Gaussian Process Regression for an obstacle-free 2D case

Prepared by Tung Nguyen
Prepared for Prof. James MacLean

1. Introduction

In the first part of this experiment, we shall use the Gaussian Process Regression technique to predict the sensor readings at a test location on a 3D map where training data (input = location ↔ target = sensor readings) have been collect in advance. To execute this part, please run the function `PredictSensorReadings.m.`

In the second part, we shall use the Gaussian Process Regression technique to infer the robot's location where a set of sensor readings have been collected. In other words, sensor readings are now the inputs and robot locations are now the targets. To execute this part, please run the function `PredictLocations.m.`

2. Background

We will use the Gaussian Process Regression algorithm described by Rasmussen and Williams in their book. We will make use of equations 2.22-2.24 on page 16:

$$\mathbf{f}_*|X,\mathbf{y},X_* \sim \mathcal{N}\big(\bar{\mathbf{f}}_*, \operatorname{cov}(\mathbf{f}_*)\big), \text{ where} \qquad (2.22)$$

$$\bar{\mathbf{f}}_* \triangleq \mathbb{E}[\mathbf{f}_*|X,\mathbf{y},X_*] = K(X_*,X)[K(X,X)+\sigma_n^2 I]^{-1}\mathbf{y}, \qquad (2.23)$$

$$\operatorname{cov}(\mathbf{f}_*) = K(X_*,X_*) - K(X_*,X)[K(X,X)+\sigma_n^2 I]^{-1}K(X,X_*). \qquad (2.24)$$

X : design matrix (D by n) where D is the dimension of the input vector, n is the number of input vectors

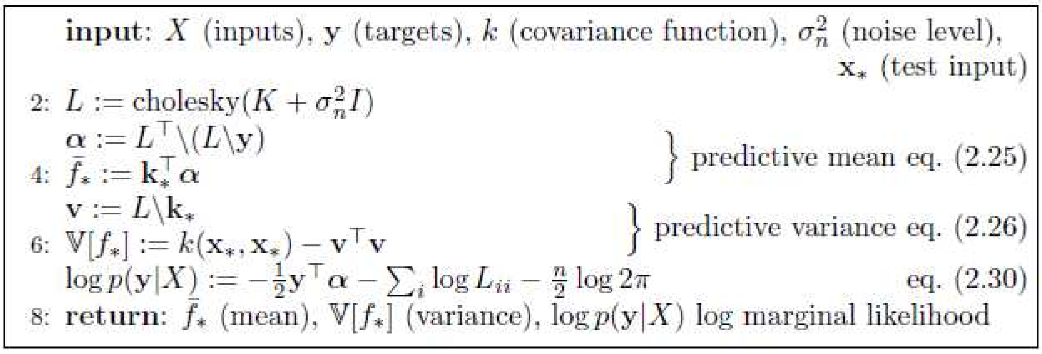
y : vector that contains targets (1 by n)

K(X,X) : covariance matrix whose entries are calculated using the squared exponential function:

$$k_y(x_p,x_q) = \sigma_f^2 \exp\left(-\frac{1}{2\ell^2}(x_p-x_q)^2\right) + \sigma_n^2\delta_{pq}. \qquad (2.31)$$

x_*	: test input
$\mathbf{f}_*$	: prediction of the test output
$\bar{\mathbf{f}}_*$	: mean of the prediction of the test output
σ_n^2	: noise variance

We will follow Rasmussen's pseudo-code. Function `simple1D_GPPredictor(X, y, hyperparams, x_star)` follows every step shown below:



```
   input: $X$ (inputs), $\mathbf{y}$ (targets), $k$ (covariance function), $\sigma_n^2$ (noise level),
                                                                            $\mathbf{x}_*$ (test input)
2: $L := \text{cholesky}(K + \sigma_n^2 I)$
   $\boldsymbol{\alpha} := L^\top \backslash (L \backslash \mathbf{y})$
                                                   } predictive mean eq. (2.25)
4: $\bar{f}_* := \mathbf{k}_*^\top \boldsymbol{\alpha}$
   $\mathbf{v} := L \backslash \mathbf{k}_*$
                                                   } predictive variance eq. (2.26)
6: $\mathbb{V}[f_*] := k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{v}^\top \mathbf{v}$
   $\log p(\mathbf{y}|X) := -\frac{1}{2}\mathbf{y}^\top \boldsymbol{\alpha} - \sum_i \log L_{ii} - \frac{n}{2} \log 2\pi$    eq. (2.30)
8: return: $\bar{f}_*$ (mean), $\mathbb{V}[f_*]$ (variance), $\log p(\mathbf{y}|X)$ log marginal likelihood
```

