Multivariate Random Variables

Rein van den Boomgaard

April 11, 2012

1 Random Variables

- A random variable X is a numerical observation of a random experiment.
- In case the outcome of X is a countable set of values (say $X \in \mathbb{Z}$), the rv is *discrete* and the *probability mass function* p_X is a mapping from $\mathbb{Z}$ to $[0,1] \subset \mathbb{R}$.

  Note that $\sum_{k\in\mathbb{Z}} p_X(k) = 1$.
- In case $X \in \mathbb{R}$ the rv is *continuous* and the *probability density function* p_X is mapping from $\mathbb{R}$ to $\mathbb{R}_+$ (i.e. the set of positive reals).

2 Multivariate Random Variables

- Multivariate pmf $p_{XYZ}(x,y,z)$, $\sum_x \sum_y \sum_z p_{XYZ}(x,y,z) = 1$
- Multivariate pdf $p_{XYZ}(x,y,z)$, $\iiint_{x,y,z} p_{XYZ}(x,y,z)dxdydz = 1$

3 Multivariate Integrals

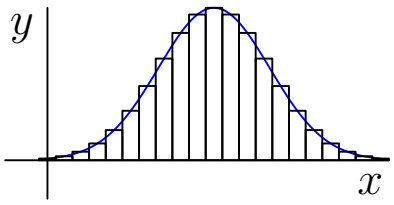

Figure 1: One dimensional integral as Riemann sum

- Scalar ($n = 1$)

$$\int_{-\infty}^{+\infty} f(x)\, dx$$

  calculates the area under the graph of the function. Here dx is the length of the infinitesimal interval and $f(x)\, dx$ is the area under the curve in the interval dx.
- Multivariate $n = 2$

$$\int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(x,y)\, dx\, dy$$

  calculates the volume under the graph of the function. Here $dx\, dy$ is the area of the infinitesimal area and $f(x,y)dxdy$ is the volume under the graph in that area.

- Multivariate $n = 3$

$$\int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(x, y, z)\, dx\, dy\, dz$$

calculates the *hypervolume* under the graph of the function. Here $dxdydz$ is the infinitesimal volume and $f(x, y, z)dxdydz$ is the hypervolume.

- Multivariate

4 Vector Random Variables

- Collect the random variables in a vector $\mathbf{X} = (X_1 \cdots X_n)^\intercal$
- Vectorial pmf or pdf: $p_{\mathbf{X}}(\mathbf{x}) = p_{X_1,\ldots,X_n}(x_1, \ldots, x_n)$
- For the pmf: $\sum_{\mathbf{x}\in\mathbb{R}^n} p_{\mathbf{X}}(\mathbf{x}) = 1$
- For the pdf: $\int_{\mathbf{x}\in\mathbb{R}^n} p_{\mathbf{X}}(\mathbf{x})d\mathbf{x} = 1$

5 Expectation

- Discrete $\mathbf{X}$: $E(\mathbf{X}) = \sum_{\mathbf{x}\in\mathbb{R}^n} \mathbf{x}\, p_{\mathbf{X}}(\mathbf{x})$
- Continuous $\mathbf{X}$: $E(\mathbf{X}) = \int_{\mathbf{x}\in\mathbb{R}^n} \mathbf{x}\, p_{\mathbf{X}}(\mathbf{x})\, d\mathbf{x}$
  - Please note that the above is a shorthand notation for:

$$E(\mathbf{X}) = \int_{\mathbf{x}\in\mathbb{R}^n} \mathbf{x}\, p_{\mathbf{X}}(\mathbf{x})\, d\mathbf{x} = \begin{pmatrix} \int_{x_1} \cdots \int_{x_n} x_1 p_{\mathbf{X}}(\mathbf{x}) dx_1 \cdots dx_n \\ \vdots \\ \int_{x_1} \cdots \int_{x_n} x_n p_{\mathbf{X}}(\mathbf{x}) dx_1 \cdots dx_n \end{pmatrix}$$

  - The first element of $E(\mathbf{X})$ as given above can be rewritten as:

$$\int_{x_1} \cdots \int_{x_n} x_1 p_{\mathbf{X}}(\mathbf{x}) dx_1 \cdots dx_n = \int_{x_1} x_1 \left(\int_{x_2} \cdots \int_{x_n} p_{\mathbf{X}}(\mathbf{x}) dx_2 \cdots dx_n \right) dx_1 = \int_{x_1} p_{X_1}(x_1) dx_1 = E(X_1)$$

- From the above it follows that: $E(\mathbf{X}) = E((X_1 \cdots X_n)^\intercal) = (E(X_1) \cdots E(X_n))^\intercal$, i.e. the expection of a vector stochast is itself a vector and the elements are the expectations of the scalar elements.

