Playing Linear Regression with R @ Metodi Probabilistici e Statistici del Mercati Finanziari (MPSMF) 2022-2023 - Rel. 01 - 2023-03-24

Laurea Magistrale in Ingegneria dell'Informazione e dell'Automazione - Università di Roma "Tor Vergata"

R. Monte - roberto.monte@uniroma2.eu
Dipartimento d'Ingegneria Civile e Ingegneria Informatica - Università di Roma "Tor vergata"

March 30, 2023

Contents

1 Simple Regression of Real Random Variables

Let $(\Omega, \mathcal{E}, \mathbf{P}) \equiv \Omega$ a probability space, let $L^2(\Omega; \mathbb{R})$ be the Hilbert space of all real random variables on Ω having finite moment of order 2, and let $X, Y \in L^2(\Omega; \mathbb{R})$. Assume we have

$$Y = f(X) + U, \tag{1}$$

where $f : \mathbb{R} \to \mathbb{R}$ is a Borel function and $U \in L^2(\Omega; \mathbb{R})$ is uncorrelated with X, in symbols $U \perp X$.

Definition 1.1 (Simple Univariate Regression)**.** Equation (1) is called a *univariate simple regression equation.*

Definition 1.2 (Regression Terms)**.** In the vast literature on regressions, the variables X, Y, and U are known with different names. The variable X is called the *independent* or *explanatory variable*; it is also called the *predictor* or *regressor* (*variable*). The variable Y is called the *dependent* or *explained variable*; it is also called the *response* or *regressand* (*variable*). The random variable U is called the *error* or *disturbance* or *noise* (*variable*).

Definition 1.3 (Regression Equation)**.** Equation (1) is also said to be the *regression equation* of Y against X with error U.

Definition 1.4 (Regression Function)**.** We call the function $f : \mathbb{R} \to \mathbb{R}$ the *regression function* of Y against X and we call the graph of f,

$$\Gamma_f \equiv \left\{(x, y) \in \mathbb{R}^2 : y = f(x),\ x \in \mathbb{R}\right\}, \tag{2}$$

the *regression curve* of Y against X.

There is no loss in generality in assuming that

$$\mathbf{E}[U] = 0, \tag{3}$$

This is a natural characteristic of any unsystematic measurement error, independently of its distribution. Furthermore, if (3) were not true, we could rewrite Equation (1) in the form

$$Y = \tilde{f}(X) + \tilde{U} \tag{4}$$

by introducing the function $\tilde{f} : \mathbb{R} \to \mathbb{R}$ given by

$$\tilde{f} \stackrel{\text{def}}{=} f(x) + \mathbf{E}[U] \tag{5}$$

and the random variable

$$\tilde{U} \stackrel{\text{def}}{=} U - \mathbf{E}[U]. \tag{6}$$

Clearly, $\tilde{U}$ would satisfy Equation (3), and its distribution function would be just the shift of the distribution function of U.

Thanks to these considerations, from now on we will assume implicitly that the error variable U always satisfies Equation (3).

Proposition 1.1 (Conditional Expectation as Regression Function)**.** *Write $L^2\left(\Omega_{\sigma(X)}; \mathbb{R}\right)$ for the subspace of all $\sigma(X)$-measurable random variables in $L^2(\Omega; \mathbb{R})$, namely, all random variables in $L^2(\Omega; \mathbb{R})$ with observable states in light of the information conveyed by X. Setting*

$$U \stackrel{def}{=} Y - \mathbf{E}[Y \mid X], \tag{7}$$

we have

$$\mathbf{E}[U] = 0 \quad \textit{and} \quad \mathbf{D}^2[U] = \mathbf{E}\left[U^2\right] = \min\left\{\mathbf{E}\left[(Y - Z)^2\right] : Z \in L^2\left(\Omega_{\sigma(X)}; \mathbb{R}\right)\right\}. \tag{8}$$

In addition, the random variable U is uncorrelated with X.

Proof. Given any random variable $X \in L^2(\Omega;\mathbb{R})$ the conditional expectation operator $\mathbf{E}[\cdot \mid X] : L^2(\Omega;\mathbb{R}) \to L^2\left(\Omega_{\sigma(X)};\mathbb{R}\right)$ satisfies the equation

$$\mathbf{E}[Y \mid X] = \arg\min\left\{\mathbf{E}\left[(Y-Z)^2\right] : Z \in L^2\left(\Omega_{\sigma(X)};\mathbb{R}\right)\right\} \tag{9}$$

for every $Y \in L^2(\Omega;\mathbb{R})$. Eventually, the operator $\mathbf{E}[\cdot \mid X]$ is the orthogonal projection of $L^2(\Omega;\mathbb{R})$ on $L^2\left(\Omega_{\sigma(X)};\mathbb{R}\right)$. Furthermore,

$$\mathbf{E}[\mathbf{E}[Y \mid X]] = \mathbf{E}[Y] \quad \text{and} \quad X\mathbf{E}[Y \mid X] = \mathbf{E}[XY \mid X], \tag{10}$$

for every $Y \in L^2(\Omega;\mathbb{R})$. As a consequence, Equation (8) is clearly satisfied. In addition, considering Equation (10), by virtue of the properties of the expectation operator, we have

$$\begin{aligned}
Cov(X, Y-\mathbf{E}[Y \mid X]) &= \mathbf{E}[X(Y-\mathbf{E}[Y \mid X])] - \mathbf{E}[X]\,\mathbf{E}[(Y-\mathbf{E}[Y \mid X])] && (11)\\
&= \mathbf{E}[XY - X\mathbf{E}[Y \mid X]] - \mathbf{E}[X](\mathbf{E}[Y] - \mathbf{E}[\mathbf{E}[Y \mid X]]) && (12)\\
&= \mathbf{E}[XY] - \mathbf{E}[X\mathbf{E}[Y \mid X]] && (13)\\
&= 0. && (14)
\end{aligned}$$

This shows that the random variable U is uncorrelated with X. ■

In light of Proposition 1.1 the regression equation

$$Y = \mathbf{E}[Y \mid X] + U, \tag{15}$$

where

$$U \stackrel{\text{def}}{=} Y - \mathbf{E}[Y \mid X] \tag{16}$$

is the best regression equation in terms of *mean square distance*, which is the distance in $L^2(\Omega;\mathbb{R})$. In this equation the regression function $f : \mathbb{R} \to \mathbb{R}$ is given by

$$f(x) \stackrel{\text{def}}{=} \mathbf{E}[Y \mid X = x], \quad \forall x \in \mathbb{R} \tag{17}$$

where $\mathbf{E}[Y \mid X = \cdot] : \mathbb{R} \to \mathbb{R}$ is the conditional expectation of Y given $X = x$, that is, the *density* or *Radon-Nikodým* derivative $dP_X^Y/dP_X : \mathbb{R} \to \mathbb{R}$ of the real measure $P_X^Y : \mathcal{B}(\mathbb{R}) \to \mathbb{R}$, given by

$$P_X^Y(B) \stackrel{\text{def}}{=} \int_{\{X \in B\}} Y d\mathbf{P}_{|\sigma(X)}, \quad \forall B \in \mathcal{B}(\mathbb{R}), \tag{18}$$

with respect to the distribution $P_X : B(\mathbb{R}) \to \mathbb{R}$ of X, given by

$$P_X(B) \stackrel{\text{def}}{=} \mathbf{P}(X \in B), \quad \forall B \in \mathcal{B}(\mathbb{R}). \tag{19}$$

Hence, $\mathbf{E}[Y \mid X = \cdot]$ satisfies the equation

$$P_X^Y(B) = \int_B \mathbf{E}[Y \mid X = x]\, dP_X(x), \tag{20}$$

for every $B \in \mathcal{B}(\mathbb{R})$.

We consider some examples.

Example 1.1 (Regression Function of Independent Random Variables)**.** Assume that the random variable Y is independent of X. Then we have

$$\mathbf{E}[Y \mid X] = \mathbf{E}[Y], \quad \text{and} \quad U = Y - \mathbf{E}[Y]. \tag{21}$$

This means that the best regression functions in terms of the mean square distance is the constant function $f : \mathbb{R} \to \mathbb{R}$ given by

$$f \stackrel{\text{def}}{=} \mathbf{E}[Y], \quad \forall x \in \mathbb{R}. \tag{22}$$

Note that the error variable U ha mean 0 and is independent of X.

Example 1.2 (Regression Function of Dependent Random Variables)**.** Assume that the random variable Y is measurable with respect to $\sigma(X)$. Then we have

$$\mathbf{E}[Y \mid X] = Y \quad \text{and} \quad U = 0. \tag{23}$$

This means that the best regression functions in terms of the mean square distance is the identity function $f : \mathbb{R} \to \mathbb{R}$ given by

$$f(x) \overset{\text{def}}{=} x, \tag{24}$$

for every $x \in \mathbb{R}$.

Example 1.3 (Regression Function of Jointy Gaussian Random Variables)**.** Assume that the random variables X and Y are jointly Gaussian distributed. Then we have

$$\mathbf{E}[Y \mid X] = \mathbf{E}[Y] + Corr(X,Y)\frac{\mathbf{D}[Y]}{\mathbf{D}[X]}(X - \mathbf{E}[X]) \quad \text{and} \quad U = Y - \mathbf{E}[Y \mid X]. \tag{25}$$

Moreover, U is independent of X. Note that if X and Y are uncorrelated Equation (25) becomes Equation (21)

Regression Function of Jointy Gaussian Random Variables. Since X and Y are jointly Gaussian distributed both the random variable X and the random vector (X,Y) are absolutely continuous with density functions $f_X : \mathbb{R} \to \mathbb{R}$ and $f_{X,Y} : \mathbb{R}^2 \to \mathbb{R}$ given by

$$f_X(x) = \frac{1}{\sqrt{2\pi}\sigma_X}\exp\left(-\frac{(x-\mu_X)^2}{2\sigma_X^2}\right) \tag{26}$$

and

$$f_{X,Y}(x,y) = \frac{1}{2\pi\sigma_X\sigma_Y\sqrt{1-\rho_{X,Y}^2}}\exp\left(-\frac{1}{2(1-\rho_{X,Y}^2)}\left[\left(\frac{x-\mu_X}{\sigma_X}\right)^2 - 2\rho_{X,Y}\left(\frac{x-\mu_X}{\sigma_X}\right)\left(\frac{y-\mu_Y}{\sigma_Y}\right) + \left(\frac{y-\mu_Y}{\sigma_Y}\right)^2\right]\right), \tag{27}$$

respectively, where $\mu_X \equiv \mathbf{E}[X]$, $\mu_Y \equiv \mathbf{E}[Y]$, $\sigma_X \equiv \mathbf{D}[X]$, $\sigma_Y \equiv \mathbf{D}[Y]$, and $\rho_{X,Y} \equiv Corr(X,Y)$. Therefore, considering that the functions f_X and $f_{X,Y}$ are Gaussian and setting

$$f_{Y|X}(x,y) \equiv \frac{f_{X,Y}(x,y)}{f_X(x)} = \frac{1}{\sqrt{2\pi}\sigma_Y\sqrt{1-\rho_{X,Y}^2}}\exp\left(-\frac{\left(y-\mu_Y-\rho_{X,Y}\frac{\sigma_Y}{\sigma_X}(x-\mu_X)\right)^2}{2\sigma_Y^2(1-\rho_{X,Y}^2)}\right), \tag{28}$$

we can write

$$\begin{aligned}
P_X^Y(B) &= \int_{\{X\in B\}} Y\,d\mathbf{P}_{|\sigma(X)} = \int_\Omega 1_{\{X\in B\}}Y\,d\mathbf{P} && (29)\\
&= \int_{\mathbb{R}^2} 1_B(x)\,y f_{X,Y}(x,y)\,d\mu_L^2(x,y) = \int_{\mathbb{R}^2} 1_B(x)\,y f_{Y|X}(x,y) f_X(x) d\mu_L^2(x,y) && (30)\\
&= \int_{\mathbb{R}} 1_B(x)\left(\int_{\mathbb{R}} y f_{Y|X}(x,y) d\mu_L(y)\right) f_X(x) d\mu_L(x) && (31)\\
&= \int_B \left(\int_{\mathbb{R}} y f_{Y|X}(x,y) d\mu_L(y)\right) dP_X(x). && (32)
\end{aligned}$$

It then follows

$$\mathbf{E}[Y \mid X = x] = \int_{\mathbb{R}} y f_{Y|X}(x,y) d\mu_L(y). \tag{33}$$

On the other hand, setting $v \equiv \frac{1}{\sqrt{2}\sigma_Y\sqrt{1-\rho_{X,Y}^2}}\left(y-\mu_Y-\rho_{X,Y}\frac{\sigma_Y}{\sigma_Y}(x-\mu_X)\right)$, where x is assumed as a parameter, a straightforward computation yields

$$\begin{aligned}
\int_{\mathbb{R}} y\frac{f_{X,Y}(x,y)}{f_X(x)}d\mu_L(y) &= \int_{-\infty}^{+\infty}\frac{y}{\sqrt{2\pi}\sigma_Y\sqrt{1-\rho_{X,Y}^2}}\exp\left(-\frac{\left(y-\mu_Y-\rho_{X,Y}\frac{\sigma_Y}{\sigma_X}(x-\mu_X)\right)^2}{2\sigma_Y^2(1-\rho_{X,Y}^2)}\right)dy \\
&= \frac{1}{\sqrt{\pi}}\int_{-\infty}^{+\infty}\left(\sqrt{2}\sigma_Y\sqrt{1-\rho_{X,Y}^2}v+\mu_Y+\rho_{X,Y}\frac{\sigma_Y}{\sigma_X}(x-\mu_X)\right)\exp\left(-v^2\right)dv \\
&= \frac{\sqrt{2}\sigma_Y\sqrt{1-\rho_{X,Y}^2}}{\sqrt{\pi}}\int_{-\infty}^{+\infty}v\exp\left(-v^2\right)dv+\frac{\mu_Y+\rho_{X,Y}\frac{\sigma_Y}{\sigma_X}(x-\mu_X)}{\sqrt{\pi}}\int_{-\infty}^{+\infty}\exp\left(-v^2\right)dv \\
&= \mu_Y+\rho_{X,Y}\frac{\sigma_Y}{\sigma_X}(x-\mu_X).
\end{aligned}$$

We then obtain,

$$\mathbf{E}[Y \mid X=x]=\mu_Y+\rho_{X,Y}\frac{\sigma_Y}{\sigma_X}(x-\mu_X), \tag{34}$$

for every $x\in\mathbb{R}$. From the latter it immediately follows Equation (25). In the end, U is independent of X. In fact, since U is a linear combination of the jointly Gaussian distributed random variables X and Y, the random variables X and U are also jointly Gaussian distributed. On the other hand, U is uncorrelated with X and uncorrelated jointly Gaussian distributed random variables are independent. ∎

As a consequence of Example 1.3, when X and Y are jointly Gaussian distributed the best regression functions in terms of the mean square distance is the linear function $f:\mathbb{R}\to\mathbb{R}$ given by

$$f(x)\overset{\text{def}}{=}\beta_0+\beta_1 x, \quad \forall x\in\mathbb{R}, \tag{35}$$

where,

$$\beta_1\equiv Corr(X,Y)\frac{\mathbf{D}[Y]}{\mathbf{D}[X]}=\frac{Cov(X,Y)}{\mathbf{D}^2[X]} \quad \text{and} \quad \beta_0\equiv\mathbf{E}[Y]-\beta_1\mathbf{E}[X]. \tag{36}$$

Furthermore, we have

$$Corr(X,Y)=\frac{\beta_1}{\sqrt{\beta_1^2+\frac{\mathbf{D}^2[U]}{\mathbf{D}^2[X]}}} \quad \text{and} \quad \beta_1^2=\frac{Corr(X,Y)^2}{1-Corr(X,Y)^2}\frac{\mathbf{D}^2[U]}{\mathbf{D}^2[X]}. \tag{37}$$

In fact, it is well known that

$$Cov(X,Y)=Corr(X,Y)\mathbf{D}[X]\mathbf{D}[Y]. \tag{38}$$

and, since U is uncorrelated with X, we obtain

$$\mathbf{D}[Y]=\sqrt{\mathbf{D}^2[Y]}=\sqrt{\mathbf{D}^2[\beta_0+\beta_1 X+U]}=\sqrt{\beta_1^2\mathbf{D}^2[X]+\mathbf{D}^2[U]}. \tag{39}$$

Replacing Equations (38) and (39) in Equation (37) for β_1, we obtain

$$Corr(X,Y)=\frac{\beta_1\mathbf{D}[X]}{\sqrt{\beta_1^2\mathbf{D}^2[X]+\mathbf{D}^2[U]}}. \tag{40}$$

Equation (40) clearly implies (37).

In general, computing the conditional expectation $\mathbf{E}[Y \mid X]$ can be a rather difficult task and we should not expect that the regression function (17) is linear. However, it still makes sense to try to determine the coefficients β_0 and β_1 which make a linear regression function of the type (35) the best linear regression function in terms of the mean square distance.

Proposition 1.2 (Linear Regression Function with Uncorrelated Noise)**.** *Assume we can write the regression equation*

$$Y = f(X) + U, \tag{41}$$

where the regression function $f : \mathbb{R} \to \mathbb{R}$ *is given by*

$$f(x) \overset{def}{=} \beta_0 + \beta_1 x, \quad \forall x \in \mathbb{R} \tag{42}$$

and the the error random variable U *is uncorrelated with* X*. Then Equations* (36) *and* (37) *hold true.*

Linear Regression Function with Uncorrelated Noise. Since U is uncorrelated with X, by wirtue of the properties of the covariance functional, we have

$$Cov(X,Y) = Cov(X, f(X)) + Cov(X,U) = Cov(X, \beta_0 + \beta_1 X) + Cov(X,U) = \beta_1 \mathbf{D}^2[X]. \tag{43}$$

Furthermore,

$$\mathbf{E}[Y] = \mathbf{E}[f(X)] + \mathbf{E}[U] = \mathbf{E}[\beta_0 + \beta_1 X] = \beta_0 + \beta_1 \mathbf{E}[X]. \tag{44}$$

These imply Equation (36). Then, as shown above, from Equation (36) and the assumption on U, Equation (37) follows. ■

Proposition 1.3 (Optimal Coefficients for a Linear Regression Function)**.** *Consider the regression equation* (41)*, where the regression function* $f : \mathbb{R} \to \mathbb{R}$ *is linear (see Equation* (41)*). Then, in terms of the mean square distance, the optimal choice of the coefficients* β_0 *and* β_1 *is given by Equation* (36)*. In this case, the error random variable* U *turns out to be uncorrelated with* X*. Thus, also Equation* (37) *holds true.*

Optimal Coefficients for a Linear Regression Function. In terms of the mean square distance, the optimal choice of the coefficients β_0 and β_1 is obtained by solving the minimization problem for the function $MSE : \mathbb{R}^2 \to \mathbb{R}$ given by

$$MSE(\beta_0, \beta_1) \overset{\text{def}}{=} \mathbf{E}\left[(Y - (\beta_0 + \beta_1 X))^2\right], \quad \forall (\beta_0, \beta_1) \in \mathbb{R}^2. \tag{45}$$

On the other hand,

$$\begin{aligned} \mathbf{E}\left[(Y - (\beta_0 + \beta_1 X))^2\right] &= \mathbf{E}\left[Y^2 + \beta_0^2 + \beta_1^2 X^2 - 2\beta_0 Y - 2\beta_1 XY + 2\beta_0\beta_1 X\right] && (46) \\ &= \beta_0^2 - 2\beta_0 \mathbf{E}[Y] + 2\beta_0\beta_1 \mathbf{E}[X] + \beta_1^2 \mathbf{E}\left[X^2\right] - 2\beta_1 \mathbf{E}[XY] + \mathbf{E}\left[Y^2\right]. && (47) \end{aligned}$$

Therefore, we have

$$\partial_{\beta_0} MSE(\beta_0, \beta_1) = 2\beta_0 - 2\mathbf{E}[Y] + 2\beta_1 \mathbf{E}[X] \quad \text{and} \quad \partial_{\beta_1} MSE(\beta_0, \beta_1) = 2\beta_1 \mathbf{E}\left[X^2\right] - 2\mathbf{E}[XY] + 2\beta_0 \mathbf{E}[X]. \tag{48}$$

Hence, the first order conditions yield

$$\beta_0 = \mathbf{E}[Y] - \beta_1 \mathbf{E}[X] \tag{49}$$

and

$$\mathbf{E}\left[X^2\right]\beta_1 = \mathbf{E}[XY] - \beta_0 \mathbf{E}[X] = \mathbf{E}[XY] - (\mathbf{E}[Y] - \beta_1 \mathbf{E}[X])\mathbf{E}[X]. \tag{50}$$

The latter implies

$$\left(\mathbf{E}\left[X^2\right] - \mathbf{E}[X]^2\right)\beta_1 = \mathbf{E}[XY] - \mathbf{E}[X]\mathbf{E}[Y], \tag{51}$$

that is

$$\beta_1 = \frac{Cov(X,Y)}{\mathbf{D}^2[X]}. \tag{52}$$

Equations (49) and (52) give the coordinates of a candidate minimum point for the function MSE in the desired form. On the other hand, considering the Hessian matrix $HMSE : \mathbb{R}^2 \to \mathbb{R}^2 \times \mathbb{R}^2$ of the function MSE we have

$$HMSE(\beta_0, \beta_1) = \begin{pmatrix} \partial_{\beta_0,\beta_0} MSE(\beta_0, \beta_1) & \partial_{\beta_0,\beta_1} MSE(\beta_0, \beta_1) \\ \partial_{\beta_1,\beta_0} MSE(\beta_0, \beta_1) & \partial_{\beta_1,\beta_1} MSE(\beta_0, \beta_1) \end{pmatrix} = \begin{pmatrix} 1 & \mathbf{E}[X] \\ \mathbf{E}[X] & \mathbf{E}\left[X^2\right] \end{pmatrix}. \tag{53}$$

for every $(\beta_0, \beta_1) \in \mathbb{R}^2$. It then follows

$$\det\left(HMSE\left(\beta_0, \beta_1\right)\right) = \mathbf{D}^2\left[X\right] \quad \text{and} \quad \operatorname{tr}\left(HMSE\left(\beta_0, \beta_1\right)\right) = 1 + \mathbf{E}\left[X^2\right]. \tag{54}$$

Therefore, the Hessian matrix has strictly positive eigenvalues (unless X is a Dirac random variable), which imply that the candidate minimum point is actually the minimum of MSE. Note that showing the Hessian matrix $HMSE$ has strictly positive eigenvalues is equivalent to show that the function MSE is convex.

Having shown that the optimal choice of the coefficients β_0 and β_1 is given by Equation (36), we are left with showing that in this case the error random variable U is uncorrelated with X. In fact, considering Equation (52) we obtain

$$Cov\left(U, X\right) = Cov\left(Y - \left(\beta_0 + \beta_1 X\right), X\right) = Cov\left(Y, X\right) - \beta_1 \mathbf{D}^2\left[X\right] = 0. \tag{55}$$

■

The regression analysis aims to investigate the structure of the regression function $f : \mathbb{R} \to \mathbb{R}$ in light of some realizations of the variables X and Y and additional assumptions on the noise term U. However, in general, determining the structure of the regression function from scratch is a too ambitious goal. Actually, we usually postulate a specific form of the regression function depending on some unknown vector of parameters, namely, we assume that

$$f\left(x\right) \equiv f\left(x; \theta\right), \tag{56}$$

where $f\left(x; \theta\right)$ has a specific form and $\theta \equiv \left(\theta_1, \ldots, \theta_M\right) \in \Theta \subseteq \mathbb{R}^M$, for some $M \geq 1$, is a vector of real parameters. Because of this, the goal of the regression analysis reduces to the determination of the best estimate, in a sense that will be made precise below, for the values to assign to the entries of θ.

Definition 1.5 (Regression Parameters)**.** We call the vector $\theta \equiv \left(\theta_1, \ldots, \theta_M\right)$ introduced in Equation (56) the *regression parameter vector*. We also refer to the entries of θ as the *regression parameters*.

Example 1.4 (Trivial Regression)**.** The simplest example of regression function is the function $f : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ given by

$$f\left(x; \theta\right) \overset{\text{def}}{=} \theta, \quad \forall x \in \mathbb{R}, \tag{57}$$

where $\theta \in \mathbb{R}$ is the regression parameter.

Definition 1.6 (Trivial Regression)**.** We call a regression function of the form (57) a *simple trivial regression* of regression parameter θ and we call the regression curve Γ_f the *regression line* of *intercept* θ.

Remark (Trivial Regression). Assuming that the noise variable U is independent of the explanatory variable X and the regression function f is trivial, we have

$$\theta \equiv \mathbf{E}\left[Y\right]. \tag{58}$$

Proof. Since the noise variable U is independent of X we have

$$f\left(X\right) = \mathbf{E}\left[Y \mid X\right].$$

On the other hand, f is supposed to be trivial, which means

$$f\left(X\right) = \theta.$$

We Then, have

$$\theta = \mathbf{E}\left[f\left(X\right)\right] = \mathbf{E}\left[\mathbf{E}\left[Y \mid X\right]\right] = \mathbf{E}\left[Y\right],$$

as claimed. ■

Remark (Trivial Regression). Assuming that the noise variable U is independent of explanatory variable X and also the explained variable Y is independent of X, Then, the regression function is necessarily trivial and Equation (58) holds true.

Proof. Since the noise variable U is independent of X, we have

$$f(X) = \mathbf{E}[Y \mid X]. \tag{59}$$

On the other hand, since also the explained variable Y is independent of X, we have

$$\mathbf{E}[Y \mid X] = \mathbf{E}[Y]. \tag{60}$$

Combining (59) and (60) it follows that the regression function f is trivial and thanks to Remark 1 we obtain Equation (58). ∎

Example 1.5 (Linear Regression)**.** The simplest, non trivial, example of regression function is the function $f : \mathbb{R} \times \mathbb{R}^2 \to \mathbb{R}$ given by

$$f(x;\theta) \stackrel{\text{def}}{=} \alpha + \beta x, \quad \forall x \in \mathbb{R}, \tag{61}$$

where $\theta \equiv (\alpha, \beta) \in \mathbb{R}^2$ is the regression parameter.

Definition 1.7 (Linear Regression)**.** We call a regression function of the form (61) a *simple linear regression* of regression parameters α and β and we call the regression curve Γ_f the *regression line* of *intercept* α and *slope* β.

Remark (Linear Regression)*.* Assuming that regression function f is linear and the noise variable U is uncorrelated with the explanatory variable X, in symbols

$$Y = \alpha + \beta X + U, \quad U \perp X,$$

we recall that (see Proposition) we have

$$\beta = Corr(X,Y)\frac{\mathbf{D}[Y]}{\mathbf{D}[X]} \quad \text{and} \quad \alpha = \mathbf{E}[Y] - \beta\mathbf{E}[X]. \tag{62}$$

As a consequence,

$$Corr(X,Y) = \frac{\beta}{\sqrt{\beta^2 + \frac{\mathbf{D}^2[U]}{\mathbf{D}^2[X]}}} \quad \text{and} \quad \beta^2 = \frac{Corr(X,Y)^2}{1 - Corr(X,Y)^2}\frac{\mathbf{D}^2[U]}{\mathbf{D}^2[X]}. \tag{63}$$

Example 1.6. Other examples of commonly used regression functions are:

1. the function $f : \mathbb{R} \times \mathbb{R}^{M+1} \to \mathbb{R}$ given by

$$f(x;\theta) \stackrel{\text{def}}{=} \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \cdots + \alpha_M x^M, \quad \forall x \in \mathbb{R}, \tag{64}$$

   where $\theta \equiv (\alpha_0, \alpha_1, \ldots, \alpha_M) \in \mathbb{R}^{M+1}$, for $M \geq 2$, is the regression parameter;

2. the function $f : \mathbb{R} \times \mathbb{R}^2 \to \mathbb{R}$ given by

$$f(x;\theta) \stackrel{\text{def}}{=} \alpha \exp(\beta x), \quad \forall x \in \mathbb{R}, \tag{65}$$

   where $\theta \equiv (\alpha, \beta) \in \mathbb{R}^2$ is the regression parameter;

3. the function $f : \mathbb{R} \times \mathbb{R}^2 \to \mathbb{R}$ given by

$$f(x;\theta) \stackrel{\text{def}}{=} \frac{e^{\alpha+\beta x}}{1 + e^{\alpha+\beta x}}, \quad \forall x \in \mathbb{R}, \tag{66}$$

   where $\theta \equiv (\alpha, \beta) \in \mathbb{R}^2$ is the regression parameter.

4. the function $f : \mathbb{R} \times \mathbb{R}^{3M} \to \mathbb{R}$ given by

$$f(x;\theta) \overset{\text{def}}{=} \sum_{m=1}^{M} \alpha_m \cos(2\pi\phi_m x) + \sum_{m=1}^{M} \beta_m \cos(2\pi\phi_m x), \quad \forall x \in \mathbb{R}, \tag{67}$$

where $\theta \equiv (\alpha_1, \ldots, \alpha_M, \beta_1, \ldots, \beta_M, \phi_1, \ldots, \phi_M) \in \mathbb{R}^{3M}$.

Definition 1.8 (Polynomial Regression)**.** We call a regression function of the form (64), in 1 of Example 1.6, the *polynomial regression* of order m and regression parameters α_0, α_1, α_2,...,α_m. In particular, when $m = 2$ we speak of *quadratic regression* and we call the regression curve Γ_f the *regression parabola* of *intercept* α_0 and *vertex* $\left(-\alpha_1/2\alpha_2, \left(4\alpha_0\alpha_2 - \alpha_1^2\right)/4\alpha_2\right)$.

Definition 1.9 (Exponential Regression)**.** We call a regression function of the form (65), in 2 of Example 1.6, the *exponential regression* of regression parameters α and β.

Definition 1.10 (Logistic Regression)**.** We call a regression function of the form (66), in 3 of Example 1.6, the *logistic regression* of regression parameter α and β.

Definition 1.11 (Harmonic Regression)**.** We call a regression function of the form (67), in 4 of Example 1.6, the *harmonic regression* of order M and regression *amplitude* parameters $\alpha_1, \ldots, \alpha_M$, $\beta_1, \ldots, \beta_M$ and *frequency* parameters $\omega_1, \ldots, \omega_M$.

In Finance, Economics, and Medical Sciences [resp. Physics and Engineering] the exponential and the logistic [resp. harmonic] regressions play an important role. Below, we show some simple examples of applied regression equations.

