ejercicio10

April 2, 2020

1 Ejercicio 10:

El vector X es gaussiano de media nula y matriz de covarianza:

$$C_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2.5 & -0.5 \\ 0 & -0.5 & 2.5 \end{bmatrix}$$

1. Obtenga N = 10000 realizaciones del vector X. Estime la matriz de covarianza y compare el resultado con Cx.
2. Obtenga la matriz A tal que Z = AX tenga componentes descorrelacionadas. Esta transformación “blanquea” las componentes de X.

1.1 Vector Aleatorio Gaussiano

Decimos que un vector aleatorio $\underline{X} \in \mathbb{R}^n$ tiene distribución gaussiana de media $\mu_{\underline{X}}$ y matriz de covarianza $C_{\underline{X}}$: $\underline{X} \sim \mathcal{N}(\mu_{\underline{x}}, C_{\underline{X}})$, si su función de densidad de probabilidad (fdp) conjunta es:

$$f_{\underline{X}}(\underline{x}) = \frac{1}{(2\pi)^{n/2}\,|C_{\underline{X}}|^{1/2}} \exp\left(-\frac{1}{2}\,(\underline{x}-\mu_{\underline{x}})^T\ C_{\underline{X}}^{-1}\ (\underline{x}-\mu_{\underline{x}})\right)$$

Decimos que $\underline{X}$ es no degenerado si $\exists\, C_{\underline{X}}^{-1}$.

Por otro lado, dado un vector gaussiano $\underline{X} \sim \mathcal{N}(\mu_{\underline{X}}, C_{\underline{X}})$, si le aplicamos una transformación: $\underline{Y} = A\underline{X} + \underline{b}$ obtenemos otro vector gaussiano: $\underline{Y} \sim \mathcal{N}(\mu_{\underline{Y}}, C_{\underline{Y}})$ cuyo vector de medias y matriz de covarianza se obtienen como:

$$\begin{aligned}
\mu_{\underline{Y}} &= \mathbb{E}\left[\underline{Y}\right] = A\,\mathbb{E}\left[\underline{X}\right] + \underline{b} \\
&= A\,\mu_{\underline{X}} + \underline{b} \\
C_{\underline{Y}} &= \mathbb{E}\left[(\underline{Y}-\mu_{\underline{Y}})\,(\underline{Y}-\mu_{\underline{Y}})^T\right] \\
&= \mathbb{E}\left[(A\underline{X}+\underline{b}-A\,\mu_{\underline{X}}-\underline{b})\,(A\underline{X}+\underline{b}-A\,\mu_{\underline{X}}-\underline{b})^T\right] \\
&= \mathbb{E}\left[(A\underline{X}-A\,\mu_{\underline{X}})\,(A\underline{X}-A\,\mu_{\underline{X}})^T\right] \\
&= A\,\mathbb{E}\left[(\underline{X}-\ \mu_{\underline{X}})\,(\underline{X}-\mu_{\underline{X}})^T\right]\ A^T \\
&= A\,C_{\underline{X}}\,A^T
\end{aligned}$$

Podemos usar esta última propiedad para:

- **Colorear** un vector gaussiano: a partir de un vector cuyas componentes son gaussianas descorrelacionadas: $\underline{Z} \sim \mathcal{N}(\underline{0}, \Lambda)$ con Λ diagonal, obtener otro vector gaussiano $\underline{X} \sim \mathcal{N}(\mu_{\underline{X}}, C_{\underline{X}})$.
- **Blanquear** un vector gaussiano: a partir de un vector gaussiano: $\underline{Y} \sim \mathcal{N}(\mu_{\underline{Y}}, C_{\underline{Y}})$ hallar otro vector gaussiano cuyas componentes estén descorrelacionadas $\mathcal{N}(\underline{0}, \Lambda)$ donde Λ es una matriz diagonal. Esto último en caso gaussiano corresponde a que las componentes del vector sean independientes.

En el Ejercicio 7 vimos la Trasnsformada de Box-Muller que nos permite generar de forma sencilla realizaciones de un vector gaussiano en $\mathbb{R}^2$: $\mathcal{N}(\underline{0}_{2\times 1}, I_{2\times 2})$ a partir de realizaciones de 2 variabes aleatorias uniformes. Coloreando ese vector podemos además generar realizaciones de un vector $\underline{X} \sim \mathcal{N}(\mu_{\underline{X}}, C_{\underline{X}})$.

En cualquiera de estos dos casos: £cómo obtenemos A a partir de una dada $C_{\underline{X}}$?

Usando la propiedad anterior, podemos obtener A factorizando la matriz $C_{\underline{X}}$ que por ser matriz de covarianza es simétrica y (semi) definida positiva. Vamos a ver 3 métodos para hacerlo.

