

STATISTICAL MACHINE LEARNING

ASSIGNMENT 2

Inez Wijnands (s4149696) & Guido Zuidhof (s4160703)

Radboud University Nijmegen

04/11/2015

The entire code listing is in a separate file. The listings shown here are merely code snippets.

1 Sequential learning

1.1 Obtaining the prior

1.

$$\tilde{\Lambda}_{a,b} = \tilde{\Sigma}_{a,b}^{-1} \quad (1.1)$$

$$= \left(\begin{array}{cc|cc} 60 & 50 & -48 & 38 \\ 50 & 50 & -50 & 40 \\ \hline -48 & -50 & 52.4 & -41.4 \\ 38 & 40 & -41.4 & 33.4 \end{array} \right) \quad (1.2)$$

Using the precision matrix $\tilde{\Lambda}$ we can use equations 2.69, 2.73 and 2.75 from Bishop to obtain the mean and covariance of the conditional distribution $p([x_1, x_2]^T | x_3 = x_4 = 0)$.

$$\Sigma_p = \Lambda_{aa}^{-1} \quad (\text{Bishop 2.73})$$

$$\Lambda_{aa} = \begin{pmatrix} 60 & 50 \\ 50 & 50 \end{pmatrix} \quad (1.3)$$

$$\Sigma_p = \begin{pmatrix} 0.1 & -0.1 \\ -0.1 & 0.12 \end{pmatrix} \quad (1.4)$$

$$\mu_p = \mu_{a|b} = \mu_a - \Lambda_{aa}^{-1} \Lambda_{ab} (\mathbf{x}_b - \mu_b) \quad (\text{Bishop 2.75})$$

We can fill in this equation, since $\tilde{\mu}$ and \mathbf{x}_b (the second partition of \mathbf{x}) are known.

$$\mu_p = \begin{pmatrix} 1 \\ 0 \end{pmatrix} - \begin{pmatrix} 60 & 50 \\ 50 & 50 \end{pmatrix}^{-1} \begin{pmatrix} -48 & 38 \\ -50 & 40 \end{pmatrix} \left(\begin{pmatrix} 0 \\ 0 \end{pmatrix} - \begin{pmatrix} 1 \\ 2 \end{pmatrix} \right) \quad (1.5)$$

$$= \begin{pmatrix} 1 \\ 0 \end{pmatrix} - \begin{pmatrix} 0.1 & -0.1 \\ -0.1 & 0.12 \end{pmatrix} \begin{pmatrix} -48 & 38 \\ -50 & 40 \end{pmatrix} \left(\begin{pmatrix} 0 \\ 0 \end{pmatrix} - \begin{pmatrix} 1 \\ 2 \end{pmatrix} \right) \quad (1.6)$$

$$= \begin{pmatrix} 0.8 \\ 0.8 \end{pmatrix} \quad (1.7)$$

2. Using the prior μ_p and Σ_p , we used the numpy-equivalent in Python for the MATLAB-function `mvnrnd` to obtain the μ_t we used for the remainder of this assignment:

```
np.random.multivariate_normal(mu_p, sigma_p, 1)
```

This resulted in:

$$\boldsymbol{\mu}_t = \begin{pmatrix} 0.28584241 \\ 1.42626702 \end{pmatrix} \quad (1.8)$$

3. The probability density is highest at the mean (as illustrated in Figure 1.1). The density decreases quickly as both x and y change, but less so when x XOR y change. In $\boldsymbol{\Sigma}_p$, the values for the x XOR y are lower, so this is consistent with our density plot.