Example 1.7 (Hooke's Law)**.** Hooke's Law states that the length L of a spring with a fixed extreme stressed or compressed by a force F, in the parallel direction to the spring, is proportional to the intensity of the stressing force. In symbols

$$L = L_0 + KF, \tag{68}$$

where L_0 is the length of the spring at rest and K is the stiffness of the spring. An experimental investigation of Hooke's Law requires considering the unavoidable imperfections in the material constituting the spring and the inaccuracy of the measures of the variables L and F. Hence, from an experimental point of view, Equation (68) should be more properly rewritten as

$$L = L_0 + KF + U \tag{69}$$

where U is a random variable independent of F which summarizes all possible deviations from (68). Equation (69) is a linear regression equation for the regressand $Y \equiv L$ against the regressor $X \equiv F$ with intercept parameter $\alpha \equiv L_0$, slope parameter $\beta \equiv K$, and noise term U.

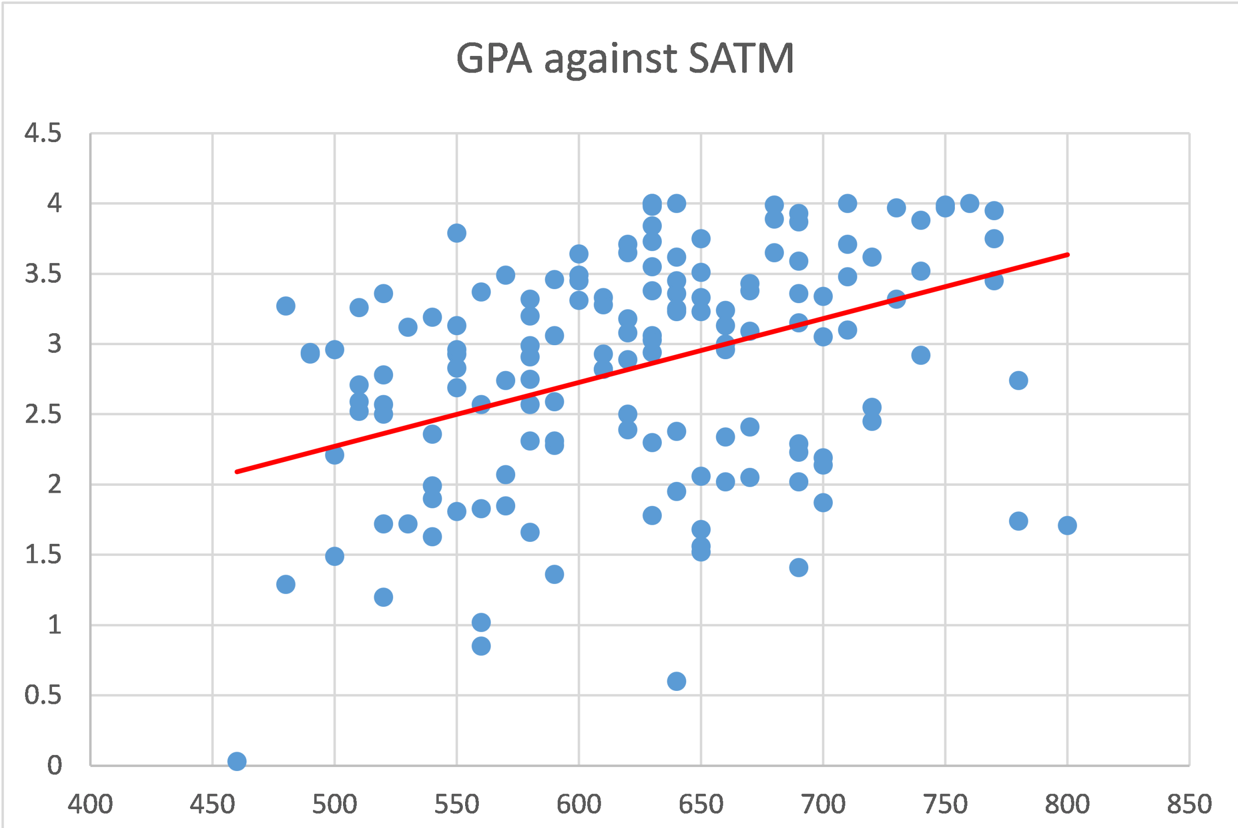
Example 1.8 (Ohm's Law)**.** Ohm's Law states that the intensity I of the electric current between the two ends of a homogeneous conductor is directly proportional to the voltage V across the ends of the conductor. The reciprocal of the constant of proportionality is known as the resistance R of the conductor. In symbols,

$$I = R^{-1}V. \tag{70}$$

As in the case of the Hooke's Law, an experimental investigation of Ohm's Law requires considering the unavoidable imperfections in the material constituting the conductor and the inaccuracy of the measures of the variables I and V. Hence, from an experimental point of view, Equation (70) should be more properly rewritten as

$$I = R^{-1}V + U. \tag{71}$$

where U is a random variable independent of V which summarizes all possible deviations from (70). Equation (71) is a linear regression equation for the regressand $Y \equiv I$ against the regressor $X \equiv V$ with intercept parameter $\alpha \equiv 0$, slope parameter $\beta \equiv R^{-1}$, and noise term U.

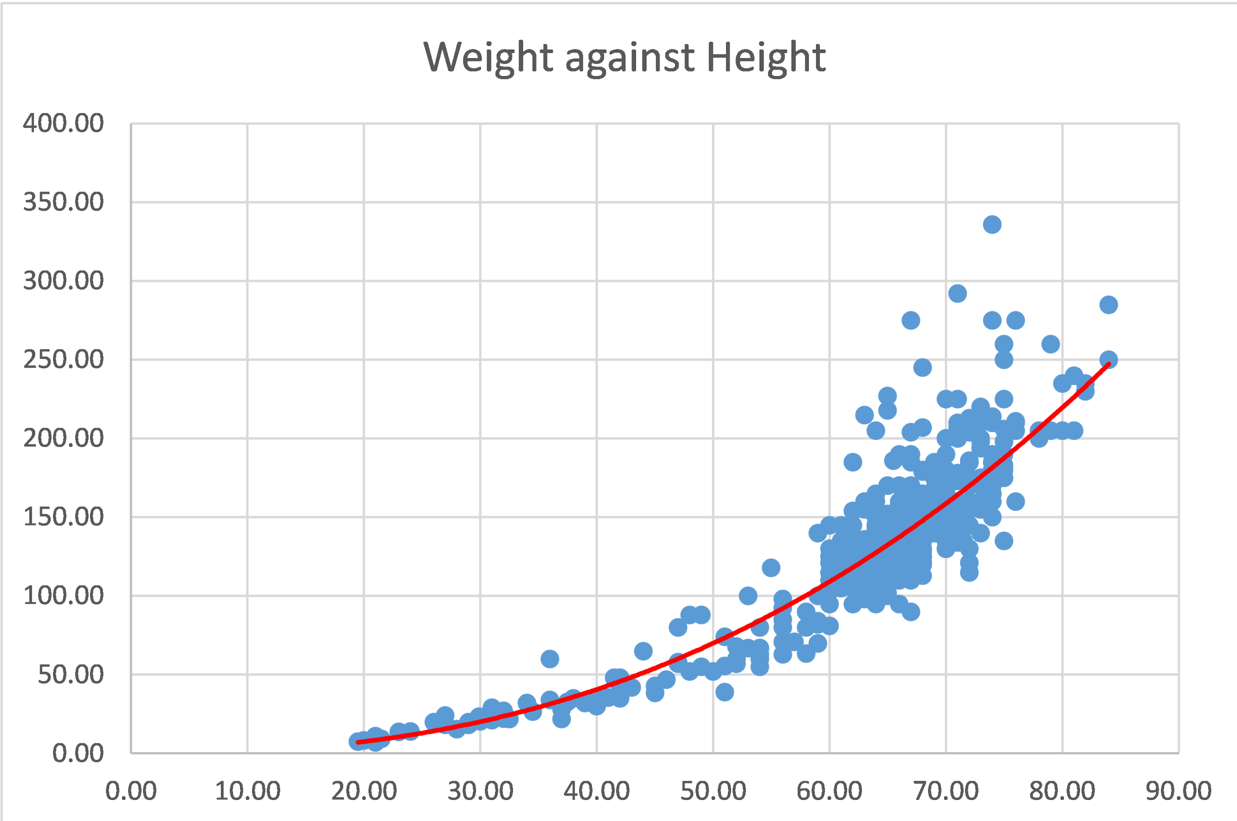


Example 1.9 (GPA against SATM)**.** Campbell & McCabe (see [**?**]) considered the problem of predicting the success of freshmen in a Computer Science Mayor on the basis of their performances at the high school level. We report a scatter plot from Campbell & McCabe's data relating the grade point average (GPA) of a single student in the computer science program, at a large midwestern university, and his/her score on the mathematics section of the scolastic attitude test (SATM). We assume the variable expressing each student's score SATM as the explanatory variable X and the variable expressing the student's GPA as the explained variable Y. Differently than Examples (1.7) and (1.8), in which a linear regression can be easily derived from a well established physical law, in this case no theoretical law is known to relate students' GPA and their score in SATM. Therefore, a possible regression has to established only on the basis of available data. However, looking at Figure 1.9, we can recognize a bit of a linear trend in the distribution of the points in the scatter plot, as evidenced by the trend line plotted. In light of this, we will consider the possibility of a linear regression of Y against X given by

$$Y = \alpha + \beta X + U,$$

where α and β are real parameters and U is the noise term.

Example 1.10 (Height against Weight)**.** Prof. N. Korevaar - University of Utah (see https://faculty.utah.edu/u0035213-NICK_KOREVAAR/teaching/index.hml) collected a number of human height-weight data which students in the Utah ACCESS class put together from friends and family on summer 2009 (see http://www.math.utah.edu/~korevaar/2270fall09/project2/htwts09.pdf, see also http://www.math.utah.edu/~korevaar/2270fall09/project2/). The goal was to relate the height and the weight of a large group of individuals, including infants, to test the body mass index (BMI) hypothesis. This hypothesis claims that the weight of a human being should be roughly proportional to the square of height. Below, we report the scatter plot relating the weight in pounds and the height in inches of the individuals in Korevaar's data set. We consider the variable expressing each individual's height as the explanatory variable X and



the variable expressing the individual's weight as the explained variable Y. Looking at Figure 1.10, we can recognize a rather pronounced quadratic trend in the distribution of the points in the scatter plot, as evidenced by the trend line plotted. In light of this, we will consider the possibility of a quadratic regression of Y against X given by

$$Y = \beta X^2 + U, \tag{72}$$

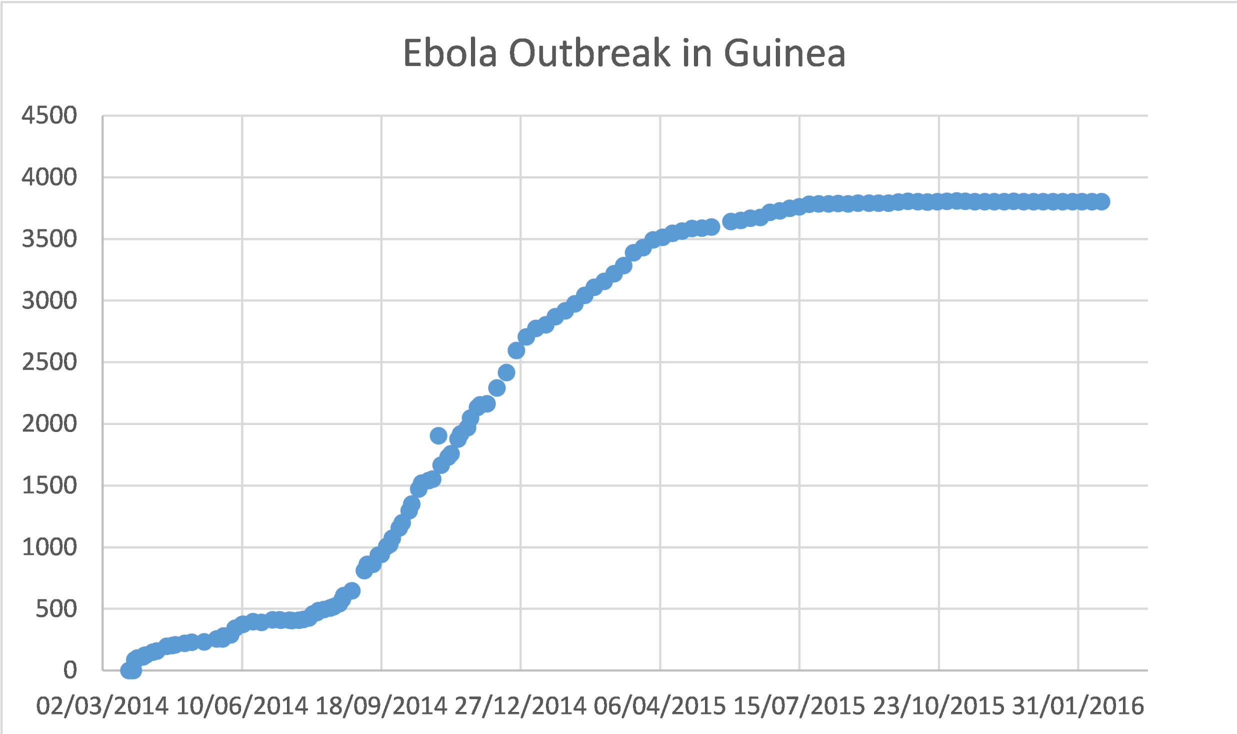
where β is a real parameter and U is the noise term. Nevertheless, a problem here is that the distance of the points in the scatter plot from the regression parabola appears to increase on the increasing of the values taken by the random variable X. This makes implausible to validate the assumption that the noise variable U in Equation (72) is uncorrelated with X.

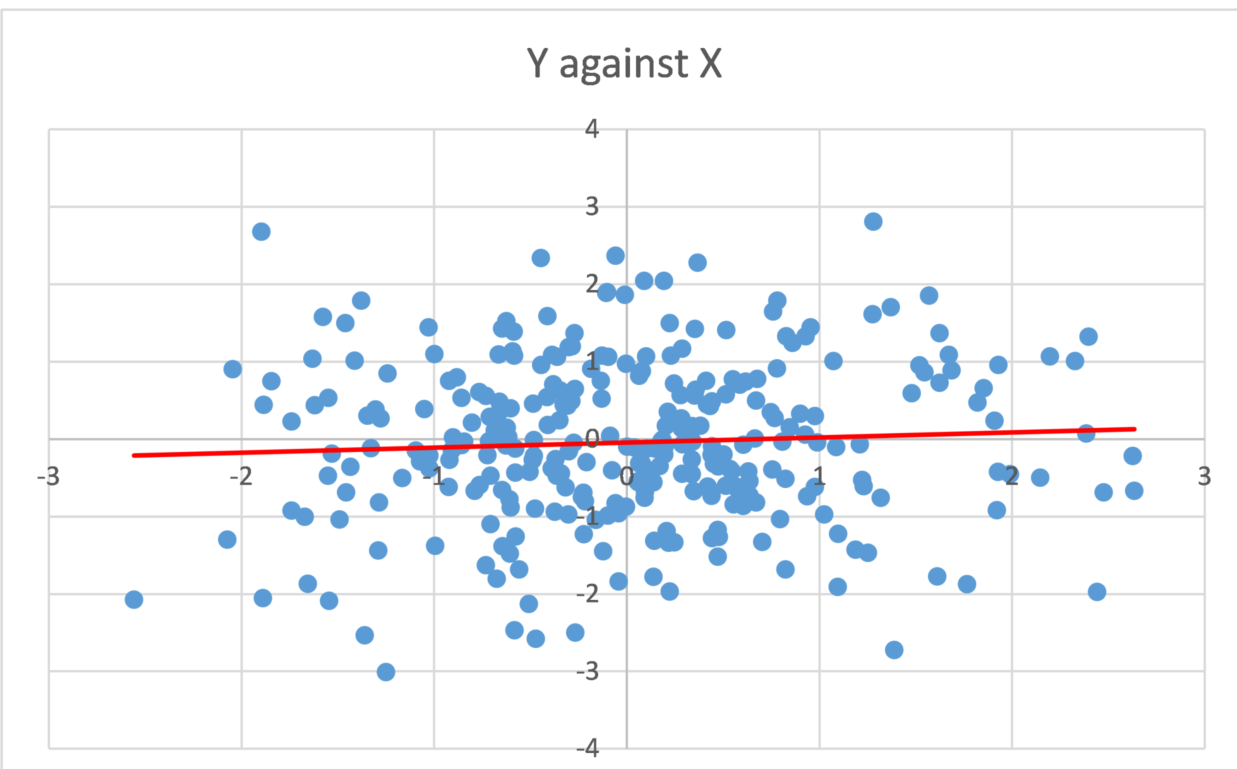
Example 1.11 (Ebola Outbreaks in Guinea)**.** The following scatter plot is drawn from the data published in WHO Situation Report n=3804 on total suspected, probable, and confirmed cases of Ebola disease during the Ebola outbreak in Guinea in the period March 25, 2014 – February 14, 2016 (see wttps://www.cdc.gov/vhf/ebola/outbreaks/2014-west-africa/cumulative-cases-graphs.html). Here, the explanatory variable X is the day of the recording and the explained variable Y is the number of the recorded cases. Looking at Figure 1.11, we can recognize a very strong logistic trend in the distribution of the points in the scatter plot. In light of this, we will consider the possibility of a logistic regression of Y against X given by

$$Y = \frac{e^{\alpha+\beta X}}{1+e^{\alpha+\beta X}} + U$$

where α and β are real parameters and U is the noise term..

Example 1.12 (Independent Gaussian Random Variables)**.** We have sampled 300 values drawn from two independent Gaussian standard random variables X and Y. Below, we report the scatter plot of Y against X. Looking at Figure 1.12, we can recognize no trend in the distribution of the points in the scatter plot, but a


Ebola Outbreak in Guinea
4500
4000
3500
3000
2500
2000
1500
1000
500
0
02/03/2014 10/06/2014 18/09/2014 27/12/2014 06/04/2015 15/07/2015 23/10/2015 31/01/2016


Y against X
4
3
2
1
0
-1
-2
-3
-4
-3
-2
-1
0
1
2
3

diffused cloud of points around the origin of the axis with a higher concentration close to the origin. Indeed, the trend line plotted is almost horizontal, meaning no trend is distinguishable. In this case, Equation (**??**) reduces to the trivial equation

$$Y = \mathbf{E}[Y] + U,$$

as it turns out by applying the conditional expectation operator $\mathbf{E}[\cdot \mid X]$ to both sides of (**??**).

2 Simple Regression of Real Random Data Sets

Let X [resp. Y] be a real random variable on a probability space $(\Omega, \mathcal{E}, \mathbf{P}) \equiv \Omega$. For some $n \geq 2$ consider n independent measurements of X [resp. Y], namely, a simple random sample $X_1, \ldots, X_n$ [resp. $Y_1, \ldots, Y_n$] of size n drawn from X [resp. Y], and write x_k [resp. y_k] for the value of the variable X [resp. Y] observed at the kth measurement. The real number x_k [resp. y_k] is the value $X_k(\omega)$ [resp. $Y_k(\omega)$], for $k = 1, \ldots, n$, on the occurring of $\omega \in \Omega$.

Definition 2.1 (Real Random Data Set)**.** We call the sequence $(x_k)_{k=1}^n$ [resp. $(y_k)_{k=1}^n$] a *real random data set* of size n drawn from X [resp. Y].

Assume there exist a function $f : \mathbb{R} \to \mathbb{R}$ and a random variable U uncorrelated with X such that Equation (1) holds true. Then, in terms of the data sets $(x_k)_{k=1}^n$ and $(y_k)_{k=1}^n$ we derive the equations

$$y_k = f(x_k) + u_k, \quad k = 1, \ldots, n, \tag{73}$$

where the sequence $(u_k)_{k=1}^n$ is made by independent realizations of the error random variable U at the kth measurement, for $k = 1, \ldots, n$, namely, u_k is the values $U_k(\omega)$ taken by the kth component of a simple random sample $U_1, \ldots, U_n$ drawn from U, on the occurring of ω.

Definition 2.2 (Observed Simple Regression Equation)**.** We call Equations (73) the *observed regression equation* of Y against X.

We stress that in regression analysis we always assume that the data set $(x_k)_{k=1}^n$ is observed. We will assume that the data set $(y_k)_{k=1}^n$ is observed or observable, but not yet observed, according to whether we are interested in estimating some parameters of the regression function or studying the properties of suitable estimators of these parameters. The true values of the parameters are to be considered not observed neither observable, as well as the sequence $(u_k)_{k=1}^n$ of the independent realizations of the error random variable U. From Equations (**??**), it is clearly seen that despite the assumption that the values of the data set $(x_k)_{k=1}^n$ are always observed, the ignorance of the true values of the parameters of the regression function makes the values of the data set $(f(x_k))_{k=1}^n$ not observable. Therefore, the possible observation of the values of the data set $(y_k)_{k=1}^n$ is not sufficient to observe the sequence of errors $(u_k)_{k=1}^n$. Conversely, the impossibility of observing the the sequence of errors $(u_k)_{k=1}^n$ makes the values of the data set $(f(x_k))_{k=1}^n$ not observable and prevents the observation of the true values of the parameters of the regression function.

Example 2.1 (Hooke's Law)**.** An investigator considers a homogeneous spring with a fixed extreme. He applies several straining and compressing forces of intensities $x_1, \ldots, x_n$ at the free end of the spring, in the parallel direction to the spring, and measures the corresponding lenght $y_1, \ldots, y_n$ of the spring. The investigator knows that the relationship between the observed data sets $(x_k)_{k=1}^n$ and $(y_k)_{k=1}^n$ is given by

$$y_k = L_0 + K x_k + u_k, \quad k = 1, \ldots, n. \tag{74}$$

where $u_1, \ldots, u_n$ are the unavoidable measurement errors (see Equations (69 and (**??**)). However, since the true values of the parameters L_0 and K are are not observable, the investigator cannot use the observed data sets $(x_k)_{k=1}^n$ and $(y_k)_{k=1}^n$ to determine the true values of the errors $u_1, \ldots, u_n$. Conversely, since the true values of the errors $u_1, \ldots, u_n$ cannot be observed, the investigator cannot use the observed data sets $(x_k)_{k=1}^n$ and $(y_k)_{k=1}^n$ to determine the true values of the parameters L_0 and K. He has to estimate them.

Example 2.2 (Ohm's law)**.** An investigator considers a homogeneous conductor with free ends. He applies several voltages $x_1, \ldots, x_n$ across the ends of the conductor, and measures the corresponding intensities $y_1, \ldots, y_n$ of the electric current between the two ends of the conductor. The investigator knows that the relationship between the observed data sets $(x_k)_{k=1}^n$ and $(y_k)_{k=1}^n$ is given by

$$y_k = R^{-1}x_k + u_k, \quad k = 1, \ldots, n. \tag{75}$$

where $u_1, \ldots, u_n$ are the unavoidable measurement errors (see Equations (69 and (**??**)). However, as in Example (2.1), since the true value of the parameters R is not observable, the investigator cannot use the observed data sets $(x_k)_{k=1}^n$ and $(y_k)_{k=1}^n$ to determine the true values of the errors $u_1, \ldots, u_n$. Conversely, since the true values of the errors $u_1, \ldots, u_n$ cannot be observed, the investigator cannot use the observed data sets $(x_k)_{k=1}^n$ and $(y_k)_{k=1}^n$ to determine the true value of the parameter R. He has to estimate it.

Given that Equations (**??**) hold true and we assumed a specific form $f(x;\theta)$ for the regression function depending on an unknown vector parameter $\theta \in \Theta \subseteq \mathbb{R}^m$, we can introduce the following *ordinary least squares estimate* of the unknown regression parameter vector θ.

Definition 2.3. We call the *sum of squared errors*, acronym *SSE*, the function $SSE : \Theta \to \mathbb{R}_+$ given by

$$SSE(\theta) \stackrel{\text{def}}{=} \sum_{k=1}^n (y_k - f(x_k;\theta))^2. \tag{76}$$

Be aware that *SSE* is also known as the *residual sum of squares* (*RSS*).

Remark. We have

$$SSE(\theta) = \sum_{k=1}^n u_k^2 \tag{77}$$

This makes clear the denomination *sum of squared errors*.

Definition 2.4. We call the *ordinary least squares*(OLS)*estimate* of the unknown regression parameter vector θ the vector $\hat{\theta}$ such that

$$\hat{\theta} = \underset{\theta\in\Theta}{\arg\min} SSE(\theta). \tag{78}$$

The structure of the function $SSE : \Theta \to \mathbb{R}_+$ leads us to consider two different classes of regression models:

- the models characterized by a regression function which may be nonlinear in the regressor but is linear in the unknown parameters or can be made linear in the parameters by a suitable transformation;
- the models characterized by a regression function which is not linear in the unknown parameters and cannot be made linear in the parameters by any transformation without over-parametrizing the model.

The first class of models, the so called *intrinsecally linear models*, allow in principle an analytic treatment of the minimization problem (78), because the linearity of the regression function in the parameters results in linear first order conditions for the minimization problem. The second class of models lead to nonlinear first order conditions for the minimization problem (78), which are often difficult to solve. For the latter models, numerical methods are often applied.

3 Simple Linear Regression of Real Datasets

For some $n \geq 2$ let $(x_k)_{k=1}^n$ [resp. $(y_k)_{k=1}^n$] be a data set of size n drawn from the real variable X [resp. Y]. Consider the linear regression function $f : \mathbb{R} \to \mathbb{R}$ given by Equation (61) in Example 1.5 and assume that Equation (**??**) holds true. Then, Equations (**??**) take the form

$$y_k = \alpha + \beta x_k + u_k, \quad k = 1, \ldots, n, \tag{79}$$

where $(\alpha, \beta) \equiv \theta$ is the vector parameter of the linear regression function and $(u_k)_{k=1}^n$ is a sequence of independent measurement errors. As a consequence, the function $SSE : \Theta \to \mathbb{R}_+$ introduced in Definition 2.3 takes the form

$$SSE(\alpha, \beta) = \sum_{k=1}^n (y_k - (\alpha + \beta x_k))^2 \tag{80}$$

and the OLS estimate of the unknown regression vector parameter (α, β) becomes the vector $\left(\hat{\alpha}, \hat{\beta}\right)$ such that

$$\left(\hat{\alpha}, \hat{\beta}\right) = \underset{(\alpha,\beta)\in\mathbb{R}^2}{\arg\min} SSE(\alpha, \beta) \tag{81}$$

Lemma 3.1. *Setting*

$$\bar{x}_n \equiv \frac{1}{n}\sum_{k=1}^n x_k, \quad \bar{y}_n \equiv \frac{1}{n}\sum_{k=1}^n y_k, \tag{82}$$

and

$$s_{X,n}^2 \equiv \frac{1}{n-1}\sum_{k=1}^n (x_k - \bar{x}_n)^2, \quad s_{X,Y,n} \equiv \frac{1}{n-1}\sum_{k=1}^n (x_k - \bar{x}_n)(y_k - \bar{y}_n). \tag{83}$$

we have

$$s_{X,n}^2 = \frac{1}{n-1}\left(\sum_{k=1}^n x_k^2 - n\bar{x}_n^2\right) \tag{84}$$

and

$$s_{X,Y,n} = \frac{1}{n-1}\left(\sum_{k=1}^n x_k y_k - n\bar{x}_n\bar{y}_n\right). \tag{85}$$

Corollary 3.1. *We have*

$$\frac{s_{X,Y,n}}{s_{X,n}^2} = \frac{\sum_{k=1}^n x_k y_k - n\bar{x}_n\bar{y}_n}{\sum_{k=1}^n x_k^2 - n\bar{x}_n^2}. \tag{86}$$

Proof. Combining (84) and (85), we clearly obtain Equation (86).% ■

Proposition 3.1. *With reference to the notation of Lemma 3.1, the OLS estimates $\hat{\alpha}$ and $\hat{\beta}$ of the regression parameters α and β are given by*

$$\hat{\alpha} \equiv \bar{y}_n - \hat{\beta}\ \bar{x}_n \quad \textit{and} \quad \hat{\beta} \equiv \frac{s_{X,Y,n}}{s_{X,n}^2}. \tag{87}$$

Proof. The OLS estimates in (87) are to be obtained by the minimization of the function $SSE : \mathbb{R}^2 \to \mathbb{R}_+$ given by (2.3). To this, we consider the first order conditions

$$\partial_\alpha SSE(\alpha, \beta) = \partial_\beta SSE(\alpha, \beta) = 0.$$

Rewriting we obtain%

$$\partial_\alpha SSE(\alpha, \beta) = 2n\alpha - 2n\bar{y}_n + 2n\beta\bar{x}_n$$

and%

$$\partial_\beta SSE(\alpha, \beta) = 2\beta\sum_{k=1}^n x_k^2 - 2\sum_{k=1}^n x_k y_k + 2\alpha n\bar{x}_n.$$

Therefore, the first order conditions yield

$$\alpha + \bar{x}_n\beta = \bar{y}_n \tag{88}$$

and

Definition 3.1. We call the line in the cartesian plane $\mathbb{R}^2$ of equation

$$y = \hat{\alpha} + \hat{\beta}x$$

the *regression line* of the data sets $(x_k)_{k=1}^n$ and $(y_k)_{k=1}^n$. We call the parameter $\hat{\alpha}$ [resp. $\hat{\beta}$] given by (87) the *intercept* [resp. *slope*] OLS *sample estimate*of the regression line.