6 Expectation

- Let $\mathbf{X}$ and $\mathbf{Y}$ be two vector rv's, then $E(\alpha\mathbf{X} + \beta\mathbf{Y}) = \alpha E(\mathbf{X}) + \beta E(\mathbf{Y})$ (note that this is true independent of the distributions of the rv's and of the fact whether they are dependent or not).
- Let A be $m \times n$ matrix then $E(A\mathbf{X}) = AE(\mathbf{X})$
  - Can your prove this? Hint: write out $E(A\mathbf{X})$ in its elements and make use of the fact that $E(a_1X_1 + \cdots + a_nX_n) = a_1E(X_1) + \cdots + a_nE(X_n)$.

7 Variance for 2 rv's

- The variance for a scalar rv:

$$\mathrm{Var}(X) = E((X - E(X))^2)$$

is a measure for the expected (squared) deviation from the mean.

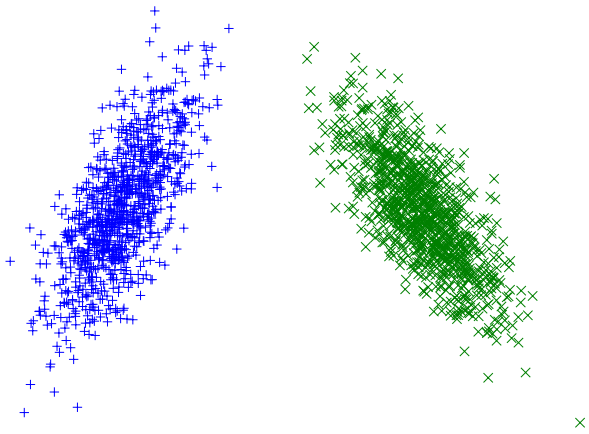

Figure 2: Scatter plot of two distributions (differing only by a rotation)

- In case of a 2d vector $\mathbf{X} = (X_1\ X_2)^\intercal$ the variance cannot be captured in one number. The spread of the distribution $p_{\mathbf{X}}$ is not only determined by the average distance from the mean, also the 'shape' of the distribution needs to be captured with the notion of the variance.

- A measure like $E(\|\mathbf{X} - E(\mathbf{X})\|^2)$ does capture the spread around the mean but it cannot capture the difference between the two distributions above (we have plotted 1000 samples from each of the distributions).
  - Can you prove that this spread measure equals the sum of the variances of the elements of vector $\mathbf{X}$?

8 Directional Variance

- To capture the orientation of the distribution we look at the scalar rv $Y = \mathbf{r}^\intercal\mathbf{X}$ where $\mathbf{r}$ is a unit vector (i.e. a direction vector with unit length $\|\mathbf{r}\| = 1$).
- Note that Y is the projection of $\mathbf{X}$ on a line in the $\mathbf{r}$-direction. Therefore the variance $\mathrm{Var}(Y)$ captures the spread of the distribution of $\mathbf{X}$ in the $\mathbf{r}$-direction.
- We have:
$$E(Y) = E(\mathbf{r}^\intercal\mathbf{X}) = \mathbf{r}^\intercal E(\mathbf{X})$$
  - This follows from the fact that $E(A\mathbf{X}) = AE(\mathbf{X})$.
- For the variance of Y we have
$$\begin{aligned}\mathrm{Var}(Y) &= E((Y - E(Y))^2)\\ &= E((\mathbf{r}^\intercal\mathbf{X} - \mathbf{r}^\intercal E(\mathbf{X}))^2)\\ &= E((\mathbf{r}^\intercal(\mathbf{X} - E(\mathbf{X}))^2)\\ &= E(\mathbf{r}^\intercal(\mathbf{X} - E(\mathbf{X}))(\mathbf{X} - E(\mathbf{X}))^\intercal\mathbf{r})\\ &= \mathbf{r}^\intercal E((\mathbf{X} - E(\mathbf{X}))(\mathbf{X} - E(\mathbf{X}))^\intercal)\mathbf{r}\end{aligned}$$
- Note that $E((\mathbf{X} - E(\mathbf{X}))(\mathbf{X} - E(\mathbf{X}))^\intercal)$ is an $n \times n$ matrix.