Figure 1 - Gaussian Process Regression algorithm by Rasmussen and Williams

3. Experiment Setup

For this experiment, we will make the following assumptions:

1. The robot's heading is always in the +y direction.
2. Sensor readings has no noise
3. The map has no obstacle

In the figure below, z1, z2, z3 and z4 are four sensor readings taken at the robot's location:

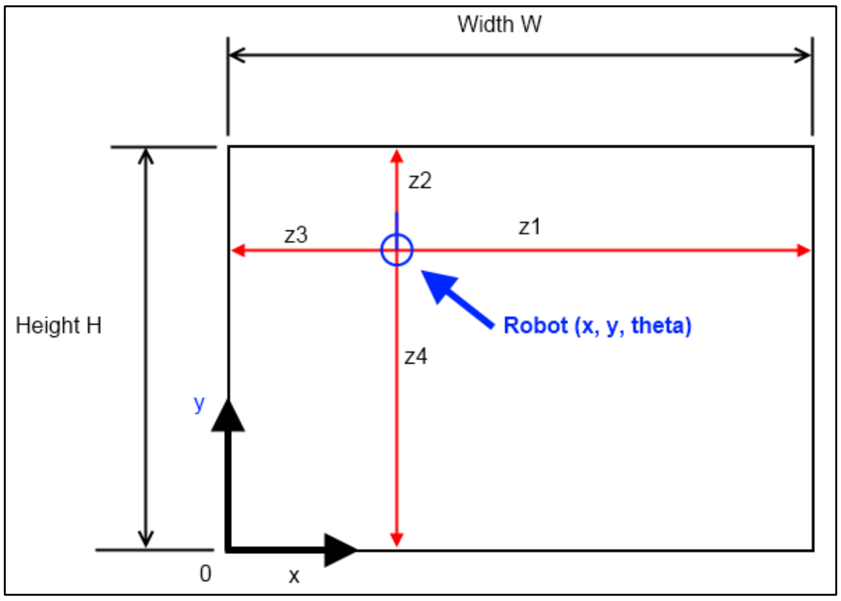


Figure 2 - Schematic of experiment setup

Following assumption 1, theta value will be ignored for now.

The sensor readings can be obtained at any $\vec{\mathbf{x}} = (x, y)$ location on the map, the set of sensor readings $\vec{\mathbf{z}}$ can be obtained as follows:

$$\vec{\boldsymbol{z}} = \begin{bmatrix} z^{(1)} \\ z^{(2)} \\ z^{(3)} \\ z^{(4)} \end{bmatrix} = \begin{bmatrix} W - x \\ H - y \\ x \\ y \end{bmatrix}$$

where W and H are the width and height of the map, respectively

Using the following transformation matrix T, we can map the robot's location to the sensor readings:

$$T\vec{\mathbf{x}} = \vec{\mathbf{z}}$$

$$T\vec{\mathbf{x}} = \begin{bmatrix} -1 & 0 & W \\ 0 & -1 & H \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} z^{(1)} \\ z^{(2)} \\ z^{(3)} \\ z^{(4)} \end{bmatrix}$$

4. Pseudo-codes

For the first part of the experiment, we perform the following steps:

Step 1. First we generate a set of n random robot positions on the map, using **rand(n)**, these robot locations are our inputs