Proposition 3.2. *Consider the sequence* $(u_k)_{k=1}^{n}$ *of independent measurement errors in Equation (79). Then, the relationship between the OLS estimates* $\hat{\alpha}$ *and* $\hat{\beta}$ *of the intercept and slope parameters of the regression line and the true values* α *and* β *are given by* <!–

$$--><!--\hat{\alpha} = \alpha + \frac{\frac{1}{n}--><!--<!--<!--\sum_{k=1}^{n}--><!--<!--u_k--><!--<!--<!--\sum_{k=1}^{n}--><}{<!--<!--<!--\sum_{k=1}^{n}--><!-} \tag{89}$$

–> <!– *and*<!–

$$--><!--\hat{\beta} = \beta + \frac{<!--<!--<!--\sum_{k=1}^{n}--><!--<!--x_k u_k - \bar{x}_n--><!--<!--<!--\sum_{k=1}^{n}--><!}{<!--<!--<!--\sum_{k=1}^{n}--><!--<!--x_k^2 - n\bar{x}_n^2} \tag{90}$$

–>

Proof. On account of (86) and (87), we can write

$$\begin{aligned}\hat{\alpha} &= \bar{y}_n - \frac{s_n(x,y)}{s_{X,n}^2}\bar{x}_n = \bar{y}_n - \frac{\sum_{k=1}^{n} x_k y_k - n\bar{x}_n\bar{y}_n}{\sum_{k=1}^{n} x_k^2 - n\bar{x}_n^2}\bar{x}_n \\ &= \frac{\bar{y}_n\left(\sum_{k=1}^{n} x_k^2 - n\bar{x}_n^2\right) - \left(\sum_{k=1}^{n} x_k y_k - n\bar{x}_n\bar{y}_n\right)\bar{x}_n}{\sum_{k=1}^{n} x_k^2 - n\bar{x}_n^2}.\end{aligned} \tag{91}$$

On the other hand, a straightforward computation yields%

$$\begin{aligned}&\bar{y}_n\left(\sum_{k=1}^{n} x_k^2 - n\bar{x}_n^2\right) - \left(\sum_{k=1}^{n} x_k y_k - n\bar{x}_n\bar{y}_n\right)\bar{x}_n \\ &= \bar{y}_n\sum_{k=1}^{n} x_k^2 - n\bar{x}_n^2\bar{y}_n - \bar{x}_n\sum_{k=1}^{n} x_k y_k + n\bar{x}_n^2\bar{y}_n \\ &= \frac{1}{n}\sum_{\ell=1}^{n} y_\ell \sum_{k=1}^{n} x_k^2 - \bar{x}_n\sum_{k=1}^{n} x_k y_k \\ &= \frac{1}{n}\sum_{\ell=1}^{n}\left(\alpha + \beta x_\ell + u_\ell\right)\sum_{k=1}^{n} x_k^2 - \bar{x}_n\sum_{k=1}^{n} x_k\left(\alpha + \beta x_k + u_k\right) \\ &= \frac{1}{n}\alpha\sum_{\ell=1}^{n}\sum_{k=1}^{n} x_k^2 + \frac{1}{n}\beta\sum_{\ell=1}^{n} x_\ell\sum_{k=1}^{n} x_k^2 + \frac{1}{n}\sum_{\ell=1}^{n} u_\ell\sum_{k=1}^{n} x_k^2 \\ &- \left(\alpha\bar{x}_n\sum_{k=1}^{n} x_k + \beta\bar{x}_n\sum_{k=1}^{n} x_k^2 + \bar{x}_n\sum_{k=1}^{n} x_k u_k\right) \\ &= \alpha\sum_{k=1}^{n} x_k^2 + \beta\bar{x}_n\sum_{k=1}^{n} x_k^2 + \frac{1}{n}\sum_{s=1}^{n} u_s\sum_{k=1}^{n} x_k^2 - \left(\alpha n\bar{x}_n^2 + \beta\bar{x}_n\sum_{k=1}^{n} x_k^2 + \bar{x}_n\sum_{k=1}^{n} x_k u_k\right) \\ &= \alpha\left(\sum_{k=1}^{n} x_k^2 - n\bar{x}_n^2\right) + \frac{1}{n}\sum_{k=1}^{n} u_k\sum_{k=1}^{n} x_k^2 - \bar{x}_n\sum_{k=1}^{n} x_k u_k.\end{aligned} \tag{92}$$

Combining (91) and (92), the desired (89) clearly follows. Similarly, we have

$$\hat{\beta} = \frac{s_{X,Y,n}}{s_{X,n}^2} = \frac{\sum_{k=1}^{n} x_k y_k - n\bar{x}_n\bar{y}_n}{\sum_{k=1}^{n} x_k^2 - n\bar{x}_n^2}, \tag{93}$$

where

$$\begin{aligned}\sum_{k=1}^{n} x_k y_k - n\bar{x}_n\bar{y}_n &= \sum_{k=1}^{n} x_k y_k - \bar{x}_n\sum_{k=1}^{n} y_k \\ &= \sum_{k=1}^{n} x_k\left(\alpha + \beta x_k + u_k\right) - \bar{x}_n\sum_{k=1}^{n}\left(\alpha + \beta x_k + u_k\right) \\ &= \sum_{k=1}^{n}\left(\alpha x_k + \beta x_k^2 + x_k u_k\right) - \bar{x}_n\left(n\alpha + \beta\sum_{k=1}^{n} x_k + \sum_{k=1}^{n} u_k\right) \\ &= n\alpha\bar{x}_n + \beta\sum_{k=1}^{n} x_k^2 + \sum_{k=1}^{n} x_k u_k - n\alpha\bar{x}_n - n\beta\bar{x}_n^2 - \bar{x}_n\sum_{k=1}^{n} u_k \\ &= \beta\left(\sum_{k=1}^{n} x_k^2 - n\bar{x}_n^2\right) + \sum_{k=1}^{n} x_k u_k - \bar{x}_n\sum_{k=1}^{n} u_k.\end{aligned} \tag{94}$$

Therefore, combining (93) and (94), we obtain (90).% ■

Definition 3.2. We call the kth *OLS estimated* or *fitted value* of the dependent variablethe real number $\hat{y}_k$ given by

$$\hat{y}_k \stackrel{\text{def}}{=} \hat{\alpha} + \hat{\beta}x_k, \quad \forall k = 1, \ldots, n. \tag{95}$$

Definition 3.3. We call the kth *OLS observed residual* the real number $\hat{u}_k$ given by

$$\hat{u}_k \overset{\text{def}}{=} y_k - \hat{y}_k, \quad \forall k = 1, \dots, n. \tag{96}$$

Note that, both the real numbers $\hat{y}_k$ and $\hat{u}_k$ are observable for any $k = 1, \dots, n$. From a graphical point of view, the kth estimated value $\hat{y}_k$ of the dependent variable is the ordinate of the point on the regression line with abscissa x_k and the kth residual is the vertical deviation of the point (x_k, y_k) from the regression line. If the residuals are small in magnitude, Then, much of the variability in the data set $(y_k)_{k=1}^n$ can be explained in terms of the variability in the data set $(x_k)_{k=1}^n$ and the linear relationship between the random variables X and Y.

Proposition 3.3. *We have*

$$\textstyle\sum_{k=1}^n \hat{u}_k = 0. \tag{97}$$

Proposition 3.4. *We have*

$$\frac{1}{n}\textstyle\sum_{k=1}^n \hat{y}_k = \bar{y}_n. \tag{98}$$

Proposition 3.5. *We have*

$$\hat{u}_k = \frac{\sum_{\ell=1}^n \left((n-1)\, s_{X,n}^2 \left(\delta_{k,\ell} - \frac{1}{n}\right) - (x_k - \bar{x}_n)(x_\ell - \bar{x}_n)\right) y_\ell}{x_k^2 - n\bar{x}_n^2}, \quad \forall k = 1, \dots, n, \tag{99}$$

where $\delta_{k,\ell}$ is the Kronecker delta, for all $k, \ell = 1, \dots, n$.

Proof. On account of (84), (85), (Equation 87), and (86), we can write

$$\begin{aligned}
\hat{u}_k &= y_k - \hat{y}_k = y_k - \left(\hat{\alpha} + \hat{\beta} x_k\right) \\
&= \left(y_k - \frac{s_{X,Y,n}^2}{s_{X,n}^2} x_k\right) - \left(\bar{y}_n - \frac{s_{X,Y,n}^2}{s_{X,n}^2} \bar{x}_n\right) \\
&= \left(y_k - \frac{\sum_{k=1}^n x_k y_k - n\bar{x}_n \bar{y}_n}{(n-1)\, s_{X,n}^2} x_k\right) - \left(\frac{1}{n}\textstyle\sum_{k=1}^n y_k - \frac{\sum_{k=1}^n x_k y_k - n\bar{x}_n \bar{y}_n}{(n-1)\, s_{X,n}^2} \bar{x}_n\right) \\
&= \left(y_k - \frac{\sum_{k=1}^n x_k y_k - \bar{x}_n \sum_{k=1}^n y_k}{(n-1)\, s_{X,n}^2} x_k\right) - \left(\frac{1}{n}\textstyle\sum_{k=1}^n y_k - \frac{\sum_{k=1}^n x_k y_k - \bar{x}_n \sum_{k=1}^n y_k}{(n-1)\, s_{X,n}^2} \bar{x}_n\right) \\
&= \left(\sum_{\ell=1}^n \delta_{k,\ell} y_\ell - \frac{\sum_{\ell=1}^n x_\ell y_\ell - \sum_{\ell=1}^n \bar{x}_n y_\ell}{(n-1)\, s_{X,n}^2} x_k\right) - \left(\textstyle\sum_{\ell=1}^n \frac{1}{n} y_\ell - \frac{\sum_{\ell=1}^n x_\ell y_\ell - \sum_{\ell=1}^n \bar{x}_n y_\ell}{(n-1)\, s_{X,n}^2} \bar{x}_n\right) \\
&= \left(\frac{(n-1)\, s_{X,n}^2 \sum_{\ell=1}^n \delta_{k,\ell} y_\ell - x_k \sum_{\ell=1}^n (x_\ell - \bar{x}_n)\, y_\ell}{(n-1)\, s_{X,n}^2}\right) - \left(\frac{(n-1)\, s_{X,n}^2 \sum_{\ell=1}^n \frac{1}{n} y_\ell - \bar{x}_n \sum_{\ell=1}^n (x_\ell - \bar{x}_n)\, y_\ell}{(n-1)\, s_{X,n}^2}\right) \\
&= \frac{\sum_{\ell=1}^n (n-1)\, s_{X,n}^2 \delta_{k,\ell} y_\ell - \sum_{\ell=1}^n x_k (x_\ell - \bar{x}_n)\, y_\ell}{\sum_{k=1}^n x_k^2 - n\bar{x}_n^2} - \frac{\sum_{\ell=1}^n \frac{n-1}{n} s_{X,n}^2 y_\ell - \sum_{\ell=1}^n (x_\ell - \bar{x}_n)\, \bar{x}_n y_\ell}{\sum_{k=1}^n x_k^2 - n\bar{x}_n^2} \\
&= \frac{\sum_{\ell=1}^n \left((n-1)\, s_{X,n}^2 \delta_{k,\ell} - x_k (x_\ell - \bar{x}_n)\right) y_\ell}{\sum_{k=1}^n x_k^2 - n\bar{x}_n^2} - \frac{\sum_{\ell=1}^n \left(\frac{n-1}{n} s_{X,n}^2 - (x_\ell - \bar{x}_n)\, \bar{x}_n\right) y_\ell}{\sum_{k=1}^n x_k^2 - n\bar{x}_n^2} \\
&= \frac{\sum_{\ell=1}^n \left((n-1)\, s_{X,n}^2 \delta_{k,s} - x_k (x_\ell - \bar{x}_n) - \frac{n-1}{n} s_{X,n}^2 + (x_\ell - \bar{x}_n)\, \bar{x}_n\right) y_\ell}{\sum_{k=1}^n x_k^2 - n\bar{x}_n^2} \\
&= \frac{\sum_{\ell=1}^n \left((n-1)\, s_{X,n}^2 \left(\delta_{k,s} - \frac{1}{n}\right) - (x_k - \bar{x}_n)(x_\ell - \bar{x}_n)\right) y_\ell}{\sum_{k=1}^n x_k^2 - n\bar{x}_n^2},
\end{aligned}$$

as desired. ∎

Definition 3.4. We call the *total sum of squares* (*TSS*), or *total variation* in the data set $(y_k)_{k=1}^n$. the sum of the squared deviations of $(y_k)_{k=1}^n$ about the horizontal line of equation $y = \bar{y}_n$, that is, the positive number

$$TSS \stackrel{\text{def}}{=} \textstyle\sum_{k=1}^n (y_k - \bar{y}_n)^2 . \tag{100}$$

The total sum of squares, TSS, expresses the overall variability in the data set $(y_k)_{k=1}^n$. In fact

Remark. We have

$$TSS = (n-1)\, s_{Y,n}^2. \tag{101}$$

Proposition 3.6. *We have*

$$TSS = \textstyle\sum_{k=1}^n (y_k - \hat{y}_k)^2 + \sum_{k=1}^n (\hat{y}_k - \bar{y}_n)^2 . \tag{102}$$

Proof. We can write

$$\begin{aligned} TSS &= \textstyle\sum_{k=1}^n (y_k - \hat{y}_k + \hat{y}_k - \bar{y}_n)^2 \\ &= \textstyle\sum_{k=1}^n (y_k - \hat{y}_k)^2 + \sum_{k=1}^n (\hat{y}_k - \bar{y}_n)^2 + 2\sum_{k=1}^n (y_k - \hat{y}_k)(\hat{y}_k - \bar{y}_n) . \end{aligned} \tag{103}$$

■

On the other hand, we have

$$\hat{y}_k - \bar{y}_n = \hat{\beta}(x_k - \bar{x}_n) \tag{104}$$

for every $k = 1, \dots, n$. In fact, Equation(Equation 87) implies

$$\hat{y}_k - \bar{y}_n = \alpha + \hat{\beta}x_k - \bar{y}_n = \bar{y}_n - \hat{\beta}\bar{x}_n + \hat{\beta}x_k - \bar{y}_n = \hat{\beta}(x_k - \bar{x}_n) . \tag{105}$$

Therefore, considering (104), (97), and (**??**), we obtain

$$\begin{aligned} \textstyle\sum_{k=1}^n (y_k - \hat{y}_k)(\hat{y}_k - \bar{y}_n) &= \hat{\beta}\textstyle\sum_{k=1}^n (y_k - \hat{y}_k)(x_k - \bar{x}_n) \\ &= \hat{\beta}\left(\textstyle\sum_{k=1}^n (y_k - \hat{y}_k)\, x_k - \bar{x}_n \sum_{k=1}^n (y_k - \hat{y}_k)\right) \\ &= \hat{\beta}\left(\textstyle\sum_{k=1}^n \hat{u}_k x_k - \bar{x}_n \sum_{k=1}^n \hat{u}_k\right) \\ &= 0. \end{aligned} \tag{106}$$

Combining (103) and (106), Equation (102) clearly follows.

Definition 3.5. We call the *explained sum of squares*, acronym *ESS*, or *explained variation*the positive number

$$ESS \stackrel{\text{def}}{=} \textstyle\sum_{k=1}^n (\hat{y}_k - \bar{y}_n)^2 . \tag{107}$$

The explained sum of squares, ESS, can be interpreted as *the amout of the total variation in the data set* $(y_k)_{k=1}^n$ *which can be explained in terms of the variability of the data set* $(x_k)_{k=1}^n$ *through the linear model.* This interpretation is better highlighted by the following Remark

Remark. We have

$$ESS = \hat{\beta}^2 (n-1)\, s_{X,n}^2. \tag{108}$$

We now apply R to study the (linear) regression of two jointly Gaussian random variables, one against the other.

Example 3.1 (Regression of Jontly Gaussian Random Variables)**.** Given any vector $\mu \equiv (\mu_1, \mu_2)^\intercal \in \mathbb{R}^2$, any matrix $A \equiv (a_{j,k})_{j,k=1}^2 \in \mathbb{R}^{2\times 2}$, and any pair of independent standard Gaussian variables Z_1 and Z_2, we know that the random vector $(X, Y)^\intercal$ satisfying the equation

$$\begin{pmatrix} X \\ Y \end{pmatrix} = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} + \begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix} \begin{pmatrix} Z_1 \\ Z_2 \end{pmatrix} \tag{109}$$

is Gaussian distributed, that is to say the random variables X and Y are jointly Gaussian distributed. Therefore, since we have

$$\mathbf{E}[X] = \mu_1, \quad \mathbf{D}^2[X] = a_{1,1}^2 + a_{1,2}^2, \quad \mathbf{E}[Y] = \mu_2, \quad \mathbf{D}^2[Y] = a_{2,1}^2 + a_{2,2}^2 \tag{110}$$

and

$$Cov(X, Y) = a_{1,1}a_{2,1} + a_{1,2}a_{2,2} \tag{111}$$

if we consider the regression of Y against X, then the regression function $f : \mathbb{R} \to \mathbb{R}$ is given by

$$f(x) = \mu_2 + \frac{a_{1,1}a_{2,1} + a_{1,2}a_{2,2}}{a_{1,1}^2 + a_{1,2}^2}(x - \mu_1) \tag{112}$$

for every $x \in \mathbb{R}$ (see (36)), and the noise variable

$$U = Y - f(X) \tag{113}$$

is Gaussian distributed.

From the computational side, we simulate two data sets *Gauss_1* and *Gauss_2* as sets of values taken by two independent Gaussian standard variables Z_1 and Z_2, respectively.

```
size <- 150
set.seed(12345, kind=NULL, normal.kind=NULL)# Setting the random seed "12345" for reproducibility.
Gauss_1 <- rnorm(n=size, mean=0, sd=1)# Simulating the sets of values taken by the "12345" Gaussian sta

set.seed(23451, kind=NULL, normal.kind=NULL)# Setting the random seed "23451" for reproducibility.
Gauss_2 <- rnorm(n=size, mean=0, sd=1)# Simulating the sets of values taken by the "23451" Gaussian sta
```

Since corresponding to different random seeds, the two simulated data set are allegedly realizations of two independent random standard Gaussian variables. To get a visual evidence of structure of the data sets, we consider the scatter plot of the data sets *Gauss_1* and *Gauss_2* against their indexing variable. To this, we start with building a data frame containing the data sets and the indexing variable itself.

```
n <- 150                                                # Choosing the data set size.
Gauss_df <- data.frame(k=1:n, Z_1=Gauss_1, Z_2=Gauss_2) # Building a data frame from the data sets *Gau
                                                        # *Gauss_2*, and the indexing variable.
head(Gauss_df)                                   # Showing the initial part of the data frame.
```

```
##   k        Z_1        Z_2
## 1 1  0.5855288  1.2202517
## 2 2  0.7094660  0.5116104
## 3 3 -0.1093033  0.2933357
## 4 4 -0.4534972  0.4111048
## 5 5  0.6058875 -2.1928016
## 6 6 -1.8179560  2.4308397
```

```r
tail(Gauss_df)                                 # Showing the final part of the data frame.
```

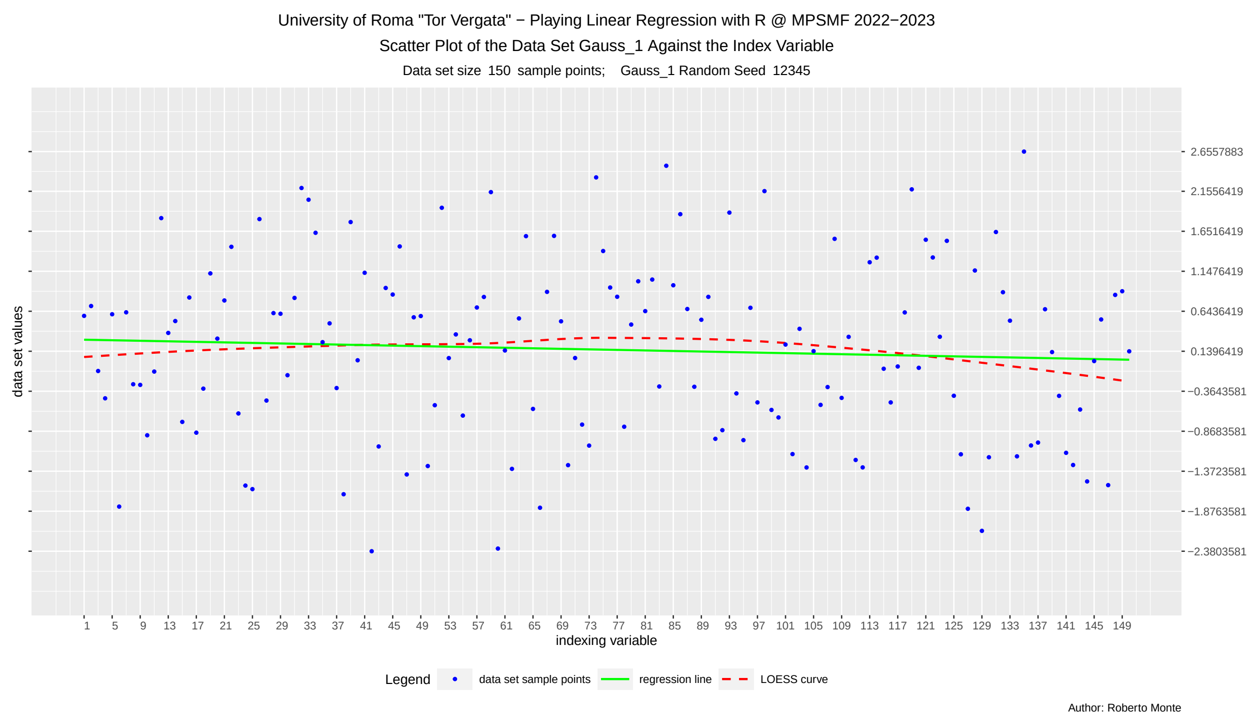
```
##       k         Z_1        Z_2
## 145 145  0.01585569  0.5995403
## 146 146  0.54016957 -1.5941019
## 147 147 -1.54729197 -0.7017336
## 148 148  0.84965293  0.2423192
## 149 149  0.89601318  0.7225111
## 150 150  0.13869100 -0.3104763
```

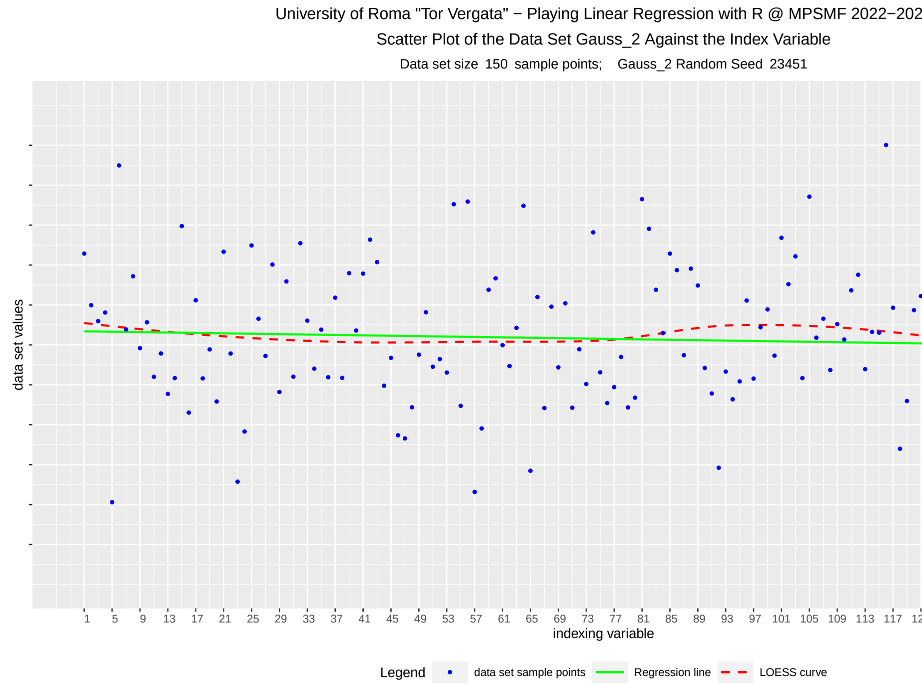
Hence, we can exploit the command *ggplot* to plot the scatter plot.

First, the scatter plots of the *Gauss_1* data set

```r
Data_df <- Gauss_df
n <- nrow(Data_df)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u0
subtitle_content <- bquote(paste("Data set size",~~.(n),~~"sample points;    Gauss_1 Random Seed",~~123
caption_content <- "Author: Roberto Monte"
# To obtain the submultiples of the length of the data set as a hint on the number of breaks to use
# library(numbers)
# primeFactors(n)
x_breaks_num <- 30
x_breaks_low <- Data_df$k[1]
x_breaks_up <- Data_df$k[n]
x_binwidth <- floor((x_breaks_up-x_breaks_low)/x_breaks_num)
x_breaks <- seq(from=x_breaks_low, to=x_breaks_up, by=x_binwidth)
if((x_breaks_up-max(x_breaks))>x_binwidth/2){x_breaks <- c(x_breaks,x_breaks_up)}
x_labs <- format(x_breaks, scientific=FALSE)
J <- 0
x_lims <- c(x_breaks_low-J*x_binwidth,x_breaks_up+J*x_binwidth)
x_name <- bquote("indexing variable")
y_breaks_num <- 10
y_max <- max(na.rm(Data_df$Z_1))
y_min <- min(na.rm(Data_df$Z_1))
y_binwidth <- round((y_max-y_min)/y_breaks_num, digits=3)
y_breaks_low <- y_min
y_breaks_up <- y_max
y_breaks <- seq(from=y_breaks_low, to=y_breaks_up, by=y_binwidth)
if((y_breaks_up-max(y_breaks))>y_binwidth/2){y_breaks <- c(y_breaks,y_breaks_up)}
y_labs <- format(y_breaks, scientific=FALSE)
y_name <- bquote("data set values")
K <- 1
y_lims <- c((y_breaks_low-K*y_binwidth), (y_breaks_up+K*y_binwidth))
col_1 <- bquote("data set sample points")
col_2 <- bquote("regression line")
col_3 <- bquote("LOESS curve")
leg_labs <- c(col_1, col_2, col_3)
leg_cols <- c("col_1"="blue", "col_2"="green", "col_3"="red")
leg_ord <- c("col_1", "col_2", "col_3")
Gauss_1_sp <- ggplot(Data_df, aes(x=k, y=Z_1)) +
  geom_smooth(alpha=1, linewidth=0.8, linetype="dashed", aes(color="col_3"),
              method="loess", formula=y ~ x, se=FALSE) +
  geom_smooth(alpha=1, linewidth=0.8, linetype="solid", aes(color="col_2"),
```

```
              method="lm" , formula=y ~ x, se=FALSE, fullrange=TRUE) +
  geom_point(alpha=1, size=1.0, shape=19, aes(color="col_1")) +
  scale_x_continuous(name=x_name, breaks=x_breaks, label=x_labs, limits=x_lims) +
  scale_y_continuous(name=y_name, breaks=y_breaks, labels=NULL, limits=y_lims,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  scale_colour_manual(name="Legend", labels=leg_labs, values=leg_cols, breaks=leg_ord,
                      guide=guide_legend(override.aes=list(shape=c(19,NA,NA),
                                                           linetype=c("blank", "solid", "dashed")))) +
  theme(plot.title=element_text(hjust=0.5), plot.subtitle=element_text(hjust=0.5),
        axis.text.x=element_text(angle=0, vjust=1),
        legend.key.width=unit(1.0,"cm"), legend.position="bottom")
plot(Gauss_1_sp)
```

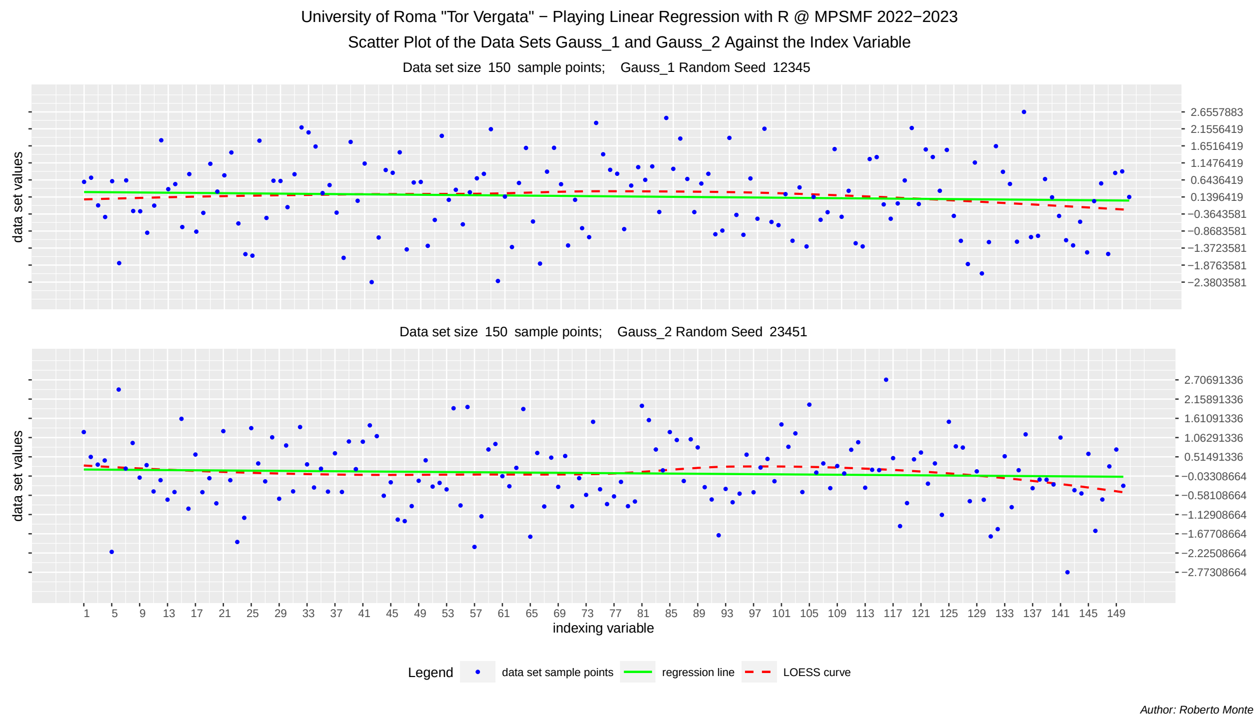




Second, the scatter plots of *Gauss_2* data set.

Third, the scatter plots of *Gauss_1* and *Gauss_2* data sets together.

```
grid_title <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u0040
grid_legend <- g_legend(Gauss_1_sp)

grid.arrange(arrangeGrob(Gauss_1_sp+theme(plot.title=element_blank(), axis.title.x=element_blank(),
                                          axis.ticks.x=element_blank(), axis.text.x=element_blank(),
                                          plot.caption=element_blank(), legend.position="none"),
                         Gauss_2_sp+theme(plot.title=element_blank(), legend.position="none", plot.capti
                         nrow=2, ncol=1, heights=c(4.5, 5.5)),
             top=textGrob(grid_title, gp=gpar(fontface=1, cex=1.1)),
             bottom=textGrob(caption_content, gp=gpar(fontface=3, fontsize=9), hjust=1, x=1),
             grid_legend, nrow=2, heights=c(10, 1))
```



From the inspection of the scatter plots of both data sets *Gauss_1* and *Gauss_2*, we have visual evidence for homogeneously spread sample points around an almost horizontal regression line. The almost flat LOESS line swinging slightly around the regression line strengthen this evidence,

The visual evidence makes us to think that both data sets have been generated by independent sampling from the same distribution.