9 Covariance

- The *covariance matrix* is defined as:

$$\mathrm{Cov}(\mathbf{X}) = E\left((\mathbf{X} - E(\mathbf{X})(\mathbf{X} - E(\mathbf{X})^\intercal\right)$$

- We will often write $\mathbf{C_X} = \mathrm{Cov}(\mathbf{X})$ or simply $\mathbf{C}$ in case the random variable is obvious from the context.
- It captures the variance of $\mathbf{X}$ in any directions: $\mathrm{Var}(\mathbf{r}^\intercal\mathbf{X}) = \mathbf{r}^\intercal C_\mathbf{X}\mathbf{r}$.
- Note that the diagonal elements of $\mathrm{Cov}(\mathbf{X})$ are the variances of the elements of $\mathbf{X}$.

10 Covariance

- Let $\mathbf{C} = \mathrm{Cov}(\mathbf{X})$ then:

$$C_{ij} = \mathrm{Cov}(X_i, X_j) = E((X_i - E(X_i))(X_j - E(X_j)))$$

- Thus:

$$\mathrm{Cov}(\mathbf{X}) = \begin{pmatrix} Cov(X_1, X_1) & \cdots & \mathrm{Cov}(X_1, X_n) \\ \vdots & & \vdots \\ Cov(X_n, X_1) & \cdots & \mathrm{Cov}(X_n, X_n) \end{pmatrix}$$

- Let X, Y and Z be scalar rv's, then:

$$\mathrm{Cov}(aX + bY, Z) = a\,\mathrm{Cov}(X, Z) + b\,\mathrm{Cov}(Y, Z)$$

  This follows directly from the definition and properties of the expectation.

- Let A be a $m \times n$ matrix and let $\mathbf{X}$ be a vectorial rv, then $\mathrm{Cov}(A\mathbf{X}) = A\,\mathrm{Cov}(\mathbf{X})A^\intercal$. The proof is a direct consequence of the property above (and a lot of tedious algebraic manipulation).

11 Exercise: Multivariate Stochasten (Multivariate Random Variables)

Onderstaand is een opgave uit een eerder tentamen statistisch redeneren.

Een multivariate stochast (of vector stochast) is een vector van scalaire (reeel waardige) stochasten $\mathbf{X} = (X_1 \cdots X_n)^\intercal$.

1. Wat is de verwachting $E(\mathbf{X})$ uitgedrukt in de verwachting van de scalaire stochasten?
2. Waarom kunnen we de variantie van een vectoriele stochast niet vastleggen met een scalar? Geef een (2 dimensionaal) voorbeeld waaruit dat blijkt.
3. De covariantie van twee scalaire stochasten X en Y is gegeven als $\mathrm{Cov}(X, Y) = E((X - E(X))(Y - E(Y)))$. Geef de definitie van de covariantie matrix $\mathrm{Cov}(\mathbf{X})$ in termen van de covariantie van de scalaire stochasten. Wat zijn de afmetingen van deze matrix?
4. (*) Kunt u de uitdrukking voor de verwachting van een vectoriele stochast ook bewijzen uit de definitie (voor een discrete stochast):

$$E(\mathbf{X}) = \sum_{\mathbf{x}} \mathbf{x}\, p_\mathbf{X}(\mathbf{x})$$

5. (*) Gegeven een vectoriele stochast $\mathbf{X} = (X\ Y)^\intercal$ en scalaire 2×2 matrix A. Bewijs dat $\mathrm{Cov}(A\mathbf{X}) = A\,\mathrm{Cov}(\mathbf{X})A^\intercal$. Hint: schrijf $A\mathbf{X}$ uit in componenten. Maak gebruik van de eigenschap dat $\mathrm{Cov}(aX, bY) = ab\,\mathrm{Cov}(X, Y)$ en dat $\mathrm{Cov}(X + Y, Z) = \mathrm{Cov}(X, Z) + \mathrm{Cov}(Y, Z)$.