1.1.1 Método 1

Diagonalicemos la matriz de covarianza $C_{\underline{X}}$:

$$C_{\underline{X}} = P\,D\,P^{-1}$$

Por propiedades de la matriz de covarianza sabemos que $C_{\underline{X}}$ es una matriz simétrica y por lo tanto la matriz de autovectores es ortogonal (unitaria) y cumple: $P^{-1} = P^T$. Entonces:

$$C_{\underline{X}} = P\,D\,P^T$$

Por otro lado, D es una matriz diagonal y entonces podemos descomponerla como: $D = D^{1/2}\,D^{1/2}$. Asimismo, $D^{1/2}$ también es una matriz diagonal y por lo tanto es simétrica. Con todo esto:

$$C_{\underline{X}} = P\,D^{1/2}\left(D^{1/2}\right)^T P^T = A\,A^T$$

con $A = P\,D^{1/2}$.

Podemos ver que A calculada de esta manera no es simétrica.

1.1.2 Método 2

Usamos nuevamente la descomposición de autovectores y autovalores, pero esta vez logramos una una A simétrica:

$$\begin{aligned}
C_{\underline{X}} &= P\,D^{1/2}\left(D^{1/2}\right)^T P^T \\
&= P\,D^{1/2}\,I\left(D^{1/2}\right)^T P^T \\
&= P\,D^{1/2}\,P^T\,P\left(D^{1/2}\right)^T P^T \\
&= \left(P\,D^{1/2}\,P^T\right)\left(P\,D^{1/2}\,P^T\right)^T \\
&= A\,A^T
\end{aligned}$$

donde aprovechamos el hecho que P es una matriz ortogonal y entonces: $P^T = P^{-1}$. Podemos ver que en este caso: $A = \left(P\,D^{1/2}\,P^T\right)$ si es una matriz simétrica.

1.1.3 Método 3

Usamos la Descomposición de Cholesky de $C_{\underline{X}}$ para decomponerla en el producto de una matriz y su traspueta:

$$C_{\underline{X}} = A\,A^T$$

Volvamos al Ejercicio 10...

1.2 Punto 1

Obtenga N = 10000 realizaciones del vector X. Estime la matriz de covarianza y compare el resultado con Cx.

Vamos a asumir que asumir que partimos de N realizaciones de un vector normal y vamos a colorearlo.

```
In [1]: # importamos las librerías que vamos a utilizar
        import math as m
        import numpy as np
        import scipy as sp
        import matplotlib.pyplot as plt
```

Generamos N realizaciones de un vector gaussiano normal: $\underline{Y} \sim \mathcal{N}(\mu_{\underline{Y}}, C_{\underline{Y}})$ con $\mu_{\underline{Y}} = \underline{0}_{3\times 1}$ y $C_{\underline{Y}} = I_{3\times 3}$.

```
In [2]: N = int(1e4)
        Y = np.random.randn(3,N)
```

A partir de las realizaciones de $\underline{Y}$ generamos las N realizaciones de $\underline{X}$ pedidas. Primero notemos que en este caso: $\underline{b} = \underline{0}$. Luego, hallemos la matriz A, tal que:

$$\begin{aligned} C_{\underline{X}} &= A\,I\,A^T \\ &= A\,A^T \end{aligned}$$

```
In [3]: Cx = np.array([[1,0,0],[0,2.5,-0.5],[0,-0.5,2.5]])
        # hallemos la descomposición en autovalores y autovectores de Cx
        L,P = np.linalg.eig(Cx)
        Pi = P.transpose()
        D = np.diag(L)
```

- Método 1: $A = P\,D^{1/2}$

```
In [4]: A1 = np.dot(P,np.sqrt(D))
        print('A = ')
        print(A1)
        # podemos verificar que Cx = A1 A1^T
        print('Cx = ')
        print(np.dot(A1,A1.transpose()))
        # aplicamos la transformación: X1 = A1 X + b
        X1 = np.dot(A1,Y)
```

```
A =
[[ 0.          0.          1.        ]
 [-1.22474487  1.          0.        ]
 [ 1.22474487  1.          0.        ]]
Cx =
[[ 1.   0.   0. ]
 [ 0.   2.5 -0.5]
 [ 0.  -0.5  2.5]]
```

- Método 2: $A = \left(P\, D^{1/2}\, P^T\right)$

```
In [5]: A2 = np.dot(A1,Pi)
        print('A = ')
        print(A2)
        # podemos verificar que Cx = A2 A2^T
        print('Cx = ')
        print(np.dot(A2,A2.transpose()))
        # aplicamos la transformación: X2 = A2 X + b
        X2 = np.dot(A2,Y)
```

```
A =
[[ 1.          0.          0.        ]
 [ 0.          1.57313218 -0.15891862]
 [ 0.         -0.15891862  1.57313218]]
Cx =
[[ 1.   0.   0. ]
 [ 0.   2.5 -0.5]
 [ 0.  -0.5  2.5]]
```