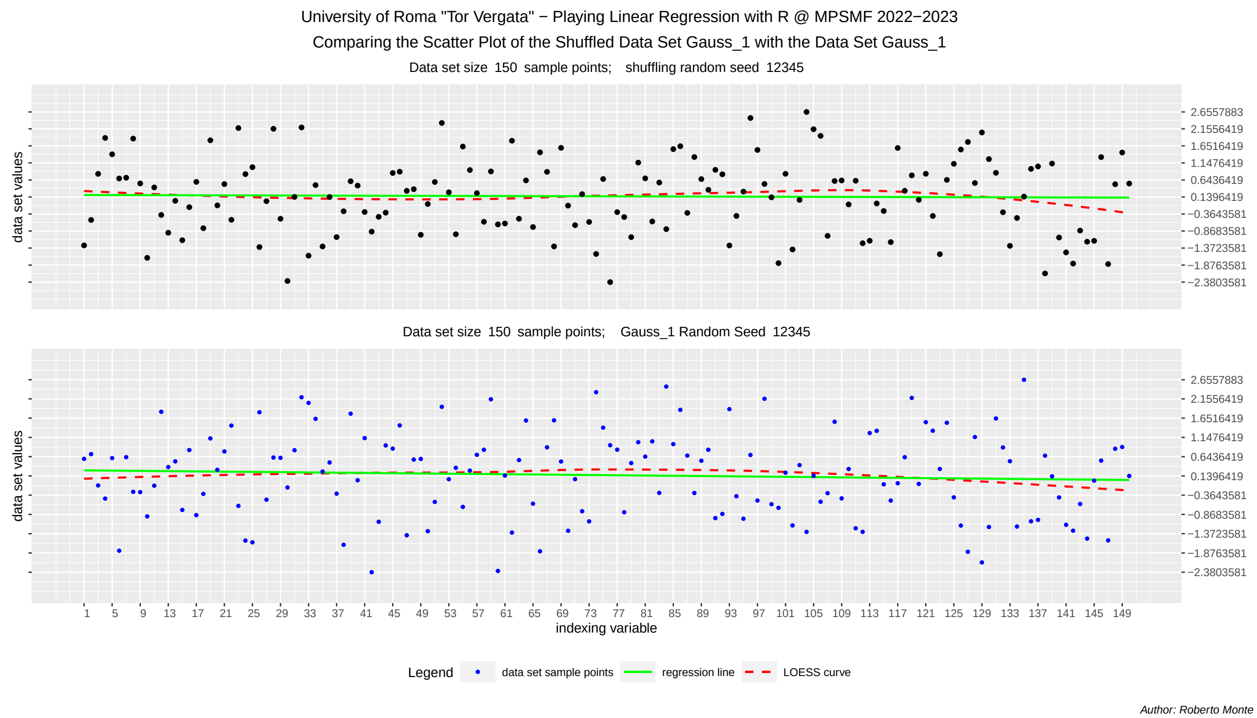
A further visual evidence, can be obtained by shuffling data sets *Gauss_1* and *Gauss_2* and comparing the scatter plots of the shuffled data sets with the original ones.

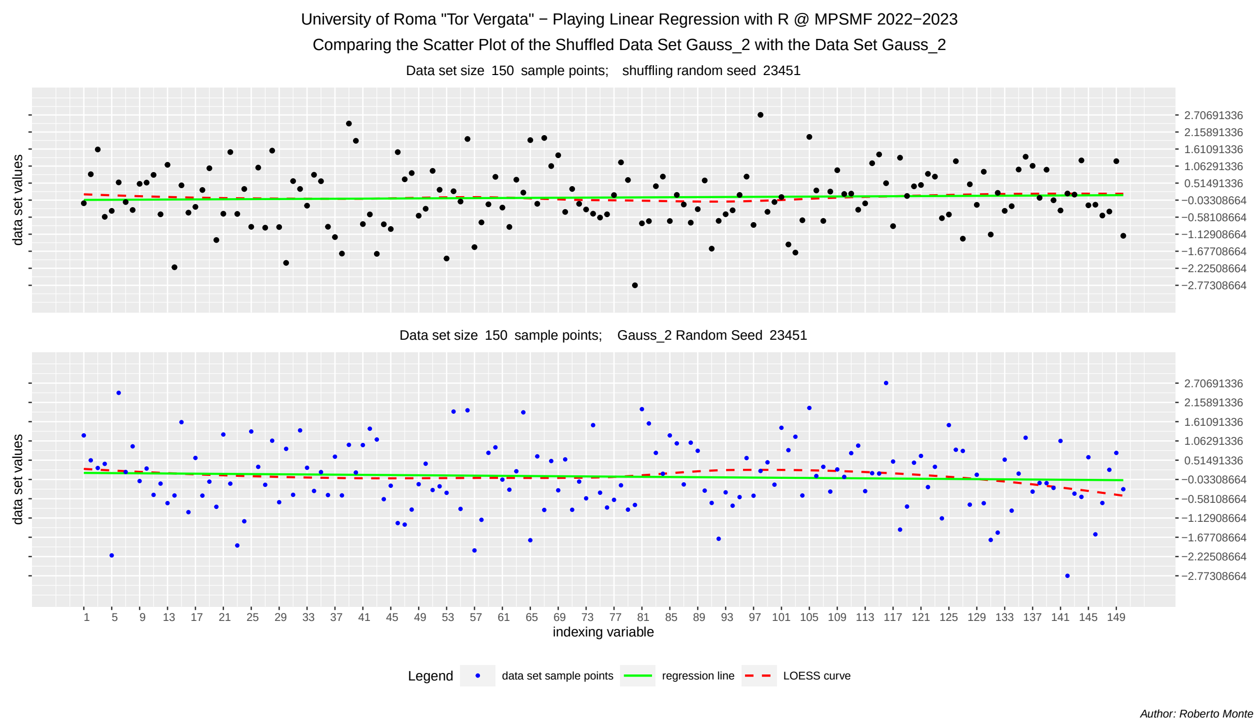
We build the shuffled data sets and add them to data frame containing the original data sets.

```
set.seed(12345)
Shuff_Z_1 <- sample(Gauss_df$Z_1, size=length(Gauss_df$Z_1), replace=FALSE, prob=NULL) # Randomly shuff
set.seed(23451)
Shuff_Z_2 <- sample(Gauss_df$Z_2, size=length(Gauss_df$Z_2), replace=FALSE, prob=NULL) # Randomly shuff
# library(tibble)
Gauss_df <- add_column(Gauss_df, Shuff_Z_1=Shuff_Z_1, Shuff_Z_2=Shuff_Z_2, .after="Z_2")
head(Gauss_df)
```

```
##   k        Z_1        Z_2  Shuff_Z_1  Shuff_Z_2
## 1 1  0.5855288  1.2202517 -1.2937153 -0.1328113
## 2 2  0.7094660  0.5116104 -0.5403861  0.8004549
## 3 3 -0.1093033  0.2933357  0.8237953  1.5974201
## 4 4 -0.4534972  0.4111048  1.8869469 -0.5700748
## 5 5  0.6058875 -2.1928016  1.4027054 -0.3789933
## 6 6 -1.8179560  2.4308397  0.6873321  0.5378371
```

We compare the scatter plot of the shuffled data sets with the original data sets.





The scatter plots of the shuffled data sets exhibit a lack of structure very similar to that of the original data sets. This constitutes visual evidence for uncorrelated randomness in the original data sets.

We consider also some computational test to confirm the visual evidence.

The issue of stationarity in the mean of the process which generates the data sets can be tackled by applying the Augmented Dickey-Fueller (ADF) test and Kwiatkowski-Phillips-Schmidt-Shin ($KPSS$) test in their simplest form. In this form, the ADF test takes as the null hypothesis that, on varying of the indexing variable, the data set is generated by a process containing a random walk component (*unit root*) while the alternative hypothesis that the data set is generated by an autoregressive process with no drift and trend, which is asymptotically stationary in the mean. On the contrary, the *KPSS* test takes as the null hypothesis that, on varying of the indexing variable, the data set is generated by an autoregressive process, while the alternative hypothesis is that the data set is generated by a process containing a random walk component. The rejection of the null hypothesis of the *ADF* test jointly with the lack of rejection of null hypothesis of the *KPSS* constitutes computational evidence for a stationary mean across the indexing variable.

Deepening the presentation of the rather complex *ADF* and *KPSS* tests is beyond the goal of these notes. However, we consider the simple application mentioned above.

```
# library(urca) # The library for this vesion of the test.
z <- Gauss_df$Z_1   # Choosing the data set to be tested.
no_lags <- 0        # Setting the lag parameter for the test.

Gauss_1_DF_none <- ur.df(z, type="none", lags=no_lags, selectlags="Fixed")
# Applying the form of the DF test which takes as the null hypothesis that the
# data set is generated by a process with a random walk component, while the
# alternative hypothesis is that the data set is generated by an autoregressive
# process with no drift and trend.
summary(Gauss_1_DF_none) # Showing the result of the test
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression none
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 - 1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.3892 -0.6737  0.2415  0.8708  2.6651
##
## Coefficients:
##         Estimate Std. Error t value Pr(>|t|)
## z.lag.1 -0.99215    0.08213  -12.08   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.135 on 148 degrees of freedom
## Multiple R-squared:  0.4965, Adjusted R-squared:  0.4931
## F-statistic: 145.9 on 1 and 148 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic is: -12.0806
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau1 -2.58 -1.95 -1.62
```

The test statistics of the *ADF* test takes value inside the rejection region at the significance level $\alpha = 0.01$ or

$\alpha = 1\%$. Therefore we can reject the null hypothesis in favor of the alternative.

```r
# library(urca)# The library for this version of the test
z <- Gauss_df$Z_1    # Choosing the data set to be tested

Gauss_1_KPSS_mu <- ur.kpss(z, type="mu", lags="nil", use.lag=NULL)
# Applying the simplest form of the KPSS test which considers the hull hypothesis that the data set is g
# by an autoregressive process with constant mean, while the alternative hypothesis is that the data se
# generated a process with a random walk component.

summary(Gauss_1_KPSS_mu)# Showing the result of the test
```

```
##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: mu with 0 lags.
##
## Value of test-statistic is: 0.1274
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

The test statistics of the *KPSS* test takes value outside the rejection region at the significance level $\alpha = 0.1$ or $\alpha = 10\%$. We cannot reject the null hypothesis in favor of the alternative.

```r
# library(urca) # The library for this vesion of the test.
z <- Gauss_df$Z_2   # Choosing the data set to be tested.
no_lags <- 0        # Setting the lag parameter for the test.

Gauss_2_DF_none <- ur.df(z, type="none", lags=no_lags, selectlags="Fixed")
# Applying the form of the DF test which takes as the null hypothesis that the data set is generated by
# a process with a random walk component, while the alternative hypothesis is that the data set is gene
# by an autoregressive process with no drift and trend.
summary(Gauss_2_DF_none) # Showing the result of the test
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression none
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 - 1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.71096 -0.55358 -0.02476  0.66302  2.71871
##
```

```
## Coefficients:
##         Estimate Std. Error t value Pr(>|t|)
## z.lag.1 -1.05843    0.08164  -12.96   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9608 on 148 degrees of freedom
## Multiple R-squared:  0.5317, Adjusted R-squared:  0.5286
## F-statistic: 168.1 on 1 and 148 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic is: -12.964
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau1 -2.58 -1.95 -1.62
```

The test statistics of the *ADF* test takes value inside the rejection region at the significance level $\alpha = 0.01$ or $\alpha = 1\%$. Therefore we can reject the null hypothesis in favor of the alternative.

```
# library(urca)# The library for this version of the test
z <- Gauss_df$Z_2    # Choosing the data set to be tested

Gauss_2_KPSS_mu <- ur.kpss(z, type="mu", lags="nil", use.lag=NULL)
# Applying the simplest form of the KPSS test which considers the hull hypothesis that the data set is
# by an autoregressive process with constant mean, while the alternative hypothesis is that the data se
# generated a process with a random walk component.
summary(Gauss_2_KPSS_mu)# Showing the result of the test
```

```
##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: mu with 0 lags.
##
## Value of test-statistic is: 0.0989
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

Note that with the goal of rejecting the null hypotesis the lowest is the value of the parameter α the strongest is the rejection. On the contrary, with the goal of not rejecting the null hypotesis the higher is the value of the parameter α (among the standard ones) the strongest is the not rejection.

As already mentioned, the rejection of the null hypothesis of random walk component in the random sample generating the data sets *Gauss_1* and *Gauss_2* in favor of the alternative hypothesis of autoregression without drift and trend, by the *ADF* test jointly with the lack of rejection of the null hypothesis of autoregression without trend against the alternative hypothesis of random walk component in the random sample generating the data sets *Gauss_1* and *Gauss_2* by the *KPSS* test constitute a significant computational evidence that the data sets *Gauss_1* and *Gauss_2* have been generated by processes with constant mean.

It is worth noting that in the summary of the ADF test the item "Call:" shows the linear regression exploited in the test. The formula notifies that the linear regression is given by

$$Z_k - Z_{k-1} = \beta_1 Z_{k-1} + U_k,$$

where Z_k [resp. U_k] is the kth sample component of the random variable Z which generates the data set [the error term U], on varying of the indexing variable $k = 1, \dots, n$ (the meaning of the "-1" is that the intercept α is forced to be 0). In addition, the item "Coefficients:" notifies us that we have the estimate

$$\hat{\beta}_1 = -0.99215 \quad \text{and} \quad \hat{\beta}_1 = -1.05843$$

according to whether it is considered the data set *Gauss_1* or *Gauss_2* and that in both cases the estimate is very significant. This means that it is very significant that the data sets *Gauss_1* and *Gauss_2* are satisfies the equations

$$Z_k = 0.00785 Z_{k-1} + U_k^{(1)} \quad \text{and} \quad Z_k = -0.05843 Z_{k-1} + U_k^{(2)}$$

where $U_k^{(1)}$ and $U_k^{(2)}$ are the kth sample component of suitable error terms. The above equations suggest that the data sets *Gauss_1* and *Gauss_2* are essentially estimated as stationary noises. Note also that the standard deviation of the error term σ_U is estimated by the Residual Standard Error as

$$\hat{\sigma}_U^{(1)} = 1.1350 \quad \text{and} \quad \hat{\sigma}_U^{(2)} = 0.9608.$$

To deal with the issue of constant variance, besides the visual inspection of the scatterplots of the data sets *Gauss_1* and *Gauss_2*, we can resort also on some tests: so called *heteroskedasticity* tests. We apply the *Breusch-Pagan* (*BP*) and *White* (W) test. Both of them take as the null hypothesis that the data set have been generated by a process with constant variance and as the alternative hypothesis that the variance of the generating process is not constant. As in the case of the *KPSS* and *ADF* tests, presenting a detailed explanation of the *BP* and W test is beyond the goal of these notes. We just show hot to apply them. To deal with the issue of constant variance, besides the visual inspection of the scatterplots of the data sets *Gauss_1* and *Gauss_2*, we apply the *Breusch-Pagan* (*BP*) and *White* (W) test.

The (unstudentized) *BP* test on the data set *Gauss_1*.

```
# Unstudentized Breusch-Pagan test
x <- Gauss_df$k    # The independent variable in the test
y <- Gauss_df$Z_1      # The dependent variable in the test
# Checking the empirical kurtosis of the data set according to which to select the option Studentize of
# library(EnvStats)# The library for the kurtosis function.
EnvStats::kurtosis(y, method="moment", excess=TRUE)
```

```
## [1] -0.6724514
```

```
# library(lmtest)# The library for this version of the test.
# The function for the BP test, which stores the results in the Gauss_1_BP list.
Gauss_1_BP <- bptest(formula=y~x, varformula=NULL, studentize=FALSE)
show(Gauss_1_BP) # The summary of the Gauss_1_BP list.
```

```
##
##  Breusch-Pagan test
##
## data:  y ~ x
## BP = 0.20687, df = 1, p-value = 0.6492
```

The (unstudentized) W test on the data set *Gauss_1*.

```r
# library(lmtest) # The library for this version of the test.
# White test
x <- Gauss_df$k
y <- Gauss_df$Z_1
var.formula <- ~ x+I(x^2) # The formula which allows to switch from *BP* to *W* test.
# The function for the W test, which stores the results in the Gauss_1_W list.
Gauss_1_W <- bptest(formula=y~x, varformula=var.formula, studentize=FALSE)
show(Gauss_1_W) # The summary of the Gauss_1_W list.
```

```
##
##  Breusch-Pagan test
##
## data:  y ~ x
## BP = 0.85373, df = 2, p-value = 0.6526
```

Both the *BP* and *W* test cannot reject the null hypothesis of homoskedasticity at any of the standard levels. In light of the visual inspections, and the results of the *BP* and *W* test, we have significant evidences to not reject the null hypothesis that the data set *Gauss_1* has been generated by a process with constant variance.

The same result holds true for the data set *Gauss_2*.

The (unstudentized) *BP* test on the data set *Gauss_2*.

```r
# Unstudentized Breusch-Pagan test
x <- Gauss_df$k    # The independent variable in the test
y <- Gauss_df$Z_2      # The dependent variable in the test
# Checking the empirical kurtosis of the data set according to which to select the option Studentize of
# library(EnvStats)# The library for the kurtosis function.
EnvStats::kurtosis(y, method="moment", excess=TRUE)
```

```
## [1] 0.1237472
```

```r
# library(lmtest)# The library for this version of the test.
# The function for the BP test, which stores the results in the Gauss_1_BP list.
Gauss_2_BP <- bptest(formula=y~x, varformula=NULL, studentize=FALSE)
show(Gauss_2_BP) # The summary of the Gauss_1_BP list.
```

```
##
##  Breusch-Pagan test
##
## data:  y ~ x
## BP = 0.011773, df = 1, p-value = 0.9136
```

The (unstudentized) *W* test on the data set *Gauss_2*.

```r
# library(lmtest)# The library for this version of the test.
# White test
x <- Gauss_df$k
y <- Gauss_df$Z_2
var.formula <- ~ x+I(x^2) # The formula which allows to switch from *BP* to *W* test.
# The function for the W test, which stores the results in the Gauss_1_W list.
Gauss_2_W <- bptest(formula=y~x, varformula=var.formula, studentize=FALSE)
show(Gauss_2_W)# The summary of the Gauss_1_W list.
```

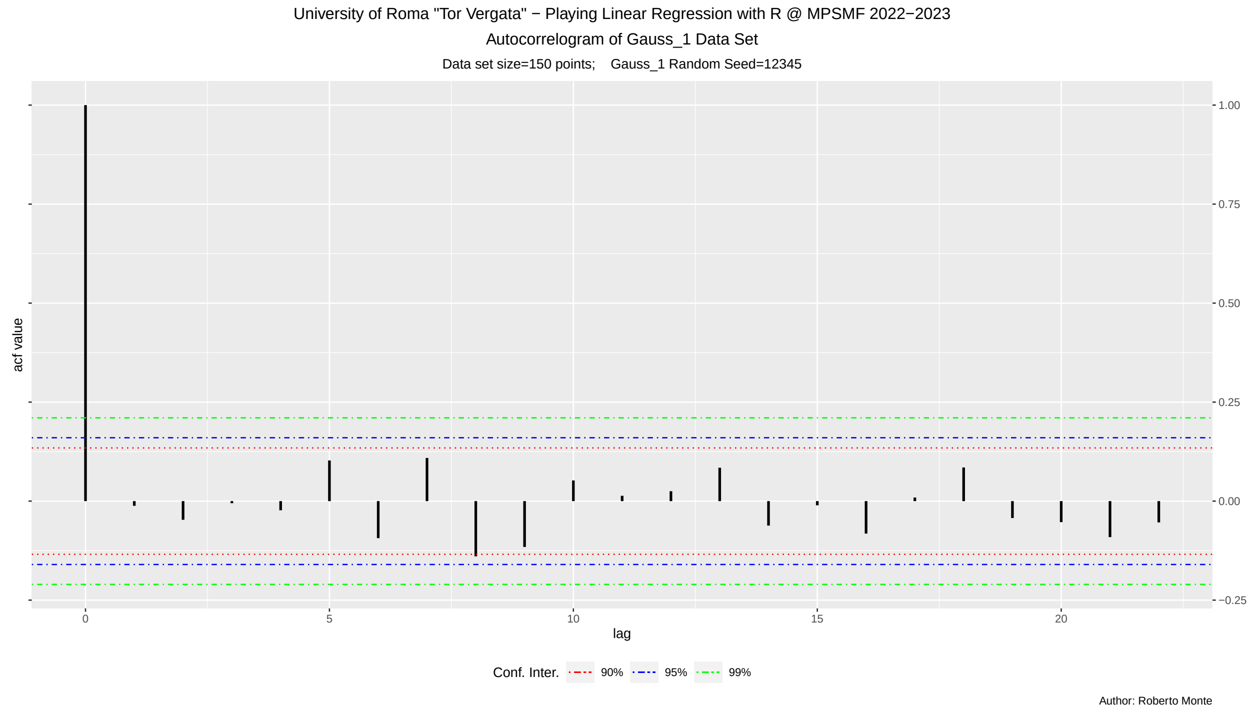
```
##
##  Breusch-Pagan test
##
## data:  y ~ x
## BP = 0.16078, df = 2, p-value = 0.9228
```

The third issue is that the data sets have been generated by independent random sampling from the same distribution (not necessarily Gaussian). We can make a visual check of this by plotting the autocorrelogram and the partial autocorrelogram of the data sets.

The autocorrelogram of the data set *Gauss_1*.

```
z <- Gauss_df$Z_1
n <- length(z)
maxlag <- ceiling(10*log10(n))
Aut_Fun_z <- acf(z, lag.max=maxlag, type="correlation", plot=FALSE)
ci_90 <- qnorm((1+0.90)/2)/sqrt(n)
ci_95 <- qnorm((1+0.95)/2)/sqrt(n)
ci_99 <- qnorm((1+0.99)/2)/sqrt(n)
Plot_Aut_Fun_z <- data.frame(lag=Aut_Fun_z$lag, acf=Aut_Fun_z$acf)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u00
paste("Autocorrelogram of Gauss_1 Data Set")))
subtitle_content <- bquote(paste("Data set size=", .(n), " points;    Gauss_1 Random Seed=12345"))
caption_content <- "Author: Roberto Monte"
ggplot(Plot_Aut_Fun_z, aes(x=lag, y=acf)) +
  geom_segment(aes(x=lag, y=rep(0,length(lag)), xend=lag, yend=acf), size=1, col="black") +
  geom_hline(aes(yintercept=-ci_90, color="CI_90"), show.legend=TRUE, lty=3) +
  geom_hline(aes(yintercept=ci_90, color="CI_90"), lty=3) +
  geom_hline(aes(yintercept=ci_95, color="CI_95"), show.legend=TRUE, lty=4) +
  geom_hline(aes(yintercept=-ci_95, color="CI_95"), lty=4) +
  geom_hline(aes(yintercept=-ci_99, color="CI_99"), show.legend=TRUE, lty=4) +
  geom_hline(aes(yintercept=ci_99, color="CI_99"), lty=4) +
  scale_x_continuous(name="lag", breaks=waiver(), label=waiver()) +
  scale_y_continuous(name="acf value", breaks=waiver(), labels=NULL,
                     sec.axis=sec_axis(~., breaks=waiver(), labels=waiver())) +
  scale_color_manual(name="Conf. Inter.", labels=c("90%","95%","99%"),
                     values=c(CI_90="red", CI_95="blue", CI_99="green")) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  theme(plot.title=element_text(hjust=0.5),
        plot.subtitle=element_text(hjust= 0.5),
        plot.caption=element_text(hjust=1.0),
        legend.key.width=unit(0.8,"cm"), legend.position="bottom")
```

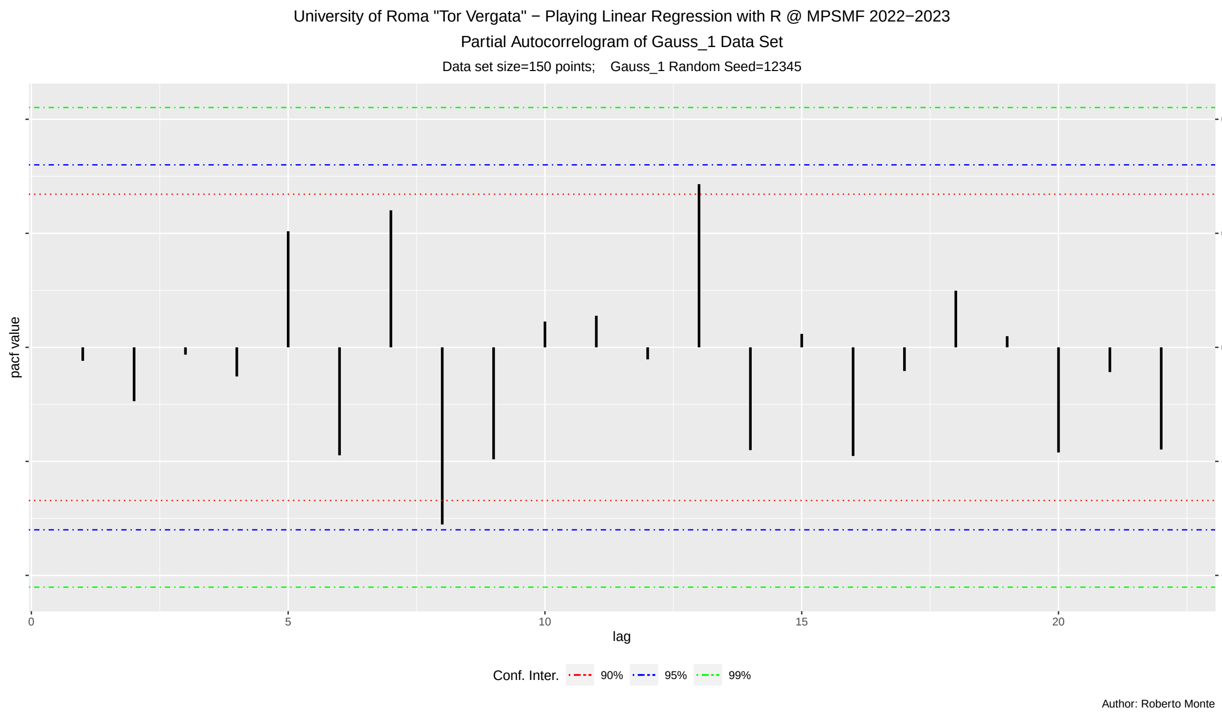
```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



The partial autocorrelogram of the data set *Gauss_1*.

```
z <- Gauss_df$Z_1
n <- length(z)
maxlag <- ceiling(10*log10(n))
P_Aut_Fun_z <- pacf(z, lag.max=maxlag, type="correlation", plot=FALSE)
ci_90 <- qnorm((1+0.90)/2)/sqrt(n)
ci_95 <- qnorm((1+0.95)/2)/sqrt(n)
ci_99 <- qnorm((1+0.99)/2)/sqrt(n)
Plot_P_Aut_Fun_z <- data.frame(lag=P_Aut_Fun_z$lag, pacf=P_Aut_Fun_z$acf)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u0
paste("Partial Autocorrelogram of Gauss_1 Data Set")))
subtitle_content <- bquote(paste("Data set size=", .(n), " points;    Gauss_1 Random Seed=12345"))
caption_content <- "Author: Roberto Monte"
ggplot(Plot_P_Aut_Fun_z, aes(x=lag, y=pacf)) +
  geom_segment(aes(x=lag, y=rep(0,length(lag)), xend=lag, yend=pacf), size=1, col="black") +
  geom_hline(aes(yintercept=-ci_90, color="CI_90"), show.legend=TRUE, lty=3) +
  geom_hline(aes(yintercept=ci_90, color="CI_90"), lty=3) +
  geom_hline(aes(yintercept=ci_95, color="CI_95"), show.legend=TRUE, lty=4) +
  geom_hline(aes(yintercept=-ci_95, color="CI_95"), lty=4) +
  geom_hline(aes(yintercept=-ci_99, color="CI_99"), show.legend=TRUE, lty=4) +
  geom_hline(aes(yintercept=ci_99, color="CI_99"), lty=4) +
  scale_x_continuous(name="lag", breaks=waiver(), label=waiver()) +
  scale_y_continuous(name="pacf value", breaks=waiver(), labels=NULL,
                     sec.axis=sec_axis(~., breaks=waiver(), labels=waiver())) +
  scale_color_manual(name="Conf. Inter.", labels=c("90%","95%","99%"),
                     values=c(CI_90="red", CI_95="blue", CI_99="green")) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  theme(plot.title=element_text(hjust=0.5),
        plot.subtitle=element_text(hjust= 0.5),
```

```
        plot.caption=element_text(hjust=1.0),
        legend.key.width=unit(0.8,"cm"), legend.position="bottom")
```

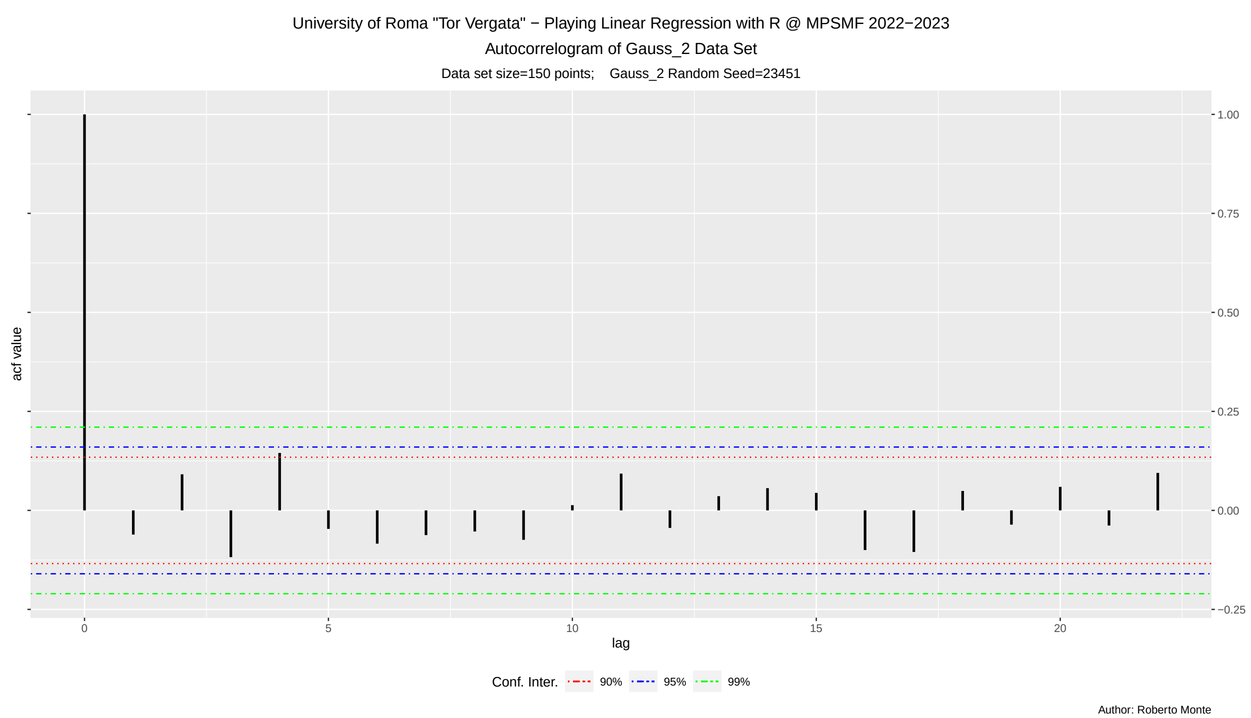


With reference to the *Gauss_1* autocorrelogram, the number of peaks corresponding to positive lags crossing the confidence lines is within the statistical tolerance. In fact, we have no peaks crossing the 95% confidence lines (the tolerance is *floor(maxlag* $*0.05$*)=floor(*$22*0.05$*)*=1 and only one peak crosses the 90% confidence lines (the tolerance is *floor(maxlag* $*0.10$*)=floor(*$22*0.10$*)*=2. With reference to the *Gauss_1* partial autocorrelogram, the number of peaks crossing the confidence lines is also within the statistical tolerance. In fact, we still have no peaks crossing the 95% confidence line (the tolerance is still *floor(maxlag* $*0.05$*)*=1*)* and only two peaks cross the 90% confidence line (the tolerance is still *floor(maxlag* $*0.10$*)*=2*)*. Therefore, we have visual evidence that the data set $Gauss_1$ has been generated by independent random sampling from the same distribution at the 90% confidence level.

The autocorrelogram of the *Gauss_2* data set.