12 LabExercise: Estimating mean and covariance from sample (steekproef)

Assume a vectorial random variable $\mathbf{X} = (X_1 \cdots X_n)^\intercal$. The probability density function $p_{\mathbf{X}}$ is unknown. Without proof we state that given N observations $\mathbf{x}_1, \ldots, \mathbf{x}_N$ an estimate for the expectation $E(\mathbf{X})$ is the vectorial mean:

$$\bar{\mathbf{x}} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{x}_i$$

and an estimate for the covariance matrix is obtained as:

$$S = \frac{1}{N-1} \sum_{i=1}^{N} (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^\intercal$$

1. Write Python functions to calculate the mean and covariance matrix of N vectors. Assume that those N vectors are assembled in one data matrix where each column is data vector. Please note that loops are not needed and that you can do it easily in three lines of code.
2. Calculate mean and covariance for the Iris dataset. Do it for the entire collection of datavectors (150 in all) and calculate the mean and covariance for the 3 classes seperately.
3. Redo the plot of the iris data set and plot the means of the entire set and of the three classes.
4. (*) more difficult but nice: visualize the covariance matrix! A neat way to do it is to calculate $\sqrt{\mathbf{r}^\intercal \mathbf{C} \mathbf{r}}$ (i.e. the standard deviation in $\mathbf{r}$ direction). When you are plotting the first coordinate against the second. Take $\mathbf{r} = (\cos\phi \ \sin\phi \ 0 \ 0)^\intercal$ for ϕ in the interval from 0 to 2π and plot $\mathbf{x} + (\mathbf{r}^\intercal \mathbf{C} \mathbf{r})\mathbf{r}$. This will plot an ellipse centered at the mean and 'adapted' to the shape of the dataset.

13 Correlated Random Variables

- Two rv's X_1 and X_2 are *uncorrelated* in case $\mathrm{Cov}(X_1, X_2) = 0$.
- Consider the random vector $\mathbf{X} = (X_1 \ X_d)^\intercal$. In case all elements are uncorrelated we have that the covariance matrix is a diagonal matrix.
- In case X and Y are independent rv's, the rv's are also uncorrelated: $X \perp Y \rightarrow \mathrm{Cov}(X, Y) = 0$.
- The reverse is not nescessarily true! (i.e. the fact that X and Y are uncorrelated doesn't imply that X and Y are independent).

14 The Normal Distribution

- For a scalar rv X that is normally distributed the pdf is defined as:

$$p_X(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

where μ is the expectation: $E(X) = \mu$, and σ^2 is the variance $\mathrm{Var}(X) = \sigma^2$.

- Now consider n independent normally distributed rv's $\mathbf{X} = (X_1 \cdots X_n)^\intercal$ where $X_i \sim \mathrm{N}(\mu_i, \sigma_i^2)$. Because of the independence we have:

$$\begin{aligned}
p_{\mathbf{X}}(\mathbf{x}) &= p_{X_1}(x_1) \cdots p_{X_n}(x_n) \\
&= \frac{1}{\sigma_1\sqrt{2\pi}} e^{-\frac{(x_1-\mu_1)^2}{2\sigma_1^2}} \cdots \frac{1}{\sigma_n\sqrt{2\pi}} e^{-\frac{(x_n-\mu_n)^2}{2\sigma_n^2}} \\
&= \frac{1}{\sigma_1 \cdots \sigma_n(\sqrt{2\pi})^n} e^{-\frac{(x_1-\mu_1)^2}{2\sigma_1^2}} \cdots e^{-\frac{(x_n-\mu_n)^2}{2\sigma_n^2}} \\
&= \frac{1}{\sigma_1 \cdots \sigma_n(\sqrt{2\pi})^n} e^{-\frac{1}{2}\left(\left(\frac{(x_1-\mu_1)}{\sigma_1}\right)^2 + \cdots + \left(\frac{(x_1-\mu_1)}{\sigma_1}\right)^2\right)} \\
&= \frac{1}{\sigma_1 \cdots \sigma_n(\sqrt{2\pi})^n} e^{-\frac{1}{2}(\mathbf{x}-\mu)^\intercal \mathbf{\Sigma}^{-1}(\mathbf{x}-\mu)} \\
&= \frac{1}{|\mathbf{\Sigma}|^{-1/2}(2\pi)^{n/2}} e^{-\frac{1}{2}(\mathbf{x}-\mu)^\intercal \mathbf{\Sigma}^{-1}(\mathbf{x}-\mu)}
\end{aligned}$$