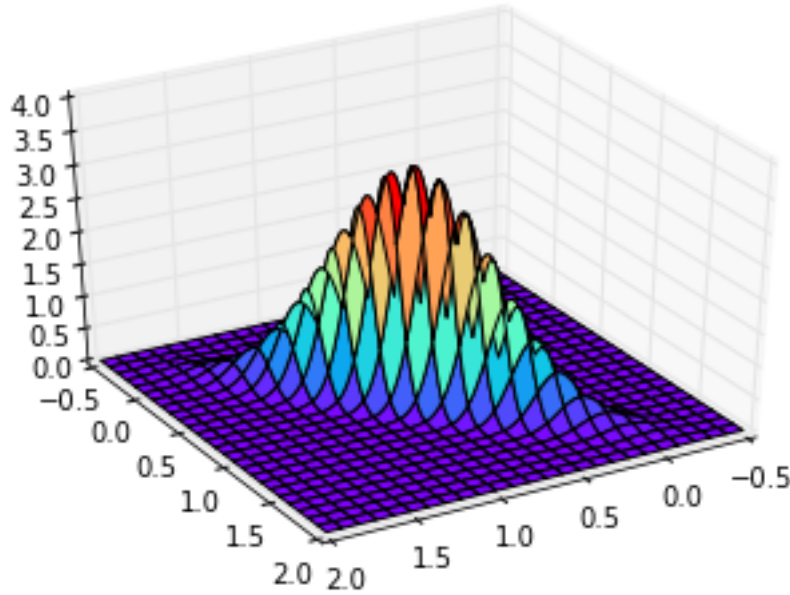


Figure 1.1: The probability density of the distribution.

1.2 Generating the data

1. We used the following function to generate our data:

```
np.random.multivariate_normal(mu_t, sigma_t, 1000)
```

- 2.

$$\boldsymbol{\mu}_{ML} = \frac{1}{N} \sum_{n=1}^N \mathbf{x}_n \quad (\text{Bishop 2.121})$$

$$= \left[\frac{1}{1000} \sum_{n=1}^{1000} \mathbf{x}_n, \frac{1}{1000} \sum_{n=1}^{1000} \mathbf{y}_n \right] \quad (1.9)$$

$$= \begin{pmatrix} 0.25383138 \\ 1.38260838 \end{pmatrix} \quad (1.10)$$

$$\boldsymbol{\Sigma}_{ML} = \frac{1}{N} \sum_{n=1}^N (\mathbf{x}_n - \boldsymbol{\mu}_{ML})(\mathbf{x}_n - \boldsymbol{\mu}_{ML})^T \quad (\text{Bishop 1.22})$$

$$= \begin{pmatrix} \frac{1}{1000} \sum_{n=1}^{1000} (\mathbf{x}_n - \boldsymbol{\mu}_{ML}(1))(\mathbf{x}_n - \boldsymbol{\mu}_{ML}(1))^T & \frac{1}{1000} \sum_{n=1}^{1000} (\mathbf{x}_n - \boldsymbol{\mu}_{ML}(1))(\mathbf{y}_n - \boldsymbol{\mu}_{ML}(2))^T \\ \frac{1}{1000} \sum_{n=1}^{1000} (\mathbf{y}_n - \boldsymbol{\mu}_{ML}(2))(\mathbf{x}_n - \boldsymbol{\mu}_{ML}(1))^T & \frac{1}{1000} \sum_{n=1}^{1000} (\mathbf{y}_n - \boldsymbol{\mu}_{ML}(2))(\mathbf{y}_n - \boldsymbol{\mu}_{ML}(2))^T \end{pmatrix} \quad (1.11)$$

$$= \begin{pmatrix} 1.90513804 & 0.72479489 \\ 0.72479489 & 3.81690496 \end{pmatrix} \quad (1.12)$$

This is calculated using the following code:

```

mu_ml = sum(data)/len(data)

sse = [0,0]
for point in data:
    point = np.matrix(point)
    sse += (point-mu_ml).T*(point-mu_ml)
sigma_ml = sse/len(data)

```

The differences with the 'true' values are:

$$\boldsymbol{\mu}_t - \boldsymbol{\mu}_{ML} = \begin{pmatrix} 0.28584241 \\ 1.42626702 \end{pmatrix} - \begin{pmatrix} 0.25383138 \\ 1.38260838 \end{pmatrix} = \begin{pmatrix} 0.03201103 \\ 0.04365864 \end{pmatrix} \quad (1.13)$$

$$\boldsymbol{\Sigma}_t - \boldsymbol{\Sigma}_{ML} = \begin{pmatrix} 2.0 & 0.8 \\ 0.8 & 4.0 \end{pmatrix} - \begin{pmatrix} 1.90513804 & 0.72479489 \\ 0.72479489 & 3.81690496 \end{pmatrix} = \begin{pmatrix} 0.09486196 & 0.07520511 \\ 0.07520511 & 0.18309504 \end{pmatrix} \quad (1.14)$$