```
z <- Gauss_df$Z_2
n <- length(z)
maxlag <- ceiling(10*log10(n))
Aut_Fun_z <- acf(z, lag.max=maxlag, type="correlation", plot=FALSE)
ci_90 <- qnorm((1+0.90)/2)/sqrt(n)
ci_95 <- qnorm((1+0.95)/2)/sqrt(n)
ci_99 <- qnorm((1+0.99)/2)/sqrt(n)
Plot_Aut_Fun_z <- data.frame(lag=Aut_Fun_z$lag, acf=Aut_Fun_z$acf)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u0
paste("Autocorrelogram of Gauss_2 Data Set")))
subtitle_content <- bquote(paste("Data set size=", .(n), " points;    Gauss_2 Random Seed=23451"))
caption_content <- "Author: Roberto Monte"
ggplot(Plot_Aut_Fun_z, aes(x=lag, y=acf)) +
  geom_segment(aes(x=lag, y=rep(0,length(lag)), xend=lag, yend=acf), size=1, col="black") +
  geom_hline(aes(yintercept=-ci_90, color="CI_90"), show.legend=TRUE, lty=3) +
  geom_hline(aes(yintercept=ci_90, color="CI_90"), lty=3) +
```

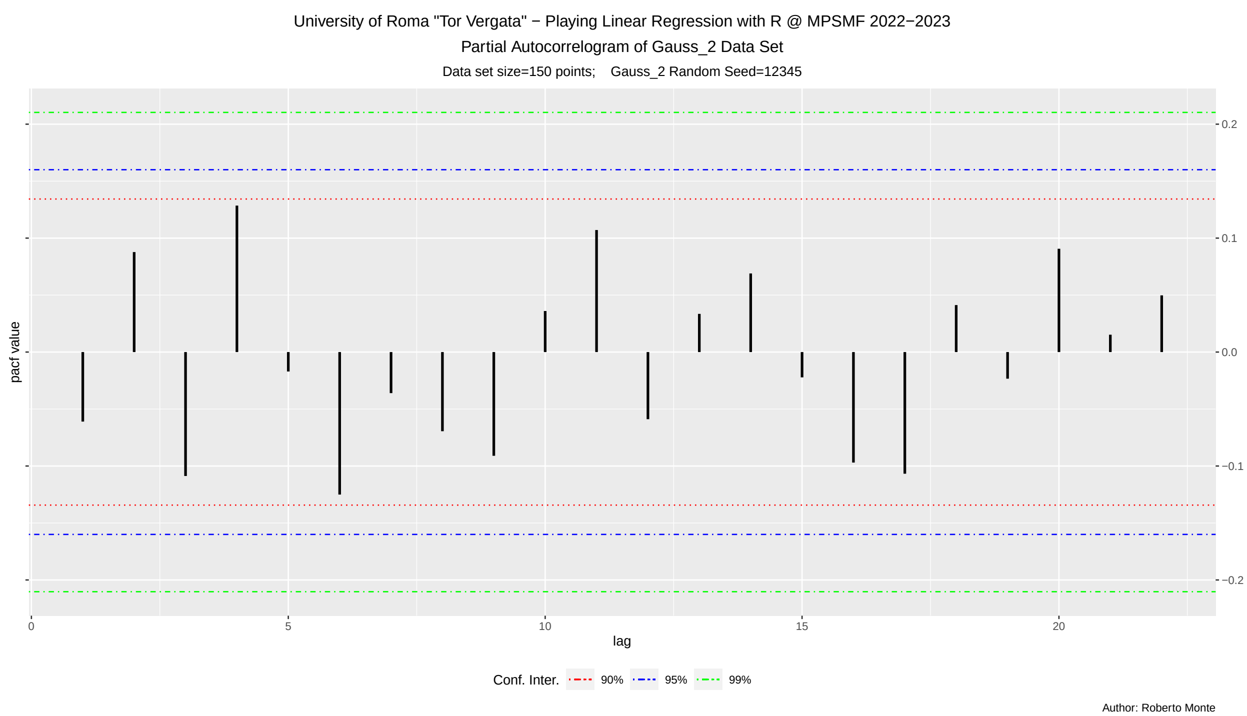
```
geom_hline(aes(yintercept=ci_95, color="CI_95"), show.legend=TRUE, lty=4) +
geom_hline(aes(yintercept=-ci_95, color="CI_95"), lty=4) +
geom_hline(aes(yintercept=-ci_99, color="CI_99"), show.legend=TRUE, lty=4) +
geom_hline(aes(yintercept=ci_99, color="CI_99"), lty=4) +
scale_x_continuous(name="lag", breaks=waiver(), label=waiver()) +
scale_y_continuous(name="acf value", breaks=waiver(), labels=NULL,
                   sec.axis=sec_axis(~., breaks=waiver(), labels=waiver())) +
scale_color_manual(name="Conf. Inter.", labels=c("90%","95%","99%"),
                   values=c(CI_90="red", CI_95="blue", CI_99="green")) +
ggtitle(title_content) +
labs(subtitle=subtitle_content, caption=caption_content) +
theme(plot.title=element_text(hjust=0.5),
      plot.subtitle=element_text(hjust= 0.5),
      plot.caption=element_text(hjust=1.0),
      legend.key.width=unit(0.8,"cm"), legend.position="bottom")
```



The partial autocorrelogram of the *Gauss_2* data set.

```
z <- Gauss_df$Z_2
n <- length(z)
maxlag <- ceiling(10*log10(n))
P_Aut_Fun_z <- pacf(z, lag.max=maxlag, type="correlation", plot=FALSE)
ci_90 <- qnorm((1+0.90)/2)/sqrt(n)
ci_95 <- qnorm((1+0.95)/2)/sqrt(n)
ci_99 <- qnorm((1+0.99)/2)/sqrt(n)
Plot_P_Aut_Fun_z <- data.frame(lag=P_Aut_Fun_z$lag, pacf=P_Aut_Fun_z$acf)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u0
paste("Partial Autocorrelogram of Gauss_2 Data Set")))
subtitle_content <- bquote(paste("Data set size=", .(n), " points;    Gauss_2 Random Seed=12345"))
caption_content <- "Author: Roberto Monte"
```

```
ggplot(Plot_P_Aut_Fun_z, aes(x=lag, y=pacf)) +
  geom_segment(aes(x=lag, y=rep(0,length(lag)), xend=lag, yend=pacf), size=1, col="black") +
  geom_hline(aes(yintercept=-ci_90, color="CI_90"), show.legend=TRUE, lty=3) +
  geom_hline(aes(yintercept=ci_90, color="CI_90"), lty=3) +
  geom_hline(aes(yintercept=ci_95, color="CI_95"), show.legend=TRUE, lty=4) +
  geom_hline(aes(yintercept=-ci_95, color="CI_95"), lty=4) +
  geom_hline(aes(yintercept=-ci_99, color="CI_99"), show.legend=TRUE, lty=4) +
  geom_hline(aes(yintercept=ci_99, color="CI_99"), lty=4) +
  scale_x_continuous(name="lag", breaks=waiver(), label=waiver()) +
  scale_y_continuous(name="pacf value", breaks=waiver(), labels=NULL,
                     sec.axis=sec_axis(~., breaks=waiver(), labels=waiver())) +
  scale_color_manual(name="Conf. Inter.", labels=c("90%","95%","99%"),
                     values=c(CI_90="red", CI_95="blue", CI_99="green")) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  theme(plot.title=element_text(hjust=0.5),
        plot.subtitle=element_text(hjust= 0.5),
        plot.caption=element_text(hjust=1.0),
        legend.key.width=unit(0.8,"cm"), legend.position="bottom")
```



With reference to the *Gauss_2* autocorrelogram and partial autocorrelogram, we can state similar considerations as for the *Gauss_1* autocorrelogram and partial autocorrelogram. Hence, we have also visual evidence that also the data set *Gauss_2* has been generated by independent random sampling from the same distribution at the 90% confidence level.

We can also consider a computational test, the Ljung-Box (LB) test, which assumes the null hypothesis that the data set is generated by independent random sampling from the same distribution. We have

```
z <- Gauss_df$Z_1
n <- length(z)
maxlag <- ceiling(10*log10(n))
```

```r
Gauss_1_LB <- Box.test(z, lag=maxlag, type="Ljung-Box")
show(Gauss_1_LB)
```

```
##
##  Box-Ljung test
##
## data:  z
## X-squared = 18.278, df = 22, p-value = 0.6894
```

and

```r
z <- Gauss_df$Z_2
n <- length(z)
maxlag <- ceiling(10*log10(n))
Gauss_2_LB <- Box.test(z, lag=maxlag, type="Ljung-Box")
show(Gauss_2_LB)
```

```
##
##  Box-Ljung test
##
## data:  z
## X-squared = 20.268, df = 22, p-value = 0.5663
```

As a consequence, for both the data sets *Gauss_1* and *Gauss_2*, we cannot reject the null hypothesis that each them has been generated by independent random sampling from the same distribution, at the significance level of 0.10.

Having checked the stationarity (in the mean), homoskedasticity (stationarity in the variance), and lack of correlation of the data sets *Gauss_1* and *Gauss_2*, it makes sense to apply a test for the analysis of independence and identical distribution. The Wald-Wolfowitz Runs test (WW) is a non-parametric test which returns the decision for the null hypothesis that the values in a dichotomous sequence of numerical or categorical data are generated by independent and identically distributed (iid) random variables, against the alternative that they are not generated by such a sequence of random variables. In other words, the null hypothesis states that each element in the sequence is independently drawn from the same distribution. For a sequence of numerical data, the test computes the runs above and below a reference value. Such a reference value can be set as the empirical mean of the sequence (default in MATLAB runstest) or the median (default in R runs.test) or it can be any user-defined value. However, note that the choice of the reference value will affect the result of the test. After setting the reference value, we center the sequence about the reference value and consider the binary sequence of the + and - signs of the elements of the centered sequence. The possible alternative values are “two.sided” (default), “left.sided” and “right.sided”. These define the alternative hypothesis. By using the alternative “left.sided” the null of randomness is tested against a trend. By using the alternative “right.sided” the null hypothesis of randomness is tested against a first order negative serial correlation.

```r
# library(randtests)
z <- Gauss_df$Z_1
Gauss_1_median_WW <- randtests::runs.test(z, alternative="two.sided", threshold=median(z), pvalue='norma
show(Gauss_1_median_WW)
```

```
##
##  Runs Test
##
## data:  z
```

```
## statistic = 0, runs = 76, n1 = 75, n2 = 75, n = 150, p-value = 1
## alternative hypothesis: nonrandomness
```

```
Gauss_1_mean_WW <- randtests::runs.test(z, alternative="two.sided", threshold=mean(z), pvalue='normal',
show(Gauss_1_mean_WW)
```

```
##
##  Runs Test
##
## data:  z
## statistic = 0.0087449, runs = 76, n1 = 77, n2 = 73, n = 150, p-value =
## 0.993
## alternative hypothesis: nonrandomness
```

From the result of the runs test we cannot reject the null assumption that the data set *Gauss_1* is generated by independent sampling from the same distribution at any standard significance level. The same result holds true for the data set *Gauss_2*.

```
# library(randtests)
z <- Gauss_df$Z_2
Gauss_2_median_WW <- randtests::runs.test(z, alternative="two.sided", threshold=median(z), pvalue='norma
show(Gauss_2_median_WW)
```

```
##
##  Runs Test
##
## data:  z
## statistic = 0.3277, runs = 78, n1 = 75, n2 = 75, n = 150, p-value =
## 0.7431
## alternative hypothesis: nonrandomness
```

```
Gauss_2_mean_WW <- randtests::runs.test(z, alternative="two.sided", threshold=mean(z), pvalue='normal',
show(Gauss_2_mean_WW)
```

```
##
##  Runs Test
##
## data:  z
## statistic = 0.66461, runs = 80, n1 = 73, n2 = 77, n = 150, p-value =
## 0.5063
## alternative hypothesis: nonrandomness
```

In light of the above results, we cannot reject the hypothesis that the data sets *Gauss_1* and *Gauss_2* have been generated by random sampling from the same distribution. Hence, we are left to guess such a generating distribution. Of course, the most direct approach would be to apply some normality test to check whether the data sets are normally distributed. However, we do not choose this approach since we are interested in presenting techniques that can be applied also in non Gaussian context.

As a first step we consider the statistical summary of the two standardized data sets.

```
z <- Gauss_df$Z_1
z_st <- (z-mean(z))/sd(z)
summary(z_st)
```

```
##      Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -2.26521 -0.73451  0.09543  0.00000  0.63976  2.22701
```

We also consider the empirical standard deviation, skewness and kurtosis which are not included in the summary.

```
Gauss_1_sd <- sd(z)
Gauss_1_skew <- EnvStats::skewness(z, method="moment")
Gauss_1_kurt <- EnvStats::kurtosis(z, method="moment", excess=TRUE)
Gauss_1_hm_df <- data.frame(stand.dev.= round(Gauss_1_sd,4), skew.=round(Gauss_1_skew,4),
                            exc.kurt. = round(Gauss_1_kurt,4))
print.data.frame(Gauss_1_hm_df, row.names = FALSE)
```

```
##  stand.dev.   skew. exc.kurt.
##      1.1211 -0.0216   -0.6725
```

Note that the six summary points of the *Gauss_1* data set, except perhaps the 3rd quartile, do not fit perfectly the corresponding summary points of the standard Gaussian distribution. In fact, for the standard Gaussian distribution we have the following summary points (to be compared to the *Gauss_1* summary points in the last row):

$$\begin{array}{cccccc} Min\,(99.73\%) & 1stQ & Median & Mean & 3rdQ & Max\,(99.73\%) \\ -3.0000 & -0.6745 & 0.0000 & 0.0000 & 0.6745 & 3.0000 \\ -2.2652 & -0.7345 & 0.0954 & 0.0000 & 0.6398 & 2.2270 \end{array} \tag{114}$$

where

$$\begin{aligned} 1stQ &= qnorm(0.25,\ mean = 0,\ sd = 1,\ lower.tail = TRUE), \\ 3rdQ &= qnorm(0.75,\ mean = 0,\ sd = 1,\ lower.tail = TRUE), \\ Min\,(99.73\%) &= Mean - 3 * sd, \qquad Max\,(99.73\%) = Mean + 3 * sd. \end{aligned} \tag{115}$$

In addition, the skewness and the excess kurtosis of the standard Gaussian distribution are both equal to 0.

A similar result we have for the *Gauss_2* data set.

```
z <- Gauss_df$Z_2
z_st <- (z-mean(z))/sd(z)
summary(z_st)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -2.9317 -0.5922 -0.1119  0.0000  0.6962  2.7625
```

```
Gauss_2_sd <- sd(z)
Gauss_2_skew <- EnvStats::skewness(z, method="moment")
Gauss_2_kurt <- EnvStats::kurtosis(z, method="moment", excess=TRUE)
show(c(round(Gauss_2_sd,4), round(Gauss_2_skew,4), round(Gauss_2_kurt,4)))
```

```
## [1] 0.9631 0.0325 0.1237
```

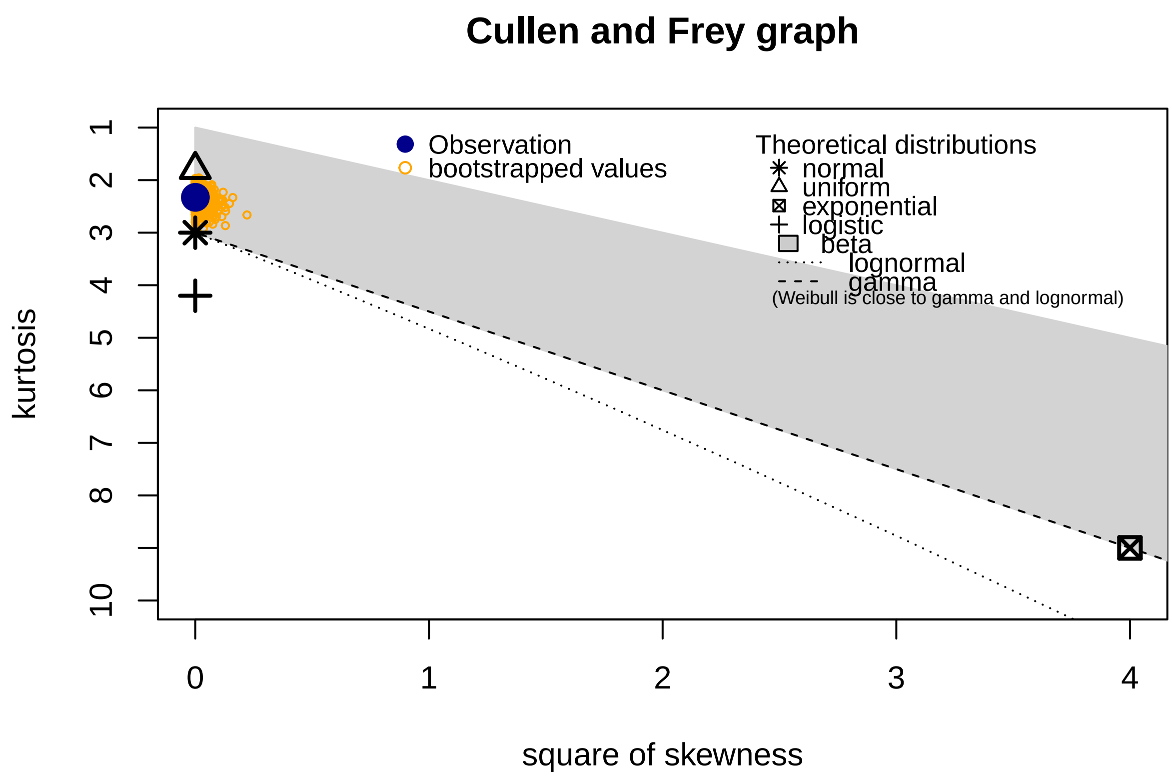
Note that also the six summary points of the *Gauss_2* data set do not fit perfectly the corresponding summary points of the standard Gaussian distribution. However, the fit is rather good.

$$\begin{array}{cccccc} Min\,(99.73\%) & 1stQ & Median & Mean & 3rdQ & Max\,(99.73\%) \\ -3.0000 & -0.6745 & 0.0000 & 0.0000 & 0.6745 & 3.0000 \\ -2.9317 & -0.5922 & -0.1119 & 0.0000 & 0.6962 & 2.7625 \end{array} \tag{116}$$

A Cullen-Frey graph can help to understand better the relationship between the distributions of the data sets *Gauss_1*, *Gauss_2* and a theoretical distribution of reference.

The Cullen-Frey graph for the data set *Gauss_1*.

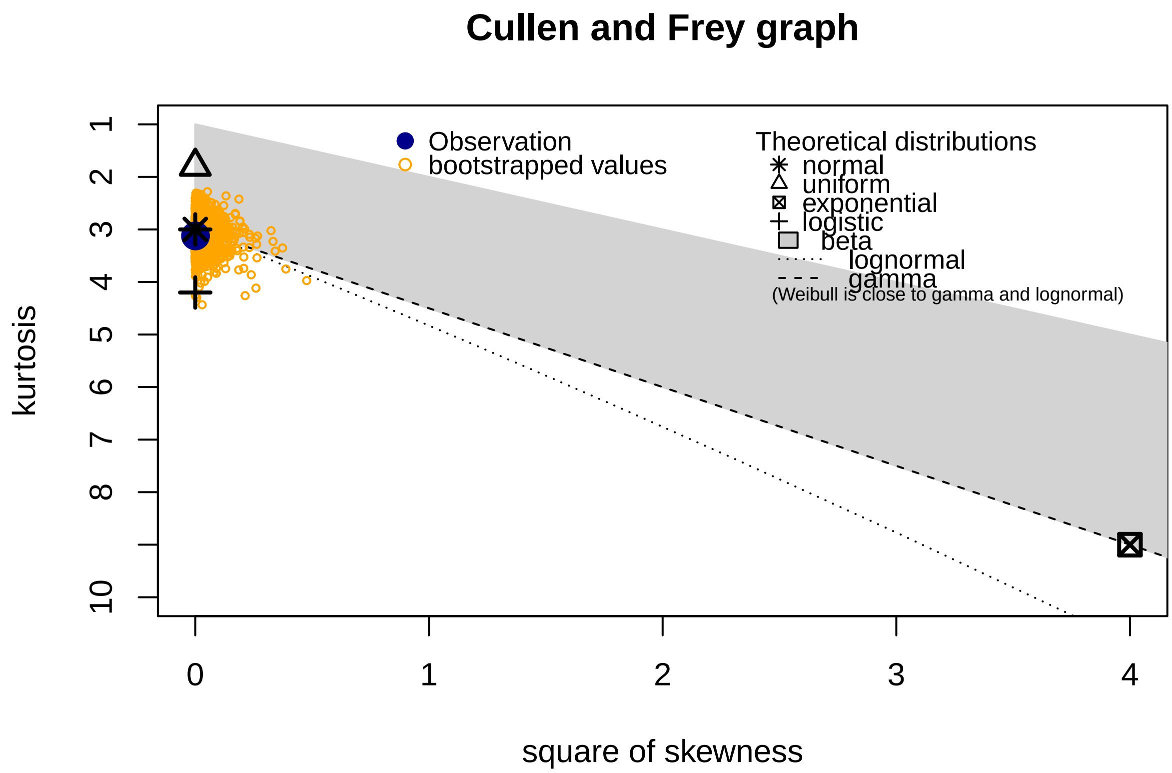
```
# library(fitdistrplus)
z <- Gauss_df$Z_1
descdist(z, discrete = FALSE, method = "sample", graph = TRUE, boot=1000)
```




```
## summary statistics
## ------
## min:  -2.380358   max:  2.655788
## median:  0.2661124
## mean:  0.1591299
## sample sd:  1.117339
## sample skewness:  -0.02162206
## sample kurtosis:  2.327549
```

The Cullen-Frey graph for the data set *Gauss_2*.

```
# library(fitdistrplus)
z <- Gauss_df$Z_2
descdist(z, discrete = FALSE, method = "sample", graph = TRUE, boot=1000)
```




```
## summary statistics
## ------
## min:  -2.773087   max:  2.710826
## median:  -0.05741142
## mean:  0.0503411
## sample sd:  0.9598529
## sample skewness:  0.03254984
## sample kurtosis:  3.123747
```

From the inspection of the Cullen-Frey graphs the suspect arises that the standardized data sets *Gauss_1* and *Gauss_2* might be Gaussian distributed. In particular, for the latter the suspect is rather strong.

Then, We draw the *Q-Q* and the *P-P* plot for the data sets *Gauss_1* and *Gauss_2* against the standard Gaussian distribution. To this we use the library *qqplotr*, which extends some functionality of the library *ggplot2*.

To draw the *Q-Q* plot of the data sets *Gauss_1* and *Gauss_2* against the standard Gaussian distribution, we need to draw the scatter plot of the empirical quantiles of the data sets *Gauss_1* and *Gauss_2* against the corresponding quantiles of the standard Gaussian distribution. In turn, to generate the empirical quantiles of the data sets *Gauss_1* and *Gauss_2* and the corresponding quantiles of the standard Gaussian distribution we need to consider a probability vector of size equal to the size of the data sets *Gauss_1* and *Gauss_2* drawn from the unifom probability distribution. In R, the standard approach is to consider the probability vector $(p_k)_{k=1}^n$ given by

$$p_k \stackrel{\text{def}}{=} \frac{k-1/2}{n}, \quad \forall k = 1, \ldots, n, \tag{117}$$

That is

```
z <- Gauss_df$Z_1
n <- length(z)
unif_probs <- seq(from=(0.5/n), to=(1-(0.5/n)), by=(1/n))
show(unif_probs[1:15])
```

```
##  [1] 0.003333333 0.010000000 0.016666667 0.023333333 0.030000000 0.036666667
##  [7] 0.043333333 0.050000000 0.056666667 0.063333333 0.070000000 0.076666667
## [13] 0.083333333 0.090000000 0.096666667
```

```
# Equivalently the desired probability vector can be generated by the function *ppoints*
ppoints <- ppoints(n)
show(ppoints[1:15])
```

```
##  [1] 0.003333333 0.010000000 0.016666667 0.023333333 0.030000000 0.036666667
##  [7] 0.043333333 0.050000000 0.056666667 0.063333333 0.070000000 0.076666667
## [13] 0.083333333 0.090000000 0.096666667
```

```
all(round(unif_probs, digits=15)==round(ppoints, digits=15))
```

```
## [1] TRUE
```

Hence, we can build a suitable data frame to draw the *Q-Q* plot for the .

```
z <- Gauss_df$Z_1                                         # Gauss_1 data set.
z_cent <- z - mean(z)                               # Centered Gauss_1 data set.
z_cent_qemp <- qemp(ppoints(length(z)), z_cent) # Empirical quantiles of the centered Gauss_1 data set.
Gauss_quants <- qnorm(ppoints(length(z)), mean=0, sd=1) # Quantiles of the standard Gaussian distributi
                                                          # corresponding to the empirical quantiles
Gausss_1_QQ_plot_df <- data.frame(k=1:length(z), S=z_cent, X=Gauss_quants, Y=z_cent_qemp)
head(Gausss_1_QQ_plot_df)
```

```
##   k          S         X         Y
## 1 1  0.4263990 -2.713052 -2.539488
## 2 2  0.5503362 -2.326348 -2.510167
## 3 3 -0.2684332 -2.128045 -2.309673
## 4 4 -0.6126270 -1.989313 -2.037648
## 5 5  0.4467576 -1.880794 -1.993034
## 6 6 -1.9770858 -1.790751 -1.978756
```

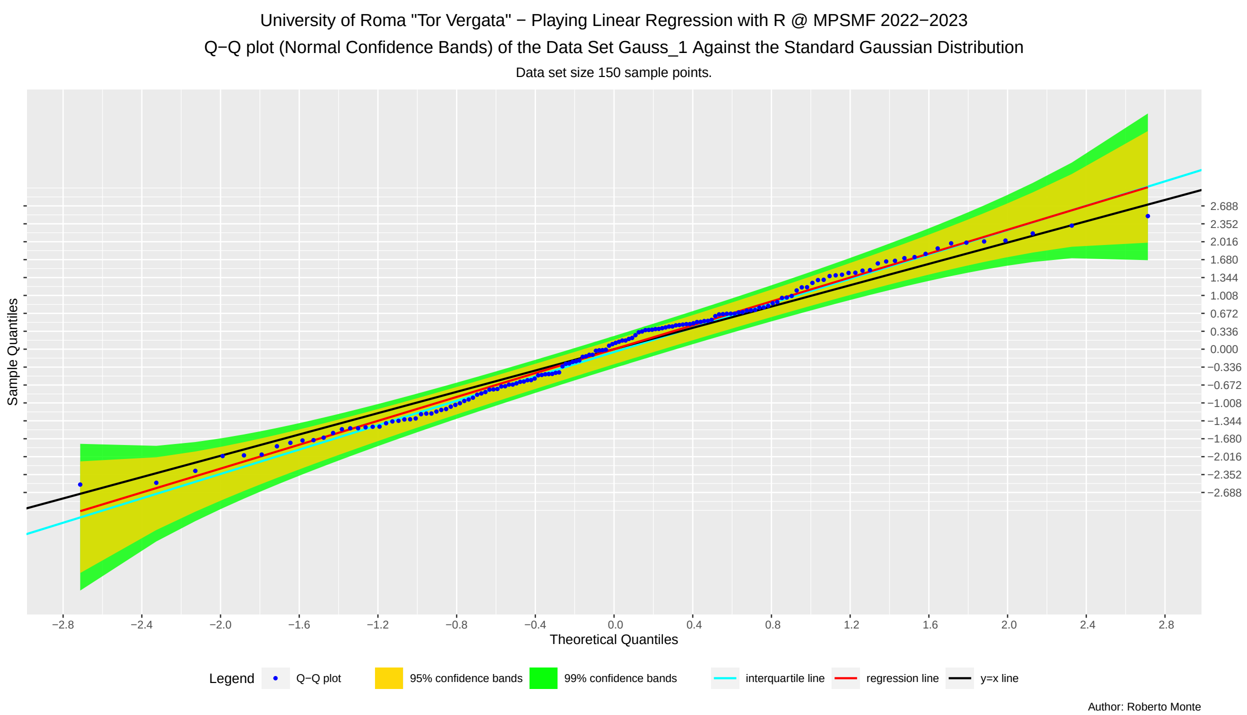
Then, we build the *Q-Q* plot with normal confidence bands.