  where:

$$\mu = \begin{pmatrix} \mu_1 \\ \vdots \\ \mu_n \end{pmatrix} \quad \text{and} \quad \mathbf{\Sigma} = \begin{pmatrix} \sigma_1^2 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & \cdots & 0 \\ \vdots & & \cdots & \vdots \\ 0 & & \cdots & \sigma_n^2 \end{pmatrix}$$

15 The Normal Distribution

- A vector rv $\mathbf{X}$ is normally distributed $\mathbf{X} \sim \mathrm{N}(\mu, \mathbf{\Sigma})$ with:

$$p_{\mathbf{X}}(\mathbf{x}) = \frac{1}{|\mathbf{\Sigma}|^{-1/2}(2\pi)^{n/2}} e^{-\frac{1}{2}(\mathbf{x}-\mu)^\intercal \mathbf{\Sigma}^{-1}(\mathbf{x}-\mu)}$$

  where $\mathbf{\Sigma} = \mathrm{Cov}(\mathbf{X})$ is the covariance matrix, that is not necessarily a diagonal matrix.

- For the normal distributed vector rv $X \sim \mathrm{N}(\mu, \mathbf{\Sigma})$ we have: $E(\mathbf{X}) = \mu$ and $\mathrm{Cov}(\mathbf{X}) = \mathbf{\Sigma}$.

16 Transforming Normal Distributed RV's

- Let $\mathbf{X} \sim \mathrm{N}(\mu, \mathbf{\Sigma})$ and let $\mathbf{Y} = A\mathbf{X} + \mathbf{b}$ where A is an $m \times n$ matrix and $\mathbf{b}$ is an m dimensional vector, then $\mathbf{Y} \sim \mathrm{N}(A\mu + \mathbf{b}, A\mathbf{\Sigma}A^\intercal)$

17 Generating Samples from $\mathrm{N}(\mu, \mathbf{\Sigma})$

- Assume we have a RNG to generate samples from the scalar normal distribution $\mathrm{N}(0, 1)$. Assume these samples are independent.
- Let $\mathbf{\Sigma} = U\Lambda U^\intercal$ be the spectral decomposition of $\mathbf{\Sigma}$ (i.e. the eigenvalue/eigenvector decomposition).
- Construct a rv $\mathbf{X}$ where each element is iid with $\mathrm{N}(0, 1)$
- Let $A = U\sqrt{\Lambda}$ then $\mathbf{Y} = A\mathbf{X} + \mu \sim \mathrm{N}(\mu, \Sigma)$.

18 Python Code for Generating Samples from $\mathrm{N}(\mu, \Sigma)$

```
from pylab import *
mu = array( [\footnote{DEFINITION NOT FOUND: 5 },\footnote{DEFINITION NOT FOUND: 6 }] )
Sigma = array( [[2, 1],[1, 2]] )  # 2x2 required covariance matrix
d, U = eig(Sigma)                 # Sigma = U L Ut
L = diagflat(d)
A = dot(U, sqrt(L))               # required transform matrix
X = randn(2,1000)                 # 2x1000 matrix with each element ~ N(0,1)
Y = dot(A,X)+tile(mu,1000)        # 2x1000 each column vector ~N(mu, Sigma)
figure(1)
clf()
plot(X\footnote{DEFINITION NOT FOUND: 0 },X\footnote{DEFINITION NOT FOUND: 1 },'+b')
plot(Y\footnotemark[1],Y\footnotemark[2],'xg')              # scatter plot of N(mu,Sigma) distri
axis('equal')
savefig('figures/normalscatter2dtest.pdf')
```