1.3 Sequential learning algorithms

- 1.
- 2.

$$p(\mathbf{x}|D_{n-1}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Lambda}^{-1}) \quad (\text{Bishop 2.113})$$

where: $\mathbf{x} = \boldsymbol{\mu}, \boldsymbol{\mu} = \boldsymbol{\mu}_{(n-1)}, \boldsymbol{\Lambda}^{-1} = \boldsymbol{\Sigma}_{(n-1)}$

$$p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{A}\mathbf{x} + \mathbf{b}, \mathbf{L}^{-1}) \quad (\text{Bishop 2.114})$$

where $\mathbf{y} = \mathbf{x}_n, \mathbf{A} = \mathbf{I}, \mathbf{x} = \boldsymbol{\mu}, \mathbf{b} = 0, \mathbf{L}^{-1} = \boldsymbol{\Sigma}^t$

$$p(\mathbf{x}|\mathbf{y}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\Sigma}\{\mathbf{A}^T \mathbf{L}(\mathbf{y} - \mathbf{b}) + \boldsymbol{\Lambda}\boldsymbol{\mu}\}, \boldsymbol{\Sigma}) \quad (\text{Bishop 2.116})$$

$$\boldsymbol{\Sigma} = (\boldsymbol{\Lambda} + \mathbf{A}^T \mathbf{L} \mathbf{A})^{-1} \quad (\text{Bishop 2.117})$$

Matching the variables we get the following equations:

$$p(\boldsymbol{\mu}|\mathbf{x}_n) = \mathcal{N}(\boldsymbol{\mu}|\mathbf{S}\{\mathbf{I}^T \boldsymbol{\Sigma}_t^{-1}(\mathbf{x}_n - 0) + \boldsymbol{\Sigma}_{(n-1)}^{-1}\}, \mathbf{S}) \quad (1.15)$$

$$= \mathcal{N}(\boldsymbol{\mu}|\mathbf{S}\{\mathbf{I}^T \boldsymbol{\Sigma}_t^{-1} \mathbf{x}_n + \boldsymbol{\Sigma}_{(n-1)}^{-1}\}, \mathbf{S}) \quad (1.16)$$

$$= \mathcal{N}(\boldsymbol{\mu}|\mathbf{S}\{\boldsymbol{\Sigma}_t^{-1} \mathbf{x}_n + \boldsymbol{\Sigma}_{(n-1)}^{-1}\}, \mathbf{S}) \quad (1.17)$$

$$\mathbf{S} = (\boldsymbol{\Sigma}_{(n-1)}^{-1} + \mathbf{I}^T \boldsymbol{\Sigma}_t^{-1} \mathbf{I})^{-1} \quad (1.18)$$

$$= (\boldsymbol{\Sigma}_{(n-1)}^{-1} + \boldsymbol{\Sigma}_t^{-1})^{-1} \quad (1.19)$$

$\boldsymbol{\mu}_n$ is the mean of the distribution $p(\boldsymbol{\mu}|\mathbf{x}_n)$, so the functions we use for our sequential learning algorithm are:

$$\boldsymbol{\Sigma}_n = \mathbf{S} \quad (1.20)$$

$$\boldsymbol{\mu}_n = \boldsymbol{\Sigma}_n\{\boldsymbol{\Sigma}_t^{-1} \mathbf{x}_n + \boldsymbol{\Sigma}_{n-1}^{-1}\}, \boldsymbol{\Sigma}_n) \quad (1.21)$$

- 3.
- 4.

2 The faulty lighthouse

2.1 Constructing the model

- 1.
- 2.
- 3.
- 4.

2.2 Generate the lighthouse data

- 1.
- 2.

2.3 Find the lighthouse

- 1.
- 2.
- 3.