```
Data_df <- Gausss_1_QQ_plot_df
n <- nrow(Data_df)
quart_probs <- c(0.25,0.75)
quart_Y <- as.vector(quantile(Data_df$Y, quart_probs))
quart_X <- qnorm(quart_probs, mean=0, sd=1)
slope <- diff(quart_Y)/diff(quart_X)
intercept <- quart_Y[1]-slope*quart_X[1]
distr <- "norm"
distr_pars <- list(mean=0, sd=1)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u0
```

```
paste("Q-Q plot (Normal Confidence Bands) of the Data Set Gauss_1 Against the Standard Gaussian Distribu
subtitle_content <- bquote(paste("Data set size ", .(n), " sample points."))
caption_content <- "Author: Roberto Monte"
x_name <- bquote("Theoretical Quantiles")
y_name <- bquote("Sample Quantiles")
x_breaks_num <- 15 # (deduced from primeFactors(n))
# x_breaks_num <- ceiling(n^(1/2)) # Tukey & Mosteller square-root rule
# x_breaks_num <- ceiling(1+log2(n)) # Sturges rule
# x_breaks_num <- ceiling((2*n)^(1/3)) # Teller & Scott rice rule
x_binwidth <- round((max(Data_df$X)-min(Data_df$X))/x_breaks_num, digits=1)
x_breaks_low <- floor((min(Data_df$X)/x_binwidth))*x_binwidth
x_breaks_up <- ceiling((max(Data_df$X)/x_binwidth))*x_binwidth
x_breaks <- c(seq(from=x_breaks_low, to=x_breaks_up, by=x_binwidth))
x_labs <- format(x_breaks, scientific=FALSE)
J <- 1.0
x_lims <- c((x_breaks_low-J*x_binwidth), (x_breaks_up+J*x_binwidth))
y_breaks_num <- length(x_breaks)
y_binwidth <- round((max(Data_df$Y)-min(Data_df$Y))/y_breaks_num, digits=3)
y_breaks_low <- floor((min(Data_df$Y)/y_binwidth))*y_binwidth
y_breaks_up <- ceiling((max(Data_df$Y)/y_binwidth))*y_binwidth
y_breaks <- c(round(seq(from=y_breaks_low, to=y_breaks_up, by=y_binwidth),3))
y_labs <- format(y_breaks, scientific=FALSE)
K <- 1.5
y_lims <- c((y_breaks_low-K*y_binwidth), (y_breaks_up+K*y_binwidth))
y1_shape <- bquote("Q-Q plot")
y1_fill <- bquote("95% confidence bands")
y2_fill <- bquote("99% confidence bands")
col_1 <- bquote("interquartile line")
col_2 <- bquote("regression line")
col_3 <- bquote("y=x line")
leg_shape_labs <- y1_shape
leg_fill_labs <- c(y1_fill, y2_fill)
leg_col_labs <- c(col_1, col_2, col_3)
leg_shape_cols <- c("y1_shape" = 19)
leg_fill_cols <- c("y1_fill"="gold", "y2_fill"="green")
leg_col_cols <- c("col_1"="cyan", "col_2"="red", "col_3"="black")
leg_shape_sort <- "y1_shape"
leg_fill_sort <- c("y1_fill", "y2_fill")
leg_col_sort <- c("col_1", "col_2", "col_3")
Gauss_1_QQ_norm_plot <- ggplot(Data_df, aes(sample=S)) +
  stat_qq_band(aes(fill="y2_fill"), distribution=distr, dparams=distr_pars, conf = 0.99, bandType = "poi
  stat_qq_band(aes(fill="y1_fill"), distribution=distr, dparams=distr_pars, conf = 0.95, bandType = "poi
#  stat_qq_line(aes(colour="col_1"), distribution=distr, dparams=distr_pars) +
  geom_abline(aes(slope=slope, intercept=intercept, colour="col_1"), size=0.8, linetype="solid", show.le
#  geom_segment(aes(x=Q[1], xend=-Q[1], y=Q[1], yend=-Q[1], colour="col_3"),
#               size=0.8, linetype="solid", show.legend=FALSE) +
  geom_abline(aes(slope=1, intercept=0, colour="col_3"), size=0.8, linetype="solid", show.legend=FALSE)
  stat_smooth(alpha=1, size=0.8, linetype="solid", aes(x=X, y=Y, colour="col_2"),
            method="lm" , formula=y~x, se=FALSE, fullrange=FALSE) +
  stat_qq_point(aes(shape="y1_shape"), colour="blue", alpha=1, size=1.0,
               distribution=distr, dparams=distr_pars) +
  scale_x_continuous(name=x_name, breaks=x_breaks, label=x_labs, limits=NULL) +
  scale_y_continuous(name=y_name, breaks=y_breaks, labels=NULL, limits=NULL,
```

```
                    sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  scale_shape_manual(name="Legend", labels=leg_shape_labs, values=leg_shape_cols, breaks=leg_shape_sort)
  scale_fill_manual(name="", labels=leg_fill_labs, values=leg_fill_cols, breaks=leg_fill_sort) +
  scale_colour_manual(name="", labels=leg_col_labs, values=leg_col_cols, breaks=leg_col_sort) +
  guides(shape=guide_legend(order=1), fill=guide_legend(order=2), colour=guide_legend(order=3)) +
  theme(plot.title=element_text(hjust=0.5, size=13.5), plot.subtitle=element_text(hjust=0.5),
        axis.text.x=element_text(angle=0, vjust=1),
        legend.key.width=unit(0.8,"cm"), legend.position="bottom")
plot(Gauss_1_QQ_norm_plot)
```

```
## Warning: The following aesthetics were dropped during statistical transformation: sample
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a `group` aesthetic or to convert a numerical
##   variable into a factor?
```



Note the confidence bands in the Q-Q plot are drawn on the basis of the following argument.

Let $X : \Omega \rightarrow \mathbb{R}$ be an absolutely continuous real random variable on a probability space $(\Omega, \mathcal{E}, \mathbf{P}) \equiv \Omega$, with distribution function $F_X : \mathbb{R} \rightarrow \mathbb{R}$, density function $f_X : \mathbb{R} \rightarrow \mathbb{R}$, and quantile function $Q_X : (0,1) \rightarrow \mathbb{R}$. It is well known that the kth order statistic $X_{(k)} : \Omega \rightarrow \mathbb{R}$ of a simple random sample $X_1, \ldots, X_n$ drawn from X is absolutely continuous with distribution function $F_{X_{(k)}} : \mathbb{R} \rightarrow \mathbb{R}$ and density function $f_{X_{(k)}} : \mathbb{R} \rightarrow \mathbb{R}$, respectively given by

$$F_{X_{(k)}}(x) = \sum_{\ell=k}^{n} \binom{n}{\ell} F_X^{\ell}(x)\,(1 - F_X(x))^{n-\ell} \quad \text{and} \quad f_{X_{(k)}}(x) = \frac{n!}{(k-1)!\,(n-k)!} F_X^{k-1}(x)\,(1 - F_X(x))^{n-k} f_X(x), \tag{118}$$

for every $x \in \mathbb{R}$ and $k = 1, \dots, n$. Assuming that $F_X : \mathbb{R} \to \mathbb{R}$ is strictly increasing, we also know that

$$Q_X = F_X^{-1} \tag{119}$$

and the kth order statistic $X_{(k)} : \Omega \to \mathbb{R}$ is asymptotically normal, for every $k = 1, \dots, n$. More specifically, we have

$$X_{(k)} \sim AN\left(Q_X\left(p_k\right), \frac{p_k\left(1-p_k\right)}{n f_X\left(Q_X\left(p_k\right)\right)^2}\right), \tag{120}$$

where $p_k \equiv \frac{k}{n+1}$, for $k = 1, \dots, n$. Thus, the standard error of the kth order statistic $X_{(k)} : \Omega \to \mathbb{R}$ can be estimated by

$$\mathbf{D}\left[X_{(k)}\right] \approx \frac{1}{f_X\left(x_k\right)}\sqrt{\frac{p_k\left(1-p_k\right)}{n}}, \tag{121}$$

where $x_k \equiv Q_X\left(p_k\right)$, for $k = 1, \dots, n$. As a consequence, referring to the Q-Q plot of a random variable $Y : \Omega \to \mathbb{R}$ against an absolutely continuous random variable $X : \Omega \to \mathbb{R}$ and considering the estimator $\hat{Y} : \Omega \to \mathbb{R}$ of Y via the linear regression, given by

$$\hat{Y} \overset{\text{def}}{=} \hat{\beta}_0 + \hat{\beta}_1 X, \tag{122}$$

where $\hat{\beta}_0$ [resp. $\hat{\beta}_1$] is the (estimated) intercept [resp. slope] of the regression line, under the assumption that X and Y have the same distribution, an approximate $100\left(1-\alpha\right)\%$ confidence interval for the kth order statistic $Y_{(k)} : \Omega \to \mathbb{R}$ of a simple random sample $Y_1, \dots, Y_n$ drawn from Y is given by

$$\left(\hat{Y}_{(k)} - z_{\alpha/2}\frac{1}{f_{\hat{Y}}\left(y_k\right)}\sqrt{\frac{p_k\left(1-p_k\right)}{n}}, \hat{Y}_{(k)} - z_{\alpha/2}\frac{1}{f_{\hat{Y}}\left(y_k\right)}\sqrt{\frac{p_k\left(1-p_k\right)}{n}}\right), \tag{123}$$

where $\hat{Y}_{(k)} : \Omega \to \mathbb{R}$ is the kth order statistic of a simple random sample $\hat{Y}_1, \dots, \hat{Y}_n$ drawn from the estimator $\hat{Y}$ with distribution function $F_{\hat{Y}} : \mathbb{R} \to \mathbb{R}$, density function $f_{\hat{Y}} : \mathbb{R} \to \mathbb{R}$, quantile function $Q_{\hat{Y}} : (0, 1) \to \mathbb{R}$, and $y_k \equiv Q_{\hat{Y}}\left(p_k\right)$, for $k = 1, \dots, n$. On the other hand, we have

$$F_{\hat{Y}}\left(y\right) = \mathbf{P}\left(\hat{Y} \leq y\right) = \mathbf{P}\left(\hat{\beta}_0 + \hat{\beta}_1 X \leq y\right) = \mathbf{P}\left(X \leq \frac{y - \hat{\beta}_0}{\hat{\beta}_1}\right) = F_X\left(\frac{y - \hat{\beta}_0}{\hat{\beta}_1}\right), \tag{124}$$

for every $y \in \mathbb{R}$. This implies

$$f_{\hat{Y}}\left(y\right) = \frac{d}{dy}F_{\hat{Y}}\left(y\right) = \frac{d}{dy}F_X\left(\frac{y - \hat{\beta}_0}{\hat{\beta}_1}\right) = \frac{1}{\hat{\beta}_1}f_X\left(\frac{y - \hat{\beta}_0}{\hat{\beta}_1}\right), \tag{125}$$

for every $y \in \mathbb{R}$. Therefore, the realization of the confidence interval in Equation (123) is given by

$$\left(\hat{\beta}_0 + \hat{\beta}_1 x_k - z_{\alpha/2}\frac{\hat{\beta}_1}{f_X\left(x_k\right)}\sqrt{\frac{p_k\left(1-p_k\right)}{n}}, \hat{\beta}_0 + \hat{\beta}_1 x_k + z_{\alpha/2}\frac{\hat{\beta}_1}{f_X\left(x_k\right)}\sqrt{\frac{p_k\left(1-p_k\right)}{n}}\right), \tag{126}$$

for every $k = 1, \dots, n$.

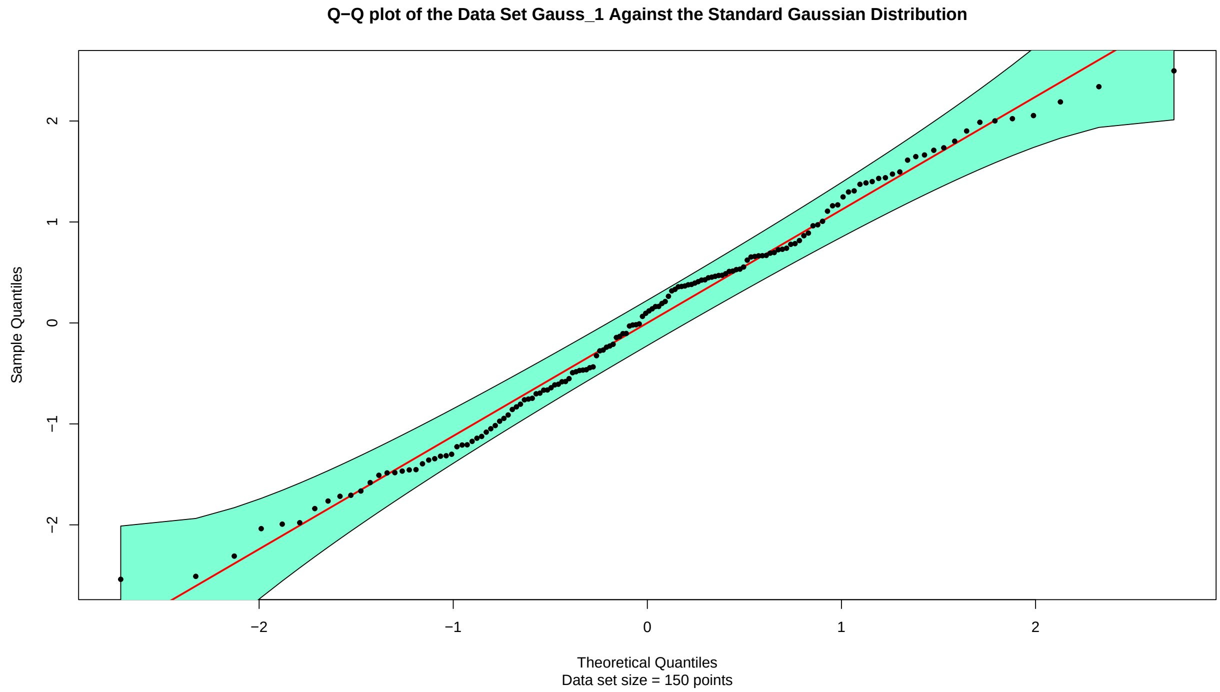
For additional details, see Fox, J. (2008), Applied Regression Analysis and Generalized Linear Models, 3nd Ed., Sage Publications, Inc.

```
z <- Gauss_df$Z_1                                       # Gauss_1 data set.
z_cent <- z - mean(z)                             # Centered Gauss_1 data set.
z_cent_qemp <- qemp(ppoints(length(z)), z_cent) # Empirical quantiles of the centered Gauss_1 data set.
Gauss_quants <- qnorm(ppoints(length(z)), mean=0, sd=1) # Quantiles of the standard Gaussian distributi
                                                        # corresponding to the empirical quantiles.
Gauss_probs <- pnorm(Gauss_quants, mean=0, sd=1)# Probabilities of the standard Gaussian distribution
```

```
                                                       # corresponding to the empirical probabilitie
X <- Gauss_quants
P <- Gauss_probs
Y <- z_cent_qemp
coef <- coefficients(lm(Y~X))    # Coefficients of the linear regression of Y agaiinst X.
a <-coef[1]                                # Intercept of the regression line.
b <-coef[2]                                # Slope of the regression line.
Y_hat <- as.numeric(fitted.values(lm(Y~X)))# Fitted values of the regression line.
# Y_hat <- a+b*X                             # Equivalent expression for the fitted values.
signif <- 0.05                               # Significance level corresponding to the confidence level
cv_z <- qnorm(1-signif/2, mean=0, sd=1)# Critical value corresponding to the significance level
SE <- (b/dnorm(X))*sqrt(P*(1-P)/n) # Standard errors of the estimator Y_hat
lower_95_conf_int <- Y_hat - cv_z*SE         # lower bound of the confidence intervals
upper_95_conf_int <- Y_hat + cv_z*SE         # upper bound of the confidence intervals
```

We draw a draft of the *Q-Q* plot with the confidence band that we have determined above.

```
# Draft of Q-Q plot + regression line + confidence bands
plot(X,Y, main="Q-Q plot of the Data Set Gauss_1 Against the Standard Gaussian Distribution",
          sub="Data set size = 150 points", xlab="Theoretical Quantiles", ylab="Sample Quantiles", type=
# lines(X,upper_95_conf_int,lty=2,lwd=2,col="blue")
# lines(X,lower_95_conf_int,lty=2,lwd=2,col="blue")
polygon(c(X,rev(X)),c(upper_95_conf_int,rev(lower_95_conf_int)), col="aquamarine")
abline(a,b, col="red", lwd=2)
points(X,Y, col="black", pch=20)
```



We can also draw the *Q-Q* plot with Kolmogorov-Smirnov confidence bands.

```
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u0
paste("Q-Q plot (Kolmogorov-Smirnov Bands) of the Data Set Gauss_1 Against the Standard Gaussian Distri
```

```r
subtitle_content <- bquote(paste("Data set size ", .(n), " sample points."))
Gauss_1_QQ_KS_plot <- ggplot(Data_df, aes(sample=S)) +
  stat_qq_band(aes(fill="y2_fill"), distribution=distr, dparams=distr_pars, conf = 0.99, bandType = "ks
  stat_qq_band(aes(fill="y1_fill"), distribution=distr, dparams=distr_pars, conf = 0.95, bandType = "ks
  geom_abline(aes(slope=slope, intercept=intercept, colour="col_1"), size=0.8, linetype="solid", show.le
  stat_smooth(alpha=1, size=0.8, linetype="solid", aes(x=X, y=Y, colour="col_2"),
              method="lm" , formula=y~x, se=FALSE, fullrange=FALSE) +
  geom_abline(aes(slope=1, intercept=0, colour="col_3"),
              size=0.8, linetype="solid", show.legend=FALSE) +
  stat_qq_point(aes(shape="y1_shape"), colour="blue", alpha=1, size=1.0,
               distribution=distr, dparams=distr_pars) +
  scale_x_continuous(name=x_name, breaks=x_breaks, label=x_labs, limits=NULL) +
  scale_y_continuous(name=y_name, breaks=y_breaks, labels=NULL, limits=NULL,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  scale_shape_manual(name="Legend", labels=leg_shape_labs, values=leg_shape_cols, breaks=leg_shape_sort)
  scale_fill_manual(name="", labels=leg_fill_labs, values=leg_fill_cols, breaks=leg_fill_sort) +
  scale_colour_manual(name="", labels=leg_col_labs, values=leg_col_cols, breaks=leg_col_sort) +
  guides(shape=guide_legend(order=1), fill=guide_legend(order=2), colour=guide_legend(order=3)) +
  theme(plot.title=element_text(hjust=0.5, size=13.5), plot.subtitle=element_text(hjust=0.5),
        axis.text.x=element_text(angle=0, vjust=1),
        legend.key.width=unit(0.8,"cm"), legend.position="bottom")
plot(Gauss_1_QQ_KS_plot)
```

```
## Warning: The following aesthetics were dropped during statistical transformation: sample
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a `group` aesthetic or to convert a numerical
##   variable into a factor?
```

Now, we draw the P–P plot with bootstrap confidence bands. First, we build a suitable data frame

```
z <- Gauss_df$Z_1                                  # Gauss_1 data set.
z_st <- (z-mean(z))/sd(z)          # Standardized Gauss_1 data set.
z_st_qemp <- qemp(ppoints(length(z_st)), z_st)# Empirical quantiles of the standardized Gauss_1 data se
z_st_pemp <- pemp(z_st_qemp, z_st)  # Empirical probabilities of the standardized Gauss_1 data set
                                                   # corresponding to the empirical quantiles.
Gauss_quants <- qnorm(ppoints(length(z)), mean=0, sd=1) # Quantiles of the standard Gaussian distributi
                                                        # corresponding to the empirical quantiles.
Gauss_probs <- pnorm(Gauss_quants, mean=0, sd=1)# Probabilities of the standard Gaussian distribution
                                                        # corresponding to the empirical probabilities.
Gausss_1_PP_plot_df <- data.frame(k=1:length(z), S=z_st, X=Gauss_probs, Y=z_st_pemp)
head(Gausss_1_PP_plot_df)
```

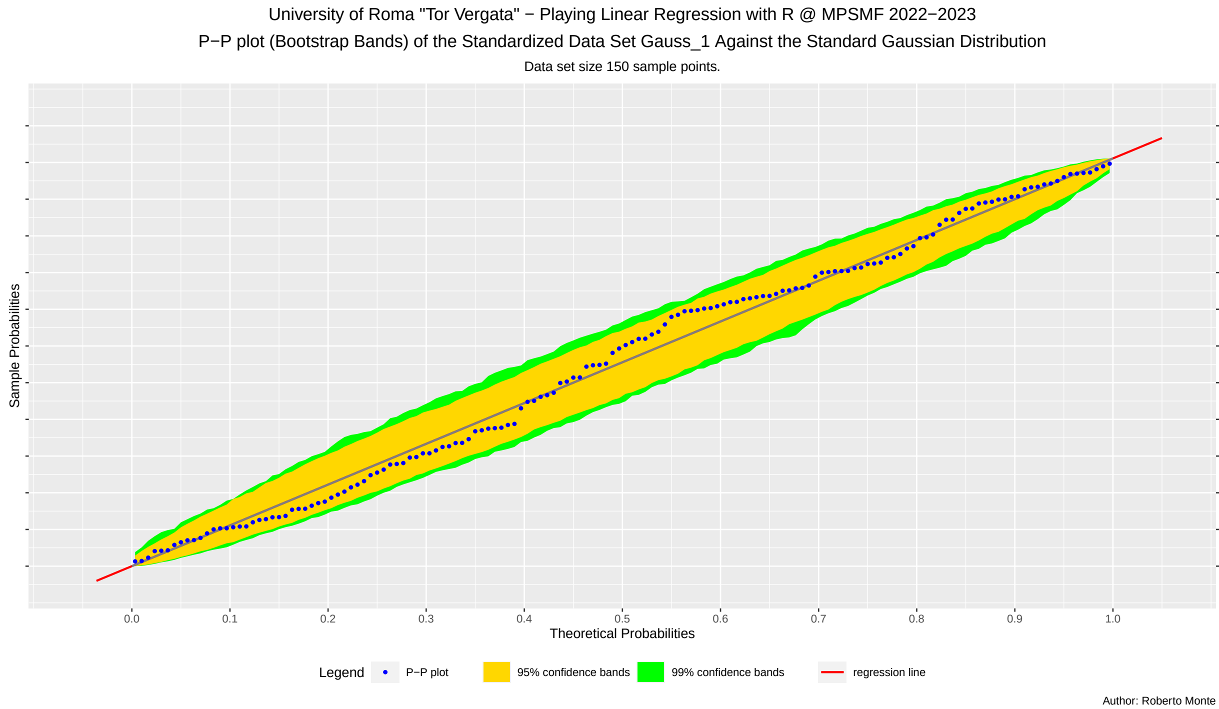
```
##   k          S           X           Y
## 1 1  0.3803459 0.003333333 0.004159734
## 2 2  0.4908973 0.010000000 0.010000000
## 3 3 -0.2394411 0.016666667 0.016666667
## 4 4 -0.5464605 0.023333333 0.023333333
## 5 5  0.3985057 0.030000000 0.030000000
## 6 6 -1.7635514 0.036666667 0.036666667
```

Second we draw the P–P plot of the residuals with boot bands (the only option available).

```
Data_df <- Gausss_1_PP_plot_df
n <- nrow(Data_df)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u00
paste("P-P plot (Bootstrap Bands) of the Standardized Data Set Gauss_1 Against the Standard Gaussian Di
subtitle_content <- bquote(paste("Data set size ", .(n), " sample points."))
caption_content <- "Author: Roberto Monte"
x_name <- bquote("Theoretical Probabilities")
y_name <- bquote("Sample Probabilities")
x_breaks_num <- 15 # (deduced from primeFactors(n))
# x_breaks_num <- ceiling(n^(1/2)) # Tukey & Mosteller square-root rule
# x_breaks_num <- ceiling(1+log2(n)) # Sturges rule
# x_breaks_num <- ceiling((2*n)^(1/3)) # Teller & Scott rice rule
x_binwidth <- round((max(Data_df$X)-min(Data_df$X))/x_breaks_num, digits=1)
x_breaks_low <- floor((min(Data_df$X)/x_binwidth))*x_binwidth
x_breaks_up <- ceiling((max(Data_df$X)/x_binwidth))*x_binwidth
x_breaks <- c(seq(from=x_breaks_low, to=x_breaks_up, by=x_binwidth))
# x_breaks <- c(1,round(seq(from=x_breaks_low, to=x_breaks_up, by=x_binwidth),3),n)
x_labs <- format(x_breaks, scientific=FALSE)
J <- 0.5
x_lims <- c(x_breaks_low-J*x_binwidth, x_breaks_up+J*x_binwidth)
y_breaks_num <- length(x_breaks)
y_binwidth <- round((max(Data_df$Y)-min(Data_df$Y))/y_breaks_num, digits=3)
y_breaks_low <- floor((min(Data_df$Y)/y_binwidth))*y_binwidth
y_breaks_up <- ceiling((max(Data_df$Y)/y_binwidth))*y_binwidth
y_breaks <- c(round(seq(from=y_breaks_low, to=y_breaks_up, by=y_binwidth),3))
y_labs <- format(y_breaks, scientific=FALSE)
K <- 0.5
y_lims <- c(y_breaks_low-K*y_binwidth, y_breaks_up+K*y_binwidth)
y1_shape <- bquote("P-P plot")
y1_fill <- bquote("95% confidence bands")
```

```r
y2_fill <- bquote("99% confidence bands")
col_1 <- bquote("y=x line")
col_2 <- bquote("regression line")
leg_shape_labs <- y1_shape
leg_fill_labs <- c(y1_fill, y2_fill)
leg_col_labs <- c(col_1, col_2)
leg_shape_cols <- c("y1_shape" = 19)
leg_fill_cols <- c("y1_fill"="gold", "y2_fill"="green")
leg_col_cols <- c("col_1"="black", "col_2"="red")
leg_shape_sort <- "y1_shape"
leg_fill_sort <- c("y1_fill", "y2_fill")
leg_col_sort <- c("col_1", "col_2")
distr <- "norm"
distr_pars <- list(mean=0, sd=1)
Gauss_1_PP_boot_plot <- ggplot(Data_df, aes(sample=S)) +
  stat_pp_band(aes(fill="y2_fill"), distribution=distr, dparams=distr_pars, conf = 0.99, bandType = "boo
  stat_pp_band(aes(fill="y1_fill"), distribution=distr, dparams=distr_pars, conf = 0.95, bandType = "boo
  stat_smooth(alpha=1, size=0.8, linetype="solid", aes(x=X, y=Y, colour="col_2"),
              method="lm" , formula=y~x, se=FALSE, fullrange=TRUE) +
  stat_pp_line(aes(colour="col_1")) +
  stat_pp_point(aes(shape="y1_shape"), colour="blue", alpha=1, size=1.0,
                distribution=distr, dparams=distr_pars) +
  scale_x_continuous(name=x_name, breaks=x_breaks, label=x_labs, limits=x_lims) +
  scale_y_continuous(name=y_name, breaks=y_breaks, labels=NULL, limits=y_lims,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  scale_shape_manual(name="Legend", labels=leg_shape_labs, values=leg_shape_cols, breaks=leg_shape_sort)
  scale_fill_manual(name="", labels=leg_fill_labs, values=leg_fill_cols, breaks=leg_fill_sort) +
  scale_colour_manual(name="", labels=leg_col_labs, values=leg_col_cols, breaks=leg_col_sort) +
  guides(shape=guide_legend(order=1), fill=guide_legend(order=2), colour=guide_legend(order=3)) +
  theme(plot.title=element_text(hjust=0.5, size=13.5), plot.subtitle=element_text(hjust=0.5),
        axis.text.x=element_text(angle=0, vjust=1),
        legend.key.width=unit(0.8,"cm"), legend.position="bottom")
plot(Gauss_1_PP_boot_plot)
```

```
## Warning: The following aesthetics were dropped during statistical transformation: sample
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a `group` aesthetic or to convert a numerical
##   variable into a factor?

## Warning: Removed 1 rows containing missing values (`geom_smooth()`).
```



From the inspection of both the *Q-Q* and *P-P* plots for the *Gauss_1* data set we have rather strong visual evidence that the data set has been drawn from a Gaussian distribution. Note also that the proximity of the interquatile and regression line to the $y = x$ line corresponds to the circumstance that the empirical variance of the data sets *Gauss_1* is estimated to be close to 1, while the pattern of the scatter plot which is nor *S-shaped* neither *reverse S-shaped* corresponds to the circumstance that the empirical excess of kurtosis of the data sets *Gauss_1* is estimated to be close to 0.

We now consider the *Q-Q* and *P-P* plots for the *Gauss_2* data set.

The *Q-Q* plot.