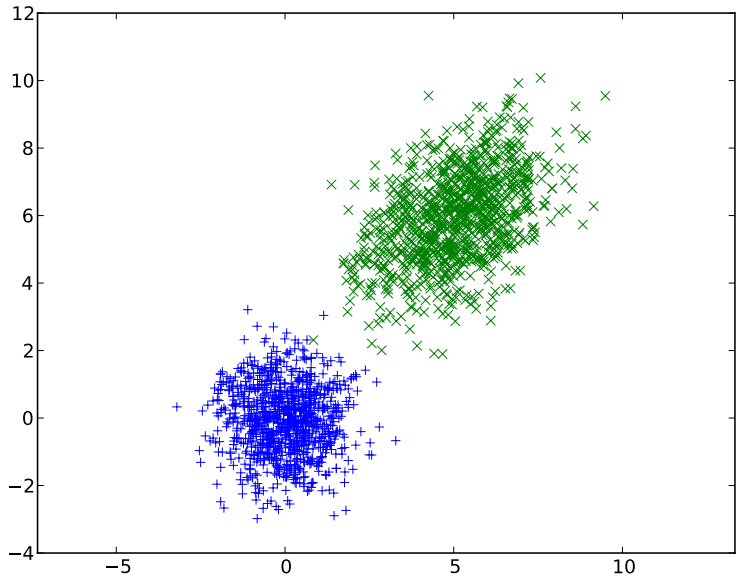

Figure 3: Samples from $\mathrm{N}(0, I)$ (blue) and $\mathrm{N}(\mu, \Sigma)$ (green).

19 Estimating the parameters

- Without proof we state that unbiased estimators for the expectation μ and covariance matrix Σ are:

$$\hat{\mu} = \bar{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i$$

and

$$\hat{\Sigma} = S = \frac{1}{n-1} \sum_{i=1}^{n} (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^\mathsf{T}$$

20 Python Code for Estimation of μ and Σ

```
from pylab import *
# First generate data set with known mu and Sigma
mu = array( [\footnotemark[3],\footnotemark[4]] )           # column vector of required mean
Sigma = array( [[2, 1],[1, 2]] )  # 2x2 required covariance matrix
d, U = eig(Sigma)                 # Sigma = U L Ut
L = diagflat(d)
A = dot(U, sqrt(L))               # required transform matrix
X = randn(2,1000)                 # 2x1000 matrix with each element ~ N(0,1)
Y = dot(A,X)+tile(mu,1000)        # 2x1000 each column vector ~N(mu, Sigma)

Ybar = mean(Y,1)                  # mean along the 1 axis (is second axis..)
Yzm = Y - tile(Ybar[:,newaxis],1000) # subtract mean from each column
S = dot(Yzm, transpose(Yzm)) / 999   # estimator for covariance matrix
print("Ybar"); print(Ybar)
print("S"); print(S)
```

```
 Ybar
 [ 4.91036589  5.87486002]
 S
 [[ 1.96216933  0.96807611]
  [ 0.96807611  2.01533553]]
```

Some remarks about the code:

- Matrix multiplication in Numpy is done with the `dot` function when dealing with `arrays`. You can use `matrix` instead of `array` to have `*` be the matrix multiplication (i do not recommend this).
- With the function call `randn(2,1000)` we make an array of 2x1000 numbers, each iid from the standard normal distribution.
- With `dot(A,X)` we transform the rv `X` to have the required covariance.
- With `dot(A,X)+tile(mu,1000)` we add the required mean to all column vectors in `X`. With `tile(mu,1000)` we repeat the mu vector 1000 times to obtain an array of size 2x1000. This is typical in languages that have arrays as basis datatypes. Instead of writing (for) loops we first make (large) arrays and then use array operators to combine these. Obviously the loops are still there but hidden in optimized code and not in interpreted (and slow) code.

