the robot does not move.

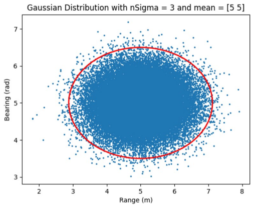
Suppose you obtain a $(range, bearing)$ observation with mean $(10.0m, 0rad)$ whose range standard deviation is $0.5m$, and whose bearing standard deviation is $0.25rad$. The noise in range and bearing measurements is Gaussian and independent.

Use python to answer the following questions:

UP-Q0. Generate a function in python that will plot the n-sigma ellipse for a 2d-guassian distribution given a mean vector and covariance matrix. Use the matlab file plotcov2d.m for reference. Try generating 10,000 samples from several different distributions (by varying the mean and covariance matrix) and then plot the points and the covariance ellipse.





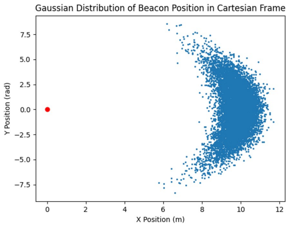


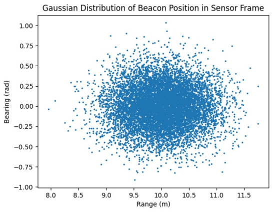
UP-Q1. Generate and plot a point cloud representing 10,000 samples from the distribution over the position of the beacon as measured in

i) the sensor frame, i.e. (r, θ) space, and

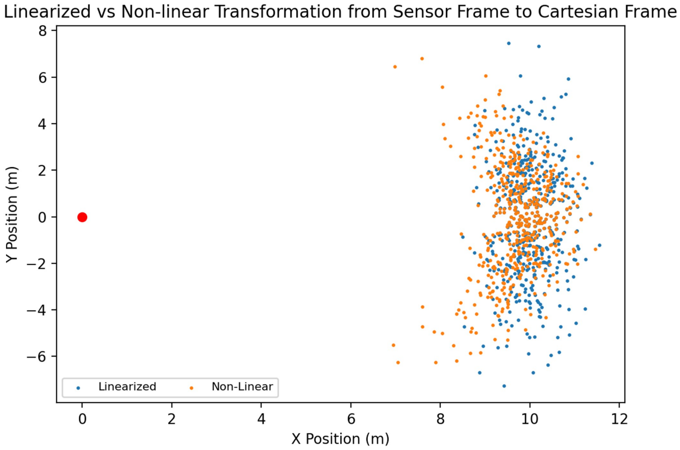
ii) the Cartesian (x, y) coordinate frame.

In other words, generate observations of $(range, bearing)$ and project these points into (x, y). (Hint: use the numpy.random.normal function).

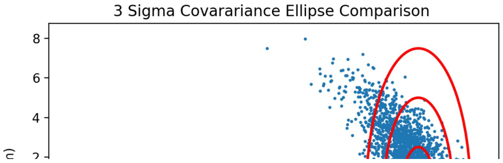


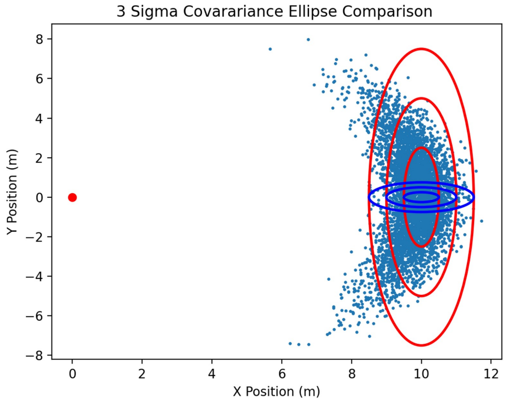


UP-Q2. What is the (linearized) covariance of the beacon position in (x, y) coordinates? In other words, write the covariance of an observation in (x, y) coordinates in terms of the covariance of the observation in (range, bearing) coordinates. The transformation is non-linear, so you will need to compute a first-order approximation (Taylor expansion) of the transformation function. Make the appropriate Jacobians easy to read in your source code, using comments if necessary.



UP-Q3. Draw in red the 1-sigma, 2-sigma, and 3-sigma contours of the analytical (linearized) covariance ellipses, super-imposed over the point clouds generated in parts UP-Q1.i and UP-Q1.ii. Now overlay in blue the actual covariance ellipses computed using sample-based expressions for the first and second moments. Do they agree? Why or why not?





UP-Q4. From a purely theoretical perspective, assuming that the underlying process is truly Gaussian, we expect 39.35% of all samples to lie within the 1-sigma contour, 86.47% of samples to lie within the 2-sigma contour, and 98.89% to lie within the 3-sigma contour. (These frequencies were computed using the cumulative chi-square distribution for two degrees-of-freedom, i.e. χ_2^2. In python, you can do this using the scipy.stats.chi2 object and the chi2.cdf function evaluated at chi2.cdf(1, 2), chi2.cdf(4,2), and chi2cdf(9, 2) respectively.)

Modify your software to count the samples falling within each (analytical) ellipse for parts UP-Q1.i and UP-Q1.ii. The error of a particular sample x, measured in "units" of sigma, is known as the Mahalanobis distance, and can be computed as $\sqrt{(x-\mu)^\intercal \Sigma^{-1}(x-\mu)}$.

UP-Q5. If the point samples were truly distributed as a Gaussian (clearly the (x, y) are not), your counts would match the theoretically predicted values. Try varying the noise parameters: under what conditions do the counts come close to matching the theoretically predicted values? What consequences to a state estimation algorithm could these sorts of errors have?

UP-Q6. Suppose now that the $(range, bearing)$ measurements are not independent but instead jointly correlated under the following three scenarios:

a) $\rho_{r\theta} = 0.1$,

b) $\rho_{r\theta} = 0.5$,

c) $\rho_{r\theta} = 0.9$.

Repeat UP-Q1 and UP-Q3. (Hint, use the Cholesky decomposition).