```
z <- Gauss_df$Z_2                                          # Gauss_1 data set.
z_cent <- z - mean(z)                                # Centered Gauss_1 data set.
z_cent_qemp <- qemp(ppoints(length(z)), z_cent) # Empirical quantiles of the centered Gauss_1 data set.
Gauss_quants <- qnorm(ppoints(length(z)), mean=0, sd=1) # Quantiles of the standard Gaussian distributi
                                                           # corresponding to the empirical quantiles
Gausss_2_QQ_plot_df <- data.frame(k=1:length(z), S=z_cent, X=Gauss_quants, Y=z_cent_qemp)
head(Gausss_2_QQ_plot_df)
```

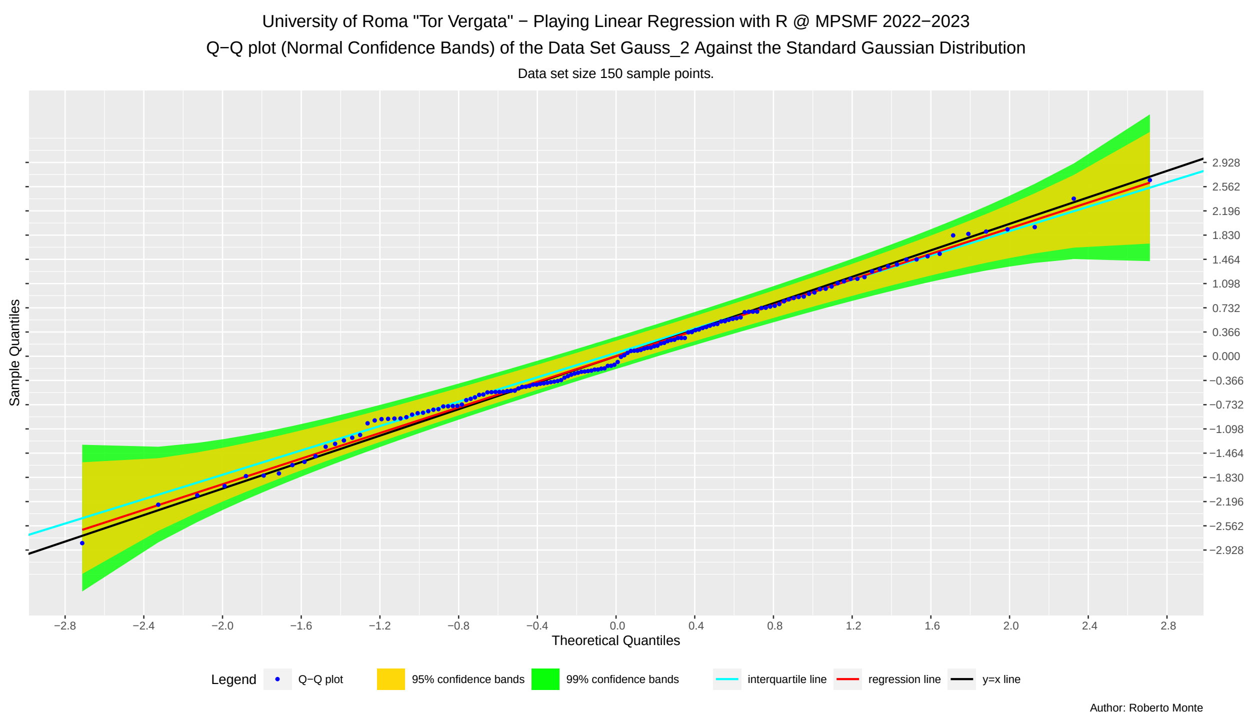
```
##   k          S         X         Y
## 1 1  1.1699106 -2.713052 -2.823428
## 2 2  0.4612693 -2.326348 -2.314228
## 3 3  0.2429946 -2.128045 -2.119162
## 4 4  0.3607637 -1.989313 -1.977785
## 5 5 -2.2431427 -1.880794 -1.828873
## 6 6  2.3804986 -1.790751 -1.803959
```

```
Data_df <- Gausss_2_QQ_plot_df
n <- nrow(Data_df)
quart_probs <- c(0.25,0.75)
quart_Y <- as.vector(quantile(Data_df$Y, quart_probs))
quart_X <- qnorm(quart_probs, mean=0, sd=1)
```

```
slope <- diff(quart_Y)/diff(quart_X)
intercept <- quart_Y[1]-slope*quart_X[1]
distr <- "norm"
distr_pars <- list(mean=0, sd=1)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u0
paste("Q-Q plot (Normal Confidence Bands) of the Data Set Gauss_2 Against the Standard Gaussian Distrib
subtitle_content <- bquote(paste("Data set size ", .(n), " sample points."))
caption_content <- "Author: Roberto Monte"
x_name <- bquote("Theoretical Quantiles")
y_name <- bquote("Sample Quantiles")
x_breaks_num <- 15 # (deduced from primeFactors(n))
# x_breaks_num <- ceiling(n^(1/2)) # Tukey & Mosteller square-root rule
# x_breaks_num <- ceiling(1+log2(n)) # Sturges rule
# x_breaks_num <- ceiling((2*n)^(1/3)) # Teller & Scott rice rule
x_binwidth <- round((max(Data_df$X)-min(Data_df$X))/x_breaks_num, digits=1)
x_breaks_low <- floor((min(Data_df$X)/x_binwidth))*x_binwidth
x_breaks_up <- ceiling((max(Data_df$X)/x_binwidth))*x_binwidth
x_breaks <- c(seq(from=x_breaks_low, to=x_breaks_up, by=x_binwidth))
x_labs <- format(x_breaks, scientific=FALSE)
J <- 1.0
x_lims <- c((x_breaks_low-J*x_binwidth), (x_breaks_up+J*x_binwidth))
y_breaks_num <- length(x_breaks)
y_binwidth <- round((max(Data_df$Y)-min(Data_df$Y))/y_breaks_num, digits=3)
y_breaks_low <- floor((min(Data_df$Y)/y_binwidth))*y_binwidth
y_breaks_up <- ceiling((max(Data_df$Y)/y_binwidth))*y_binwidth
y_breaks <- c(round(seq(from=y_breaks_low, to=y_breaks_up, by=y_binwidth),3))
y_labs <- format(y_breaks, scientific=FALSE)
K <- 1.5
y_lims <- c((y_breaks_low-K*y_binwidth), (y_breaks_up+K*y_binwidth))
y1_shape <- bquote("Q-Q plot")
y1_fill <- bquote("95% confidence bands")
y2_fill <- bquote("99% confidence bands")
col_1 <- bquote("interquartile line")
col_2 <- bquote("regression line")
col_3 <- bquote("y=x line")
leg_shape_labs <- y1_shape
leg_fill_labs <- c(y1_fill, y2_fill)
leg_col_labs <- c(col_1, col_2, col_3)
leg_shape_cols <- c("y1_shape" = 19)
leg_fill_cols <- c("y1_fill"="gold", "y2_fill"="green")
leg_col_cols <- c("col_1"="cyan", "col_2"="red", "col_3"="black")
leg_shape_sort <- "y1_shape"
leg_fill_sort <- c("y1_fill", "y2_fill")
leg_col_sort <- c("col_1", "col_2", "col_3")
Gauss_2_QQ_norm_plot <- ggplot(Data_df, aes(sample=S)) +
  stat_qq_band(aes(fill="y2_fill"), distribution=distr, dparams=distr_pars, conf = 0.99, bandType = "po
  stat_qq_band(aes(fill="y1_fill"), distribution=distr, dparams=distr_pars, conf = 0.95, bandType = "po
#  stat_qq_line(aes(colour="col_1"), distribution=distr, dparams=distr_pars) +
  geom_abline(aes(slope=slope, intercept=intercept, colour="col_1"), size=0.8, linetype="solid", show.l
#  geom_segment(aes(x=Q[1], xend=-Q[1], y=Q[1], yend=-Q[1], colour="col_3"),
#               size=0.8, linetype="solid", show.legend=FALSE) +
  geom_abline(aes(slope=1, intercept=0, colour="col_3"), size=0.8, linetype="solid", show.legend=FALSE)
  stat_smooth(alpha=1, size=0.8, linetype="solid", aes(x=X, y=Y, colour="col_2"),
```

```
                method="lm" , formula=y~x, se=FALSE, fullrange=FALSE) +
  stat_qq_point(aes(shape="y1_shape"), colour="blue", alpha=1, size=1.0,
                distribution=distr, dparams=distr_pars) +
  scale_x_continuous(name=x_name, breaks=x_breaks, label=x_labs, limits=NULL) +
  scale_y_continuous(name=y_name, breaks=y_breaks, labels=NULL, limits=NULL,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  scale_shape_manual(name="Legend", labels=leg_shape_labs, values=leg_shape_cols, breaks=leg_shape_sort)
  scale_fill_manual(name="", labels=leg_fill_labs, values=leg_fill_cols, breaks=leg_fill_sort) +
  scale_colour_manual(name="", labels=leg_col_labs, values=leg_col_cols, breaks=leg_col_sort) +
  guides(shape=guide_legend(order=1), fill=guide_legend(order=2), colour=guide_legend(order=3)) +
  theme(plot.title=element_text(hjust=0.5, size=13.5), plot.subtitle=element_text(hjust=0.5),
        axis.text.x=element_text(angle=0, vjust=1),
        legend.key.width=unit(0.8,"cm"), legend.position="bottom")
plot(Gauss_2_QQ_norm_plot)
```

```
## Warning: The following aesthetics were dropped during statistical transformation: sample
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a `group` aesthetic or to convert a numerical
##   variable into a factor?
```



The *P*–*P* plot.

```
z <- Gauss_df$Z_2                                    # Gauss_1 data set.
z_st <- (z-mean(z))/sd(z)               # Standardized Gauss_1 data set.
z_st_qemp <- qemp(ppoints(length(z_st)), z_st)# Empirical quantiles of the standardized Gauss_1 data se
z_st_pemp <- pemp(z_st_qemp, z_st)  # Empirical probabilities of the standardized Gauss_1 data set
                                        # corresponding to the empirical quantiles.
```

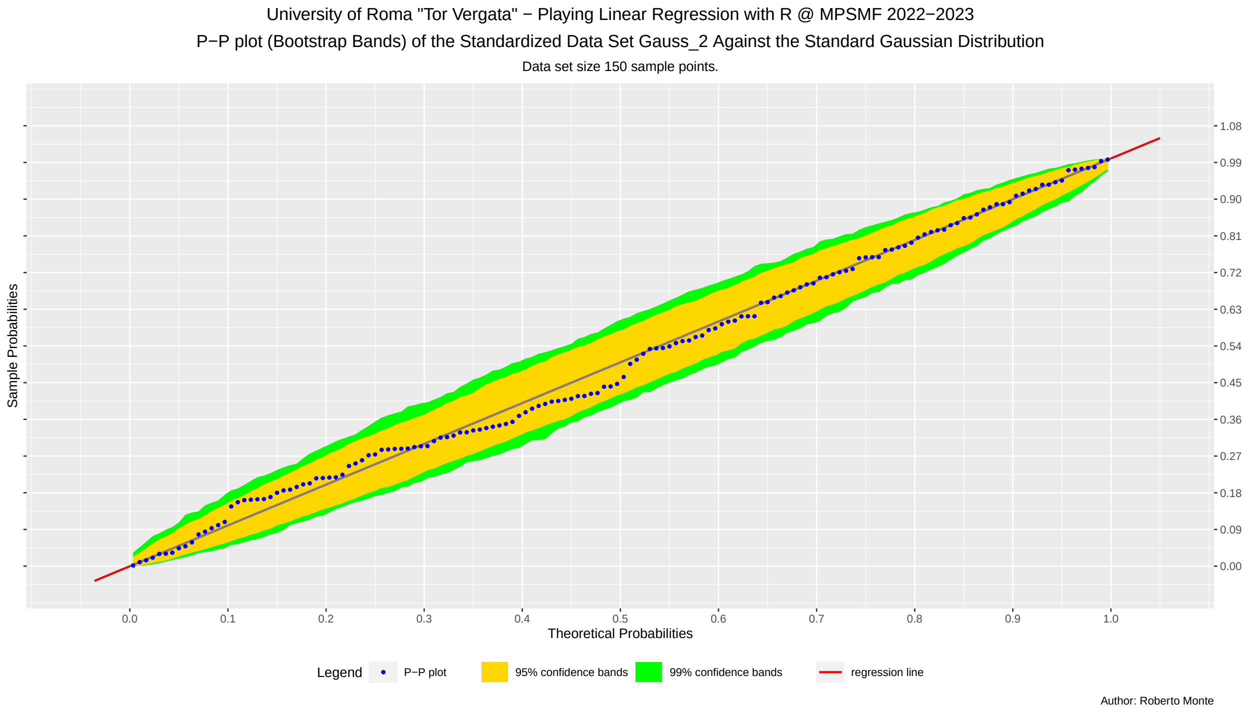
```
Gauss_quants <- qnorm(ppoints(length(z)), mean=0, sd=1) # Quantiles of the standard Gaussian distributi
                                                         # corresponding to the empirical quantiles.
Gauss_probs <- pnorm(Gauss_quants, mean=0, sd=1)# Probabilities of the standard Gaussian distribution
                                                         # corresponding to the empirical probabilities.
Gausss_2_PP_plot_df <- data.frame(k=1:length(z), S=z_st, X=Gauss_probs, Y=z_st_pemp)
head(Gausss_2_PP_plot_df)
```

```
##   k          S           X           Y
## 1 1  1.2147740 0.003333333 0.004159734
## 2 2  0.4789579 0.010000000 0.010000000
## 3 3  0.2523129 0.016666667 0.016666667
## 4 4  0.3745982 0.023333333 0.023333333
## 5 5 -2.3291621 0.030000000 0.030000000
## 6 6  2.4717853 0.036666667 0.036666667
```

```
Data_df <- Gausss_2_PP_plot_df
n <- nrow(Data_df)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u00
paste("P-P plot (Bootstrap Bands) of the Standardized Data Set Gauss_2 Against the Standard Gaussian Di
subtitle_content <- bquote(paste("Data set size ", .(n), " sample points."))
caption_content <- "Author: Roberto Monte"
x_name <- bquote("Theoretical Probabilities")
y_name <- bquote("Sample Probabilities")
x_breaks_num <- 15 # (deduced from primeFactors(n))
# x_breaks_num <- ceiling(n^(1/2)) # Tukey & Mosteller square-root rule
# x_breaks_num <- ceiling(1+log2(n)) # Sturges rule
# x_breaks_num <- ceiling((2*n)^(1/3)) # Teller & Scott rice rule
x_binwidth <- round((max(Data_df$X)-min(Data_df$X))/x_breaks_num, digits=1)
x_breaks_low <- floor((min(Data_df$X)/x_binwidth))*x_binwidth
x_breaks_up <- ceiling((max(Data_df$X)/x_binwidth))*x_binwidth
x_breaks <- c(seq(from=x_breaks_low, to=x_breaks_up, by=x_binwidth))
# x_breaks <- c(1,round(seq(from=x_breaks_low, to=x_breaks_up, by=x_binwidth),3),n)
x_labs <- format(x_breaks, scientific=FALSE)
J <- 0.5
x_lims <- c(x_breaks_low-J*x_binwidth, x_breaks_up+J*x_binwidth)
y_breaks_num <- length(x_breaks)
y_binwidth <- round((max(Data_df$Y)-min(Data_df$Y))/y_breaks_num, digits=3)
y_breaks_low <- floor((min(Data_df$Y)/y_binwidth))*y_binwidth
y_breaks_up <- ceiling((max(Data_df$Y)/y_binwidth))*y_binwidth
y_breaks <- c(round(seq(from=y_breaks_low, to=y_breaks_up, by=y_binwidth),3))
y_labs <- format(y_breaks, scientific=FALSE)
K <- 0.5
y_lims <- c(y_breaks_low-K*y_binwidth, y_breaks_up+K*y_binwidth)
y1_shape <- bquote("P-P plot")
y1_fill <- bquote("95% confidence bands")
y2_fill <- bquote("99% confidence bands")
col_1 <- bquote("y=x line")
col_2 <- bquote("regression line")
leg_shape_labs <- y1_shape
leg_fill_labs <- c(y1_fill, y2_fill)
leg_col_labs <- c(col_1, col_2)
leg_shape_cols <- c("y1_shape" = 19)
leg_fill_cols <- c("y1_fill"="gold", "y2_fill"="green")
```

```
leg_col_cols <- c("col_1"="black", "col_2"="red")
leg_shape_sort <- "y1_shape"
leg_fill_sort <- c("y1_fill", "y2_fill")
leg_col_sort <- c("col_1", "col_2")
distr <- "norm"
distr_pars <- list(mean=0, sd=1)
Gauss_2_PP_boot_plot <- ggplot(Data_df, aes(sample=S)) +
  stat_pp_band(aes(fill="y2_fill"), distribution=distr, dparams=distr_pars, conf = 0.99, bandType = "boo
  stat_pp_band(aes(fill="y1_fill"), distribution=distr, dparams=distr_pars, conf = 0.95, bandType = "boo
  stat_smooth(alpha=1, size=0.8, linetype="solid", aes(x=X, y=Y, colour="col_2"),
              method="lm" , formula=y~x, se=FALSE, fullrange=TRUE) +
  stat_pp_line(aes(colour="col_1")) +
  stat_pp_point(aes(shape="y1_shape"), colour="blue", alpha=1, size=1.0,
                distribution=distr, dparams=distr_pars) +
  scale_x_continuous(name=x_name, breaks=x_breaks, label=x_labs, limits=x_lims) +
  scale_y_continuous(name=y_name, breaks=y_breaks, labels=NULL, limits=y_lims,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  scale_shape_manual(name="Legend", labels=leg_shape_labs, values=leg_shape_cols, breaks=leg_shape_sort)
  scale_fill_manual(name="", labels=leg_fill_labs, values=leg_fill_cols, breaks=leg_fill_sort) +
  scale_colour_manual(name="", labels=leg_col_labs, values=leg_col_cols, breaks=leg_col_sort) +
  guides(shape=guide_legend(order=1), fill=guide_legend(order=2), colour=guide_legend(order=3)) +
  theme(plot.title=element_text(hjust=0.5, size=13.5), plot.subtitle=element_text(hjust=0.5),
        axis.text.x=element_text(angle=0, vjust=1),
        legend.key.width=unit(0.8,"cm"), legend.position="bottom")
plot(Gauss_2_PP_boot_plot)
```

```
## Warning: The following aesthetics were dropped during statistical transformation: sample
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a `group` aesthetic or to convert a numerical
##   variable into a factor?

## Warning: Removed 1 rows containing missing values (`geom_smooth()`).
```



The same considerations as in the case of the data set *Gauss_1* can be drawn from the inspection of the *Q-Q* and *P-P* plots for the *Gauss_2* data set.

As the last visual test to check the Gaussianity of the data set *Gauss_1* and *Gauss_2*, we plot the relative frequency and the density histograms.

First, we build a table to compare the value taken by standard statistics on the data sets with theoretical values.

Standard statistics on *Gauss_1* data set.

```
mode <- function(x) {
  d <- density(x)
  d$x[which.max(d$y)]
}
Samp_Data <- Gauss_df$Z_1
Statistics=c("mean", "median", "mode", "min. (99.73%)", "max. (99.73%)", "1st quart.", "3rd quart.", "s

Teor_Stats <- rep(0,10)
Teor_Stats[1] <- as.character(formatC(0, digits=3, format="f"))
Teor_Stats[2] <- as.character(formatC(qnorm(0.50, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE), dig
Teor_Stats[3] <- as.character(formatC(0, digits=3, format="f"))
Teor_Stats[4] <- as.character(formatC(qnorm(0.00135, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE),
Teor_Stats[5] <- as.character(formatC(qnorm(0.99865, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE),
Teor_Stats[6] <- as.character(formatC(qnorm(0.25, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE), dig
Teor_Stats[7] <- as.character(formatC(qnorm(0.75, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE), dig
Teor_Stats[8] <- as.character(formatC(1, digits=3, format="f"))
Teor_Stats[9] <- as.character(formatC(0, digits=3, format="f"))
Teor_Stats[10] <- as.character(formatC(0, digits=3, format="f"))

Samp_Stats <- rep(0,10)
Samp_Stats[1] <- as.character(formatC(mean(Samp_Data), digits=3, format="f"))
```

```
Samp_Stats[2] <- as.character(formatC(median(Samp_Data), digits=3, format="f"))
Samp_Stats[3] <- as.character(formatC(mode(Samp_Data), digits=3, format="f"))
Samp_Stats[4] <- as.character(formatC(min(Samp_Data), digits=3, format="f"))
Samp_Stats[5] <- as.character(formatC(max(Samp_Data), digits=3, format="f"))
Samp_Stats[6] <- as.character(formatC(quantile(Samp_Data,0.25), digits=3, format="f"))
Samp_Stats[7] <- as.character(formatC(quantile(Samp_Data,0.75), digits=3, format="f"))
Samp_Stats[8] <- as.character(formatC(sd(Samp_Data), digits=3, format="f"))
Samp_Stats[9] <- as.character(formatC(as.numeric(timeDate::skewness(Samp_Data, method="moment")), digit
Samp_Stats[10] <- as.character(formatC(as.numeric(timeDate::kurtosis(Samp_Data, method="excess")), digi

Table_Stats <- data.frame(Samp_Stats,Teor_Stats)
rownames(Table_Stats) <- Statistics
colnames(Table_Stats) <- c("Samp. Stats", "Teor. Stats")
```

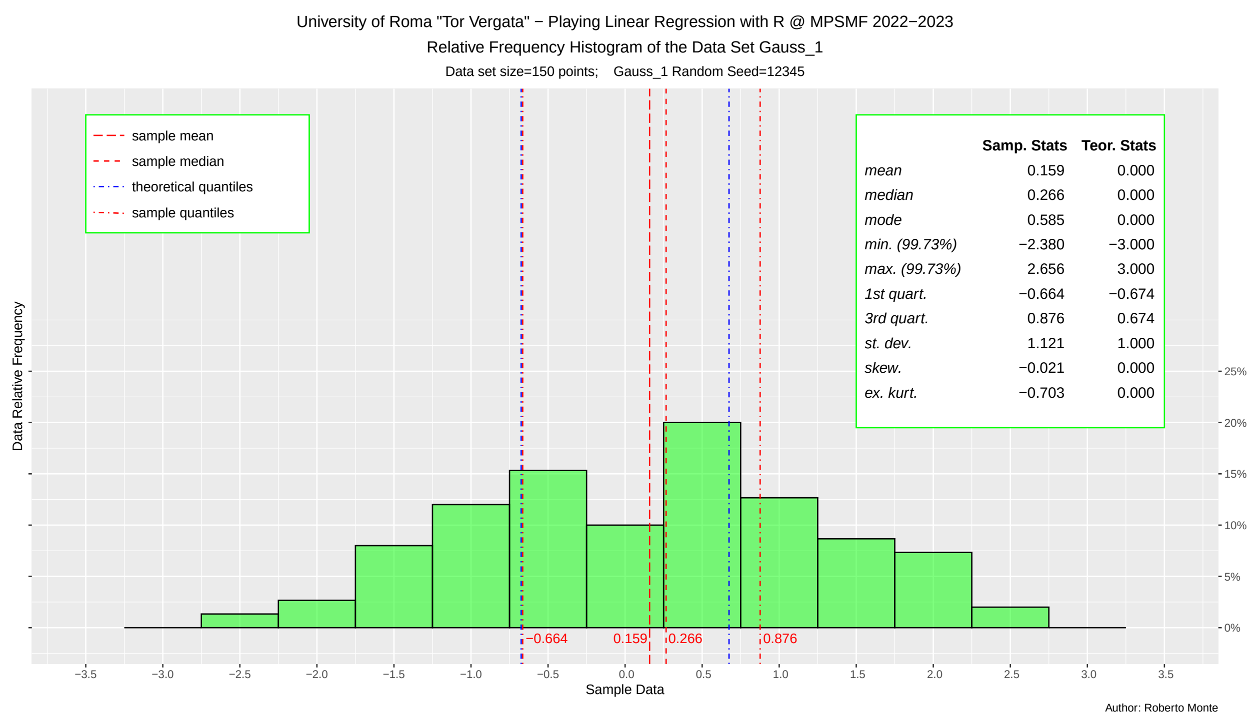
Then, we plot relative the frequency and density histograms of the data set *Gauss_1*.

The relative frequency histograms

```
# library(gridExtra)
#### Relative Frequency Histogram + Sample Statistics
Data_df <- Gauss_df
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u0
subtitle_content <- bquote(paste("Data set size=", .(n), " points;    Gauss_1 Random Seed=12345"))
caption_content <- "Author: Roberto Monte"
x_binwidth <- 0.5
x_breaks <- seq(from=-3.5, to=3.5, by=x_binwidth)
x_labs <- format(x_breaks, scientific=FALSE)
x_lims <- c(-3.5,3.5)
y_breaks <- seq(from=0, to=0.25, by=0.05)
y_labs <- format(percent(y_breaks), scientific=FALSE)
y_lims <- c(-0.010,0.50)
tt3 <- ttheme_minimal(core=list(fg_params=list(hjust=1, x=0.90)),
                      rowhead=list(fg_params=list(hjust=0, x=0)))
Table_Stats_Grob <- tableGrob(Table_Stats, theme=tt3)
Data_df_rel_freq_hist <- ggplot(Data_df, aes(x=Z_1)) +
  geom_histogram(binwidth=x_binwidth , aes(y=stat(count)/sum(count)), color="black", fill="green", alph
  scale_x_continuous(name="Sample Data", breaks=x_breaks, labels=x_labs, limits=x_lims) +
  scale_y_continuous(name="Data Relative Frequency", breaks=y_breaks, labels=NULL, limits=y_lims,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  theme(plot.title=element_text(hjust=0.5),
        plot.subtitle=element_text(hjust=0.5),
        plot.caption=element_text(hjust=1.0)) +
  geom_vline(aes(xintercept=as.numeric(quantile(Samp_Data,0.25))),
             colour="red", linetype="dotdash", size=0.5) +
  annotate("text", x=as.numeric(quantile(Samp_Data,0.25))+0.020, y=-0.01, colour="red",
           label=Samp_Stats[6], hjust=0) +
  geom_vline(aes(xintercept=as.numeric(quantile(Samp_Data,0.75))),
             colour="red", linetype="dotdash", size=0.5) +
  annotate("text", x=as.numeric(quantile(Samp_Data,0.75))+0.020, y=-0.01, colour="red",
           label=Samp_Stats[7], hjust=0) +
  geom_vline(aes(xintercept=mean(Samp_Data)), colour="red", linetype="longdash", size=0.5) +
  annotate("text", x=mean(Samp_Data)-0.235, y=-0.01, colour="red",
```

```
           label=Samp_Stats[1], hjust=0) +
  geom_vline(aes(xintercept=median(Samp_Data)), colour="red", linetype="dashed", size=0.5) +
  annotate("text", x=median(Samp_Data)+0.015, y=-0.01, colour="red",
           label=Samp_Stats[2], hjust=0) +
  geom_vline(aes(xintercept=qnorm(0.25, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE)),
             colour="blue", linetype="dotdash", size=0.5) +
  geom_vline(aes(xintercept=qnorm(0.75, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE)),
             colour="blue", linetype="dotdash", size=0.5) +
  annotate("rect", xmin=1.50, xmax=3.50, ymin=0.195, ymax=0.5, colour="green", fill="white") +
  annotation_custom(Table_Stats_Grob, xmin=1.75, xmax=3.30, ymin=0.3, ymax=0.4) +
  annotate("rect", xmin=-3.50, xmax=-2.05, ymin=0.385, ymax=0.500, colour="green", fill="white") +
  annotate("segment", x=-3.45, xend=-3.25, y=0.480, yend=0.480, colour="red", lty="longdash") +
  annotate("text", x=-3.20, y=0.480, colour="black", label="sample mean", hjust=0) +
  annotate("segment", x=-3.45, xend=-3.25, y=0.455, yend=0.455, colour="red", lty="dashed") +
  annotate("text", x=-3.20, y=0.455, colour="black", label="sample median", hjust=0) +
  annotate("segment", x=-3.45, xend=-3.25, y=0.430, yend=0.430, colour="blue", lty="dotdash") +
  annotate("text", x=-3.20, y=0.430, colour="black", label="theoretical quantiles", hjust=0) +
  annotate("segment", x=-3.45, xend=-3.25, y=0.405, yend=0.404, colour="red", lty="dotdash") +
  annotate("text", x=-3.20, y=0.405, colour="black", label="sample quantiles", hjust=0)
plot(Data_df_rel_freq_hist)
```

```
## Warning: `stat(count)` was deprecated in ggplot2 3.4.0.
## i Please use `after_stat(count)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```