21 Exercise: Transforming multivariate rv's

(This exercise is taken from exercises out of the reader written by Bert van Es)

Let $\mathbf{X} = (X_1\ X_2)^\intercal$ be a random vector with expectation μ and covariance matrix Σ given by:

$$\mu = \begin{pmatrix} 1 \\ -1 \end{pmatrix} \quad \text{and} \quad \Sigma = \begin{pmatrix} 1 & 0 \\ 0 & 4 \end{pmatrix}$$

Calculate the expectation and covariance of

$$\mathbf{Z} = \begin{pmatrix} Z_1 \\ Z_2 \end{pmatrix} = \begin{pmatrix} X_1 + X_2 \\ X_1 - X_2 \end{pmatrix}$$

Can we conclude that Z_1 and Z_2 are independent?

22 Exercise: Correlation vs Dependence

(This exercise is taken from exercises out of the reader written by Bert van Es)

Let $\mathbf{X} = (X_1\ X_2)^\intercal$ be a random vector with expectation $(0\ 0)^\intercal$ (a zero mean random vector) and covariance matrix

$$\Sigma = \begin{pmatrix} 1 & 0 \\ 0 & c \end{pmatrix}$$

for $c > 0$. For which values of c are the components Z_1 and Z_2 uncorrelated.

23 LabExercise: Generating & Visualizing Samples from $\mathrm{N}(\mu, \Sigma)$

- Generate a data matrix where each column is a 4-dimensional vector $\mathbf{x} = (x_1\ x_2\ x_3\ x_4)^\intercal$ drawn from a multivariate Normal distribution with mean μ and covariance matrix Σ. Look at the code in these handouts for inspiration.
- Make a figure with 12 scatter plots aranged in a 4×4 matrix. At the i, j position (indexed as in a matrix) draw the scatter plot of x_i versus x_j. Evidently at the i, i positions there are no interesting scatterplots to draw. This is a well known way to get a visual overview of multidimensional data sets. Using `matplotlib` you will need the `subplot` function to plot several axes into one figure.

24 LabExercise: Estimating mean ($\bar{x}$) and covariance matrix (S)

- From a sample (the datamatrix from the exercise above) calculate the estimators for the mean and covariance. Experiment to observe that the accuracy of the estimators is dependent on N: the number of vectors in our sample.
- Repeat the above estimation of μ (for fixed n) and collect all these in a data matrix. For these estimations you can calculate the covariance matrix as well. Is your result in accordance with the theory?

25 LabExercise: Scatter Plots of Iris Dataset

In one of the following exercises in this course we will need a well known data set. The data set is downloaded from the UCI Machine Learning Repository http://archive.ics.uci.edu/ml/. To quote from the site:

"This is perhaps the best known database to be found in the pattern recognition literature. Fisher's paper is a classic in the field and is referenced frequently to this day. (See Duda & Hart, for example.) The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant. One class is linearly separable from the other 2; the latter are NOT linearly separable from each other."

Of 150 Iris flowers, 50 of each of 3 classes of Iris's, biologists have measured "SepalLength", "SepalWidth", "PetalLength" and "PetalWidth". In the datafile there are 150 lines. Each line contains the four feature measurements separated by a comma and as last item on the line the class of the Iris.

In Python you can use the function `loadtxt` to read the data from the file file:data/iris.data. As we would like to have the data in a Numpy matrix (a homogeneous data type, meaning that all elements in the array have the same datatype) we have to convert the class label (in the fifth column) into a numerical label. The `loadtxt` function provides an elegant mechanism to accomplish such column specific dataconversion.

Define the function `cnvt` as in

```
def cnvt(s):
    tab = {'Iris-setosa':1.0, 'Iris-versicolor':2.0, 'Iris-virginica':3.0}
    if tab.has_key(s):
        return tab[s]
    else:
```

```
        return -1.0
```

then the call

```
data = loadtxt('data/iris.data', delimiter=',', dtype=float, converters={4: cnvt})
```

will load the Iris dataset as an 150x5 array of floats. Now the last column encodes the class with a number.

Write a Python function to make a figure where in an 4x4 matrix, 12 scatterplots are shown. On the i, j position (indexing as in a matrix) the scatter plot of the i-th variable on the horizontal axis and the j-th variable on the vertical axis. The diagonal elements need not be plotted. Make sure that each of 3 different classes are either drawn in different colors or with different symbols.

Based on the scatterplot, would you think it is doable to write a function that will classify an unknown feature vector to belong to one of the three classes?