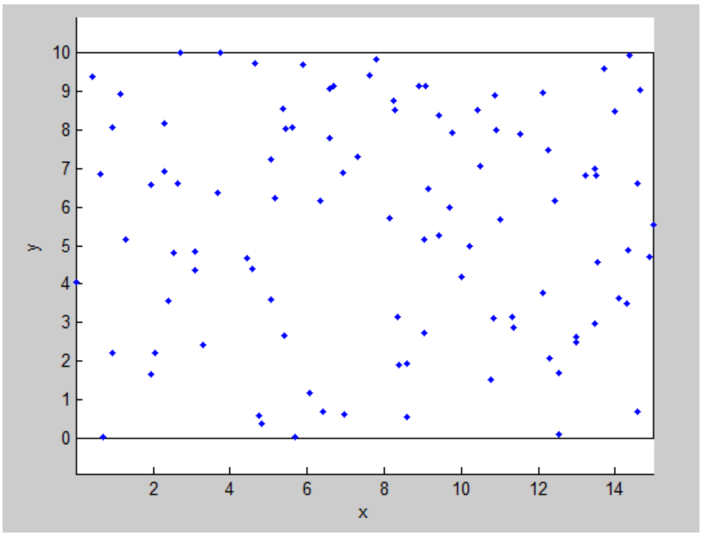


Figure 3 - Randomly generated robot poses on the map

Our design matrix X will be constructed as follows:

$$X = \begin{bmatrix} x_1 & x_2 & & x_n \\ y_1 & y_2 & \dots & y_n \\ 1 & 1 & & 1 \end{bmatrix}$$

Step 2. Next we will collect the targets (sensor readings) at the input locations above. We do so by using the T matrix mentioned earlier:

$$T\vec{\mathbf{x}} = \begin{bmatrix} -1 & 0 & W \\ 0 & -1 & H \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \begin{bmatrix} z^{(1)}{}_i \\ z^{(2)}{}_i \\ z^{(3)}{}_i \\ z^{(4)}{}_i \end{bmatrix}$$

The target vectors are as follows:

$$\mathbf{z}^{(1)} = \begin{bmatrix} z^{(1)}{}_1 & z^{(1)}{}_2 & \dots & z^{(1)}{}_n \end{bmatrix}$$
$$\mathbf{z}^{(2)} = \begin{bmatrix} z^{(2)}{}_1 & z^{(2)}{}_2 & \dots & z^{(2)}{}_n \end{bmatrix}$$
$$\mathbf{z}^{(3)} = \begin{bmatrix} z^{(3)}{}_1 & z^{(3)}{}_2 & \dots & z^{(3)}{}_n \end{bmatrix}$$
$$\mathbf{z}^{(4)} = \begin{bmatrix} z^{(4)}{}_1 & z^{(4)}{}_2 & \dots & z^{(4)}{}_n \end{bmatrix}$$

Step 3. Now that we have the training data, we will use the "Leave-one-out" approach to evaluate the performance of Gaussian Process Regression. We may do so using the function `simple1D_GPPredictor(X, y, hyperparams, x_star)` and pick each input in the training set as our test input.

For the first part of the experiment, we perform the following steps:

Step 1. First we generate a set of n random robot positions on the map, using **rand(n)**, these robot locations are our inputs, as shown in **Figure 3**.

Our target vectors will be constructed as follows:

$$\mathbf{x} = [x_1 \quad x_2 \quad \cdots \quad x_n]$$
$$\mathbf{y} = [y_1 \quad y_2 \quad \cdots \quad y_n]$$

Step 2. Next we will collect the targets (sensor readings) at the input locations above. We do so by using the T matrix mentioned earlier:

$$T\vec{\mathbf{x}} = \begin{bmatrix} -1 & 0 & W \\ 0 & -1 & H \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \begin{bmatrix} z^{(1)}{}_i \\ z^{(2)}{}_i \\ z^{(3)}{}_i \\ z^{(4)}{}_i \end{bmatrix}$$

Our design matrix will be constructed as follows:

$$X = \begin{bmatrix} z^{(1)}{}_1 & z^{(1)}{}_2 & & z^{(1)}{}_n \\ z^{(2)}{}_1 & z^{(2)}{}_2 & \cdots & z^{(2)}{}_n \\ z^{(3)}{}_1 & z^{(3)}{}_2 & & z^{(3)}{}_n \\ z^{(4)}{}_1 & z^{(4)}{}_2 & & z^{(4)}{}_n \end{bmatrix}$$

Step 3. Now that we have the training data, we will use the “Leave-one-out” approach to evaluate the performance of Gaussian Process Regression. We may do so using the function `simple1D_GPPredictor(X, y, hyperparams, x_star)` and pick each input in the training set as our test input.