- Método 3: Descomposición de Cholesky

```
In [6]: A3 = np.linalg.cholesky(Cx)
        print('A = ')
        print(A3)
        # podemos verificar que Cx = A3 A3^T
        print('Cx = ')
        print(np.dot(A3,A3.transpose()))
        # aplicamos la transformación: X3 = A3 X + b
        X3 = np.dot(A3,Y)
```

```
A =
[[ 1.          0.          0.        ]
 [ 0.          1.58113883  0.        ]
 [ 0.         -0.31622777  1.54919334]]
Cx =
[[ 1.   0.   0. ]
 [ 0.   2.5 -0.5]
```

```
 [ 0.  -0.5  2.5]]
```

A partir de las realizaciones generadas, podemos verificar la transformación calculando empíricamente $C_{\underline{X}}$:

```
In [7]: # estimacion de la matriz de covarianza
        Cx_calc = np.cov(X1)
        # estimacion de la media
        mux_calc = np.mean(X1,axis=1)
        print('Cx = ')
        print(np.matrix.round(Cx_calc,1))
        print('mux = ')
        print(np.matrix.round(mux_calc,1))
```

```
Cx =
[[ 1.  -0.  -0. ]
 [-0.   2.5 -0.5]
 [-0.  -0.5  2.5]]
mux =
[-0.  0.  0.]
```

1.3 Punto 2

Obtenga la matriz A tal que Z = AX tenga componentes descorrelacionadas. Esta transformación "blanquea" las componentes de X.

En este punto queremos blanquear las componentes del vector $\underline{X}$, es decir, queremos descorrelacionar las componentes del vector $\underline{X} = [X_1, X_2, X_3]^T$ obtenido en el punto anterior. La matriz de covarianza:

$$\begin{aligned}
C_{\underline{Z}} &= \mathbb{E}\left[\underline{Z}\,\underline{Z}^T\right] \\
&= \mathbb{E}\left[\begin{bmatrix} Z_1 \\ Z_2 \\ Z_3 \end{bmatrix} [Z_1, Z_2, Z_3]\right] \\
&= \begin{bmatrix} \mathbb{E}\left[Z_1^2\right] & \mathbb{E}\left[Z_1 Z_2\right] & \mathbb{E}\left[Z_1 Z_3\right] \\ \mathbb{E}\left[Z_2 Z_1\right] & \mathbb{E}\left[Z_2^2\right] & \mathbb{E}\left[Z_2 Z_3\right] \\ \mathbb{E}\left[Z_3 Z_1\right] & \mathbb{E}\left[Z_3 Z_2\right] & \mathbb{E}\left[Z_3^2\right] \end{bmatrix} \\
&= \begin{bmatrix} \sigma_{Z_1}^2 & 0 & 0 \\ 0 & \sigma_{Z_2}^2 & 0 \\ 0 & 0 & \sigma_{Z_3}^2 \end{bmatrix}
\end{aligned}$$

donde vemos que las correlaciones entre Z_i y Z_j con $i \neq j$, que corresponden con los términos fuera de la diagonal, son cero. Por lo tanto, tenemos: $\underline{X} \sim \mathcal{N}(\underline{0}, C_{\underline{X}})$ y queremos obtener $\underline{Z} \sim \mathcal{N}(\underline{0}, \Lambda)$ con Λ una matriz diagonal.

Para esto podemos usar alguno de los métodos que vimos más arriba para obtener A:

$$
\begin{aligned}
\Lambda &= A\, C_{\underline{X}}\, A^T \\
&= A\, P\, D\, P^T\, A
\end{aligned}
$$

Si hacemos $A = P^T$ (donde P es la matriz de autovectores) entonces $\Lambda = D$ que es una matriz diagonal (matriz de autovalores).

```
In [8]: A4 = Pi
        print('A = ')
        print(A4)
        Z = np.dot(A4,X3)
```

```
A =
[[ 0.         -0.70710678  0.70710678]
 [ 0.          0.70710678  0.70710678]
 [ 1.          0.          0.        ]]
```

A partir de las realizaciones generadas, podemos verificar la transformación calculando empíricamente $C_{\underline{Z}}$:

```
In [9]: Cz_calc = np.cov(Z)
        muz_calc = np.mean(Z,axis=1)
        print("Cz = ")
        print(np.matrix.round(Cz_calc,1))
        print("muz = ")
        print(np.matrix.round(muz_calc,1))
```

```
Cz =
[[ 3. -0.  0.]
 [-0.  2. -0.]
 [ 0. -0.  1.]]
muz =
[-0.  0.  0.]
```