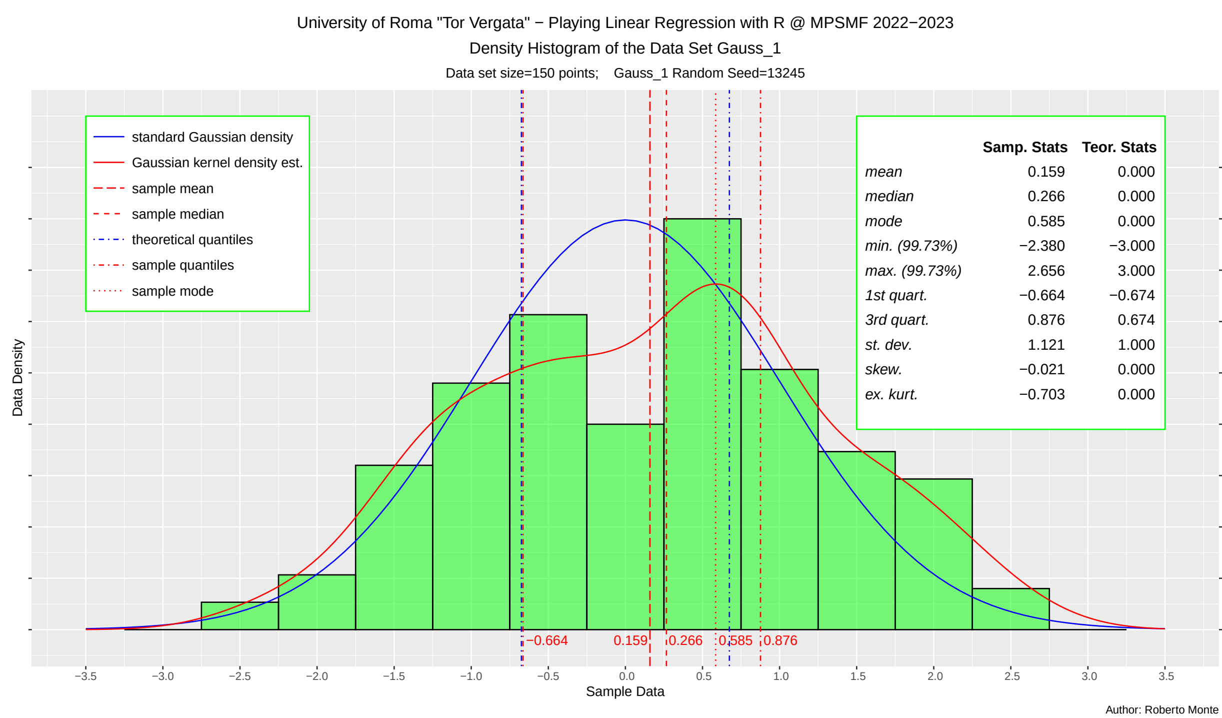


The density histograms

```r
# library(gridExtra)
#### Density Histogram + Sample Statistics + Density Kernel Estimation
Data_df <- Gauss_df
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u00
subtitle_content <- bquote(paste("Data set size=", .(n), " points;    Gauss_1 Random Seed=13245"))
caption_content <- "Author: Roberto Monte"
x_binwidth <- 0.5
x_breaks <- seq(from=-3.5, to=3.5, by=x_binwidth)
x_labs <- format(x_breaks, scientific=FALSE)
x_lims <- c(-3.5,3.5)
y_breaks <- seq(from=0, to=0.45, by=0.05)
y_labs <- format(y_breaks, scientific=FALSE)
y_lims <- c(-0.010,0.50)
tt3 <- ttheme_minimal(core=list(fg_params=list(hjust=1, x=0.90)),
                      rowhead=list(fg_params=list(hjust=0, x=0)))
Table_Stats_Grob <- tableGrob(Table_Stats, theme=tt3)
Data_df_dens_hist <- ggplot(Data_df, aes(x=Z_1)) +
  geom_histogram(binwidth=x_binwidth, aes(y=..density..), # binwidth=0.5,  # Density Histogram
                 color="black", fill="green", alpha=0.5) +
  scale_x_continuous(name="Sample Data", breaks=x_breaks, labels=x_labs, limits=x_lims) +
  scale_y_continuous(name="Data Density", breaks=y_breaks, labels=NULL, limits=y_lims,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  theme(plot.title=element_text(lineheight=0.6, face="bold", hjust=0.5),
        plot.subtitle=element_text(hjust= 0.5),
        plot.caption=element_text(hjust=1.0)) +
  stat_function(fun=dnorm, colour="blue", args=list(mean=0, sd=1)) +
  geom_vline(aes(xintercept=as.numeric(quantile(Samp_Data,0.25))),
             colour="red", linetype="dotdash", size=0.5) +
  annotate("text", x=as.numeric(quantile(Samp_Data,0.25))+0.020, y=-0.01, colour="red",
           label=Samp_Stats[6], hjust=0) +
  geom_vline(aes(xintercept=as.numeric(quantile(Samp_Data,0.75))),
             colour="red", linetype="dotdash", size=0.5) +
  annotate("text", x=as.numeric(quantile(Samp_Data,0.75))+0.020, y=-0.01, colour="red",
           label=Samp_Stats[7], hjust=0) +
  geom_vline(aes(xintercept=mean(Samp_Data)), colour="red", linetype="longdash", size=0.5) +
  annotate("text", x=mean(Samp_Data)-0.235, y=-0.01, colour="red",
           label=Samp_Stats[1], hjust=0) +
  geom_vline(aes(xintercept=median(Samp_Data)), colour="red", linetype="dashed", size=0.5) +
  annotate("text", x=median(Samp_Data)+0.015, y=-0.01, colour="red",
           label=Samp_Stats[2], hjust=0) +
  geom_vline(aes(xintercept=mode(Samp_Data)), colour="red", linetype="dotted", size=0.5) +
  annotate("text", x=mode(Samp_Data)+0.020, y=-0.01, colour="red",
           label=Samp_Stats[3], hjust=0) +
  geom_vline(aes(xintercept=qnorm(0.25, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE)),
             colour="blue", linetype="dotdash", size=0.5) +
  geom_vline(aes(xintercept=qnorm(0.75, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE)),
             colour="blue", linetype="dotdash", size=0.5) +
  geom_density(alpha=.2, colour="red") +
  annotate("rect", xmin=1.50, xmax=3.50, ymin=0.195, ymax=0.5, colour="green", fill="white") +
  annotation_custom(Table_Stats_Grob, xmin=1.75, xmax=3.30, ymin=0.3, ymax=0.4) +
  annotate("rect", xmin=-3.50, xmax=-2.05, ymin=0.310, ymax=0.500, colour="green", fill="white") +
```

```r
  annotate("segment", x= -3.45, xend=-3.25, y=0.480, yend=0.480, colour="blue") +
  annotate("text", x=-3.20, y=0.480, colour="black", label="standard Gaussian density", hjust=0) +
  annotate("segment", x= -3.45, xend=-3.25, y=0.455, yend=0.455, colour="red") +
  annotate("text", x=-3.20, y=0.455, colour="black", label="Gaussian kernel density est.", hjust=0) +
  annotate("segment", x= -3.45, xend=-3.25, y=0.430, yend=0.430, colour="red", lty="longdash") +
  annotate("text", x=-3.20, y=0.430, colour="black", label="sample mean", hjust=0) +
  annotate("segment", x= -3.45, xend=-3.25, y=0.405, yend=0.405, colour="red", lty="dashed") +
  annotate("text", x=-3.20, y=0.405, colour="black", label="sample median", hjust=0) +
  annotate("segment", x= -3.45, xend=-3.25, y=0.380, yend=0.380, colour="blue", lty="dotdash") +
  annotate("text", x=-3.20, y=0.380, colour="black", label="theoretical quantiles", hjust=0) +
  annotate("segment", x= -3.45, xend=-3.25, y=0.355, yend=0.355, colour="red", lty="dotdash") +
  annotate("text", x=-3.20, y=0.355, colour="black", label="sample quantiles", hjust=0) +
  annotate("segment", x= -3.45, xend=-3.25, y=0.330, yend=0.330, colour="red", lty="dotted") +
  annotate("text", x=-3.20, y=0.330, colour="black", label="sample mode", hjust=0)
plot(Data_df_dens_hist)
```

```
## Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.
## i Please use `after_stat(density)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```



We also plot the relative frequency and density histograms of the data set *Gauss_2*.

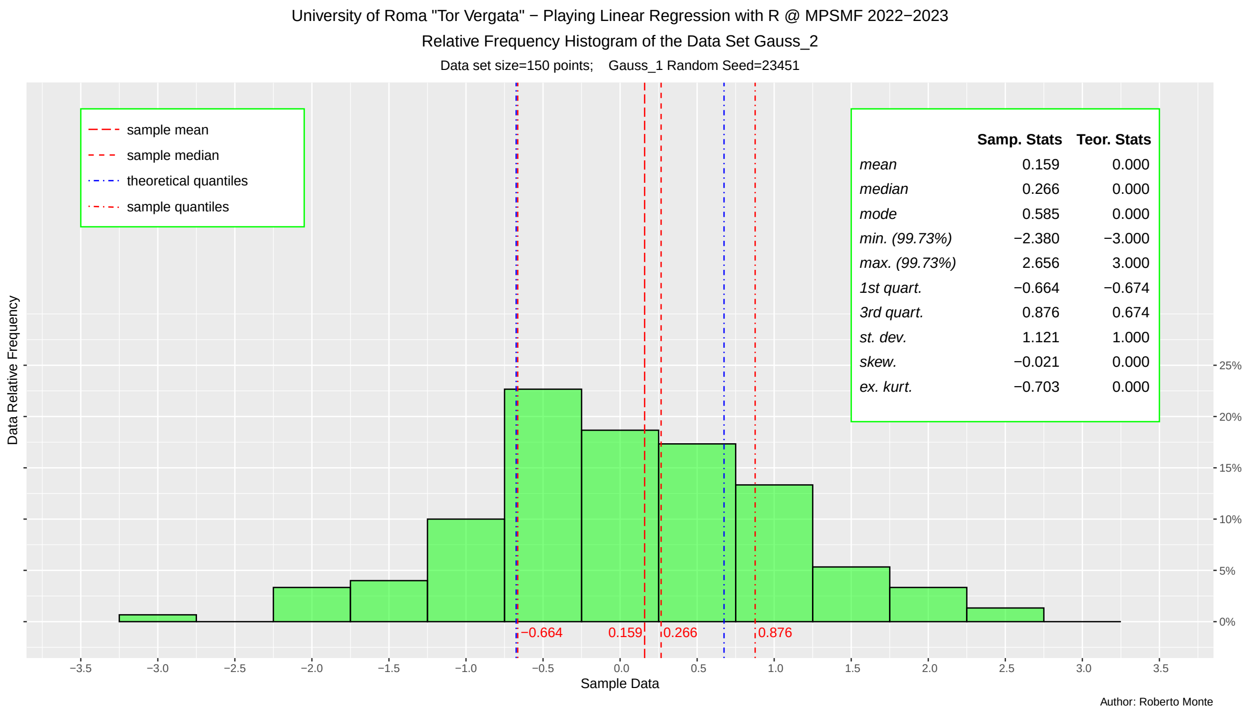
The relative frequency histograms

```r
# library(gridExtra)
#### Relative Frequency Histogram + Sample Statistics
```

```
Data_df <- Gauss_df
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u00
subtitle_content <- bquote(paste("Data set size=", .(n), " points;    Gauss_1 Random Seed=23451"))
caption_content <- "Author: Roberto Monte"
x_binwidth <- 0.5
x_breaks <- seq(from=-3.5, to=3.5, by=x_binwidth)
x_labs <- format(x_breaks, scientific=FALSE)
x_lims <- c(-3.5,3.5)
y_breaks <- seq(from=0, to=0.25, by=0.05)
y_labs <- format(percent(y_breaks), scientific=FALSE)
y_lims <- c(-0.010,0.50)
tt3 <- ttheme_minimal(core=list(fg_params=list(hjust=1, x=0.90)),
                      rowhead=list(fg_params=list(hjust=0, x=0)))
Table_Stats_Grob <- tableGrob(Table_Stats, theme=tt3)
Data_df_rel_freq_hist <- ggplot(Data_df, aes(x=Z_2)) +
  geom_histogram(binwidth=x_binwidth , aes(y=stat(count)/sum(count)), color="black", fill="green", alpha
  scale_x_continuous(name="Sample Data", breaks=x_breaks, labels=x_labs, limits=x_lims) +
  scale_y_continuous(name="Data Relative Frequency", breaks=y_breaks, labels=NULL, limits=y_lims,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  theme(plot.title=element_text(hjust=0.5),
        plot.subtitle=element_text(hjust=0.5),
        plot.caption=element_text(hjust=1.0)) +
  geom_vline(aes(xintercept=as.numeric(quantile(Samp_Data,0.25))),
             colour="red", linetype="dotdash", size=0.5) +
  annotate("text", x=as.numeric(quantile(Samp_Data,0.25))+0.020, y=-0.01, colour="red",
           label=Samp_Stats[6], hjust=0) +
  geom_vline(aes(xintercept=as.numeric(quantile(Samp_Data,0.75))),
             colour="red", linetype="dotdash", size=0.5) +
  annotate("text", x=as.numeric(quantile(Samp_Data,0.75))+0.020, y=-0.01, colour="red",
           label=Samp_Stats[7], hjust=0) +
  geom_vline(aes(xintercept=mean(Samp_Data)), colour="red", linetype="longdash", size=0.5) +
  annotate("text", x=mean(Samp_Data)-0.235, y=-0.01, colour="red",
           label=Samp_Stats[1], hjust=0) +
  geom_vline(aes(xintercept=median(Samp_Data)), colour="red", linetype="dashed", size=0.5) +
  annotate("text", x=median(Samp_Data)+0.015, y=-0.01, colour="red",
           label=Samp_Stats[2], hjust=0) +
  geom_vline(aes(xintercept=qnorm(0.25, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE)),
             colour="blue", linetype="dotdash", size=0.5) +
  geom_vline(aes(xintercept=qnorm(0.75, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE)),
             colour="blue", linetype="dotdash", size=0.5) +
  annotate("rect", xmin=1.50, xmax=3.50, ymin=0.195, ymax=0.5, colour="green", fill="white") +
  annotation_custom(Table_Stats_Grob, xmin=1.75, xmax=3.30, ymin=0.3, ymax=0.4) +
  annotate("rect", xmin=-3.50, xmax=-2.05, ymin=0.385, ymax=0.500, colour="green", fill="white") +
  annotate("segment", x=-3.45, xend=-3.25, y=0.480, yend=0.480, colour="red", lty="longdash") +
  annotate("text", x=-3.20, y=0.480, colour="black", label="sample mean", hjust=0) +
  annotate("segment", x=-3.45, xend=-3.25, y=0.455, yend=0.455, colour="red", lty="dashed") +
  annotate("text", x=-3.20, y=0.455, colour="black", label="sample median", hjust=0) +
  annotate("segment", x=-3.45, xend=-3.25, y=0.430, yend=0.430, colour="blue", lty="dotdash") +
  annotate("text", x=-3.20, y=0.430, colour="black", label="theoretical quantiles", hjust=0) +
  annotate("segment", x=-3.45, xend=-3.25, y=0.405, yend=0.404, colour="red", lty="dotdash") +
  annotate("text", x=-3.20, y=0.405, colour="black", label="sample quantiles", hjust=0)
```

```
plot(Data_df_rel_freq_hist)
```

```
## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```

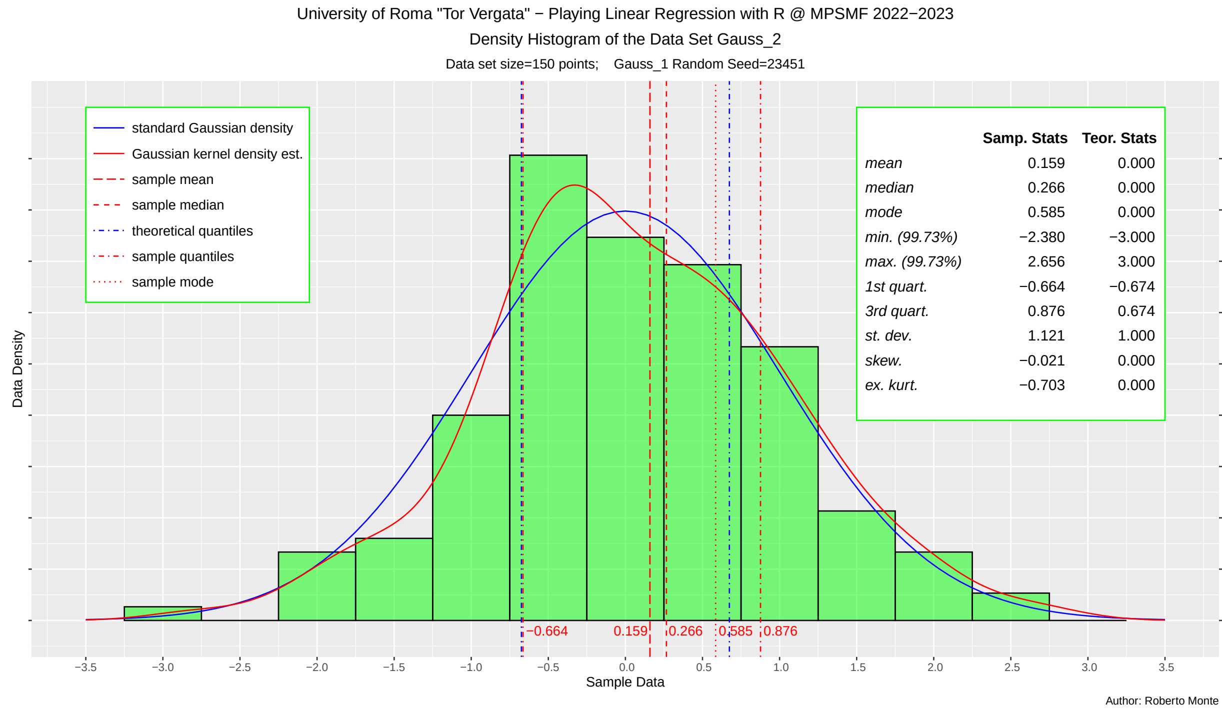


The density histograms

```
# library(gridExtra)
#### Density Histogram + Sample Statistics + Density Kernel Estimation
Data_df <- Gauss_df
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u00
subtitle_content <- bquote(paste("Data set size=", .(n), " points;    Gauss_1 Random Seed=23451"))
caption_content <- "Author: Roberto Monte"
x_binwidth <- 0.5
x_breaks <- seq(from=-3.5, to=3.5, by=x_binwidth)
x_labs <- format(x_breaks, scientific=FALSE)
x_lims <- c(-3.5,3.5)
y_breaks <- seq(from=0, to=0.45, by=0.05)
y_labs <- format(y_breaks, scientific=FALSE)
y_lims <- c(-0.010,0.50)
tt3 <- ttheme_minimal(core=list(fg_params=list(hjust=1, x=0.90)),
                      rowhead=list(fg_params=list(hjust=0, x=0)))
Table_Stats_Grob <- tableGrob(Table_Stats, theme=tt3)
Data_df_dens_hist <- ggplot(Data_df, aes(x=Z_2)) +
  geom_histogram(binwidth=x_binwidth, aes(y=..density..), # binwidth=0.5,  # Density Histogram
                 color="black", fill="green", alpha=0.5) +
  scale_x_continuous(name="Sample Data", breaks=x_breaks, labels=x_labs, limits=x_lims) +
  scale_y_continuous(name="Data Density", breaks=y_breaks, labels=NULL, limits=y_lims,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
```

```r
  theme(plot.title=element_text(lineheight=0.6, face="bold", hjust=0.5),
        plot.subtitle=element_text(hjust= 0.5),
        plot.caption=element_text(hjust=1.0)) +
  stat_function(fun=dnorm, colour="blue", args=list(mean=0, sd=1)) +
  geom_vline(aes(xintercept=as.numeric(quantile(Samp_Data,0.25))),
             colour="red", linetype="dotdash", size=0.5) +
  annotate("text", x=as.numeric(quantile(Samp_Data,0.25))+0.020, y=-0.01, colour="red",
           label=Samp_Stats[6], hjust=0) +
  geom_vline(aes(xintercept=as.numeric(quantile(Samp_Data,0.75))),
             colour="red", linetype="dotdash", size=0.5) +
  annotate("text", x=as.numeric(quantile(Samp_Data,0.75))+0.020, y=-0.01, colour="red",
           label=Samp_Stats[7], hjust=0) +
  geom_vline(aes(xintercept=mean(Samp_Data)), colour="red", linetype="longdash", size=0.5) +
  annotate("text", x=mean(Samp_Data)-0.235, y=-0.01, colour="red",
           label=Samp_Stats[1], hjust=0) +
  geom_vline(aes(xintercept=median(Samp_Data)), colour="red", linetype="dashed", size=0.5) +
  annotate("text", x=median(Samp_Data)+0.015, y=-0.01, colour="red",
           label=Samp_Stats[2], hjust=0) +
  geom_vline(aes(xintercept=mode(Samp_Data)), colour="red", linetype="dotted", size=0.5) +
  annotate("text", x=mode(Samp_Data)+0.020, y=-0.01, colour="red",
           label=Samp_Stats[3], hjust=0) +
  geom_vline(aes(xintercept=qnorm(0.25, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE)),
             colour="blue", linetype="dotdash", size=0.5) +
  geom_vline(aes(xintercept=qnorm(0.75, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE)),
             colour="blue", linetype="dotdash", size=0.5) +
  geom_density(alpha=.2, colour="red") +
  annotate("rect", xmin=1.50, xmax=3.50, ymin=0.195, ymax=0.5, colour="green", fill="white") +
  annotation_custom(Table_Stats_Grob, xmin=1.75, xmax=3.30, ymin=0.3, ymax=0.4) +
  annotate("rect", xmin=-3.50, xmax=-2.05, ymin=0.310, ymax=0.500, colour="green", fill="white") +
  annotate("segment", x= -3.45, xend=-3.25, y=0.480, yend=0.480, colour="blue") +
  annotate("text", x=-3.20, y=0.480, colour="black", label="standard Gaussian density", hjust=0) +
  annotate("segment", x= -3.45, xend=-3.25, y=0.455, yend=0.455, colour="red") +
  annotate("text", x=-3.20, y=0.455, colour="black", label="Gaussian kernel density est.", hjust=0) +
  annotate("segment", x= -3.45, xend=-3.25, y=0.430, yend=0.430, colour="red", lty="longdash") +
  annotate("text", x=-3.20, y=0.430, colour="black", label="sample mean", hjust=0) +
  annotate("segment", x= -3.45, xend=-3.25, y=0.405, yend=0.405, colour="red", lty="dashed") +
  annotate("text", x=-3.20, y=0.405, colour="black", label="sample median", hjust=0) +
  annotate("segment", x= -3.45, xend=-3.25, y=0.380, yend=0.380, colour="blue", lty="dotdash") +
  annotate("text", x=-3.20, y=0.380, colour="black", label="theoretical quantiles", hjust=0) +
  annotate("segment", x= -3.45, xend=-3.25, y=0.355, yend=0.355, colour="red", lty="dotdash") +
  annotate("text", x=-3.20, y=0.355, colour="black", label="sample quantiles", hjust=0) +
  annotate("segment", x= -3.45, xend=-3.25, y=0.330, yend=0.330, colour="red", lty="dotted") +
  annotate("text", x=-3.20, y=0.330, colour="black", label="sample mode", hjust=0)
plot(Data_df_dens_hist)
```

```
## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```



On the computational side, we will apply four normality tests to the data sets *Gauss_1* and *Gauss_2*: the Shapiro-Wilks (SW), D'agostino-Pearson (DP), Anderson-Darling (AD), and Jarque-Bera (JB) test. It should be noted that the above normality tests (and others) rely on the assumption that the data sets have been generated by means of independent random sampling from some distribution. Therefore, to apply them we implicitly need the support of the previous results.

The SW test

```
# Shapiro-Wilks (*SW*) test.
# library(stats)
z <- Gauss_df$Z_1
Gauss_1_SW <- shapiro.test(z)
show(Gauss_1_SW)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  z
## W = 0.98765, p-value = 0.2054
```

```
z <- Gauss_df$Z_2
Gauss_2_SW <- shapiro.test(z)
show(Gauss_2_SW)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  z
## W = 0.99511, p-value = 0.8994
```

By applying the *SW* test we cannot reject the null hypothesis of Gaussianity for both the data sets *Gauss_1* and *Gauss_2*.

The *DP* test

```
# D'Agostino-Pearson (*DP*) test.
# library(fBasics)
z <- Gauss_df$Z_1
Gauss_1_DP <- dagoTest(z)
show(Gauss_1_DP)
```

```
##
## Title:
##  D'Agostino Normality Test
##
## Test Results:
##   STATISTIC:
##     Chi2 | Omnibus: 5.3526
##     Z3  | Skewness: -0.113
##     Z4  | Kurtosis: -2.3108
##   P VALUE:
##     Omnibus  Test: 0.06882
##     Skewness Test: 0.9101
##     Kurtosis Test: 0.02084
```

```
z <- Gauss_df$Z_2
Gauss_2_DP <- dagoTest(z)
show(Gauss_2_DP)
```

```
##
## Title:
##  D'Agostino Normality Test
##
## Test Results:
##   STATISTIC:
##     Chi2 | Omnibus: 0.368
##     Z3  | Skewness: 0.17
##     Z4  | Kurtosis: 0.5823
##   P VALUE:
##     Omnibus  Test: 0.8319
##     Skewness Test: 0.865
##     Kurtosis Test: 0.5603
```

By applying the *DP* test we cannot reject the null hypothesis of Gaussianity for both the data sets *Gauss_1* and *Gauss_2*.

The *AD* test

```
# Anderson-Darling (*AD*) test.
# library(nortest)
z <- Gauss_df$Z_1
Gauss_1_AD <- ad.test(z)
show(Gauss_1_AD)
```

```
##
##  Anderson-Darling normality test
##
## data:  z
## A = 0.51181, p-value = 0.1923
```

```r
z <- Gauss_df$Z_2
Gauss_2_AD <- ad.test(z)
show(Gauss_2_AD)
```

```
##
##  Anderson-Darling normality test
##
## data:  z
## A = 0.32361, p-value = 0.5225
```

By applying the *AD* test we cannot reject the null hypothesis of Gaussianity for both the data sets *Gauss_1* and *Gauss_2*.

The *JB* test

```r
# Jarque-Bera (*JB*) test.
# library(tseries)
z <- Gauss_df$Z_1
Gauss_1_JB <- jarque.bera.test(z)
show(Gauss_1_JB)
```

```
##
##  Jarque Bera Test
##
## data:  z
## X-squared = 2.8379, df = 2, p-value = 0.242
```

```r
z <- Gauss_df$Z_2
Gauss_2_JB <- jarque.bera.test(z)
show(Gauss_2_JB)
```

```
##
##  Jarque Bera Test
##
## data:  z
## X-squared = 0.1222, df = 2, p-value = 0.9407
```

By applying the *JB* test we cannot reject the null hypothesis of Gaussianity for both the data sets *Gauss_1* and *Gauss_2*.

So far, we have collected a highly significant evidence that the data sets *Gauss_1* and *Gauss_2* have been generated by independent random sampling from a Gaussian distributions. However, note that the computational tests perform significantly better in case of the *Gauss_2* data set. This should be interpreted in light of the significant difference between the graph of the empirical density function of the *Gauss_1* data set and the graph of the standard gaussian density.

Since we cannot reject the null hypothesis of independent sampling from a Gaussian distribution for the generation of both the data sets *Gauss_1* and *Gauss_2*, in tackling the problem of determining confidence intervals for the *location* (mean) and *scale* (standard deviation) parameters of the data sets and performing hypothesis tests for the true values of these parameters we have to assume that *Gauss_1* and *Gauss_2* have actually been generated by independent sampling from a Gaussian distributions.

We recall that, for any $\alpha \in (0,1)$, the realization of the $(1-\alpha)\%$ confidence interval for the mean of the data set *Gauss_1* and *Gauss_2* is given by

$$\left(\bar{x}_n - t_{\frac{\alpha}{2},n-1}\frac{s_{X,n}}{\sqrt{n}}, \bar{x}_n + t_{\frac{\alpha}{2},n-1}\frac{s_{X,n}}{\sqrt{n}}\right), \tag{127}$$

where where $\bar{x}_n$ [resp. $s_{X,n}$] is the realization of the sample mean [resp. unbiased standard deviation], given by

$$\bar{x}_n \equiv \frac{1}{n}\sum_{k=1}^{n} x_k, \qquad s_{X,n} \equiv \sqrt{\frac{1}{n-1}\sum_{k=1}^{n}(x_k - \bar{x}_n)^2} \tag{128}$$

$t_{\frac{\alpha}{2},n-1}$ is the upper tail critical value of level $\alpha/2$ of the Student t-distribution with $n-1$ degrees of freedom, and $n \equiv 150$.

With reference to the data set *Gauss_1* [resp. *Gauss_2*], rounding to the 6th decimal place, we have

$$\bar{x}_n \approx 0.159130, \quad s_n(x) \approx 1.12108 \qquad [\text{resp.}\bar{x}_n \approx 0.050341, \quad s_n(x) \approx 0.963069]. \tag{129}$$

Furthermore, choosing $\alpha = 0.05$, we have

$$t_{\frac{\alpha}{2},n-1} \equiv t_{0,025,149} \approx 1.976013 \tag{130}$$

In fact,

```
show(c(round(mean(Gauss_1), digits=6), round(sd(Gauss_1), digits=6)))
```

```
## [1] 0.159130 1.121082
```

```
show(c(round(mean(Gauss_2), digits=6), round(sd(Gauss_2), digits=6)))
```

```
## [1] 0.050341 0.963069
```

```
show(round(qt(p=0.025, df=149, lower.tail=FALSE, log.p=FALSE), digits=6))
```

```
## [1] 1.976013
```

As a consequence, for the data set *Gauss_1* we obtain

$$\bar{x}_n - t_{\frac{\alpha}{2},n-1}\frac{s_n(x)}{\sqrt{n}} = 0.159130 - 1.976013 * \frac{1.121082}{\sqrt{150}} \approx -0.021746 \tag{131}$$

and

$$\bar{x}_n + t_{\frac{\alpha}{2},n-1}\frac{s_n(x)}{\sqrt{n}} = 0.159130 + 1.976013 * \frac{1.12108}{\sqrt{150}} \approx 0.340006 \tag{132}$$

In turn, for the data set *Gauss_2* we have

$$\bar{x}_n - t_{\frac{\alpha}{2},n-1}\frac{s_n(x)}{\sqrt{n}} = 0.050341 - 1.976013 * \frac{0.963069}{\sqrt{150}} \approx -0.105041 \tag{133}$$

and

$$\bar{x}_n + t_{\frac{\alpha}{2},n-1}\frac{s_n(x)}{\sqrt{n}} = 0.050341 + 1.976013 * \frac{0.963069}{\sqrt{150}} \approx 0.205723 \tag{134}$$

Therefore, with reference to the *Gauss_1* [resp. *Gauss_2*] data set, the realization of the 95% confidence interval for the mean is given by

$$(-0.021746,\ 0.340006) \qquad [\text{resp.}\,(-0.105041,\ 0.205723)\,] \tag{135}$$

Note that, by applying the R command *t.test(x, mu=..., conf.level=...)*, we can compute directly the above intervals. In fact,

```
Gauss_1_095_t_test <- t.test(Gauss_1, mu=mean(Gauss_1), conf.level=0.95)
Gauss_2_095_t_test <- t.test(Gauss_2, mu=mean(Gauss_2), conf.level=0.95)
show(c(round(Gauss_1_095_t_test$conf.int, digits=6),round(Gauss_2_095_t_test$conf.int, digits=6)))
```

```
## [1] -0.021746  0.340006 -0.105041  0.205723
```

Note also that choosing the confidence level $\alpha = 90\%$ that is $\alpha = 0.9$ the confidence interval for the mean of the data sets *Gauss_1* and *Gauss_2* are given respectively by

```
Gauss_1_090_t_test <- t.test(Gauss_1, mu=mean(Gauss_1), conf.level=0.90)
Gauss_2_090_t_test <- t.test(Gauss_2, mu=mean(Gauss_2), conf.level=0.90)
show(c(round(Gauss_1_090_t_test$conf.int, digits=6),round(Gauss_2_090_t_test$conf.int, digits=6)))
```

```
## [1]  0.007625  0.310635 -0.079810  0.180492
```

Therefore, with the goal of including the value zero in the confidence interval, the confidence level cannot be improved for the data set *Gauss_1*, but can be improved for the data set *Gauss_2*.

In the end note that, since the size $n = 150$ of the data sets can be tought as "large", we could consider the approximate realization of the confidence interval for the mean given by

$$\left(\bar{x}_n - z_{\frac{\alpha}{2}}\frac{s_{X,n}}{\sqrt{n}},\, \bar{x}_n + z_{\frac{\alpha}{2}}\frac{s_{X,n}}{\sqrt{n}}\right), \tag{136}$$

where $z_{\frac{\alpha}{2}}$ is the upper tail critical value of level $\alpha/2$ of the Gauss distribution. Rounding to the 6th decimal place, we have

$$z_{\frac{\alpha}{2}} = 1.959964 \tag{137}$$

Therefore, the approximate realization of the confidence interval for the mean of the data set Gauss_1 [resp. Gauss_2] is given by

$$(-0.020277,\ 0.338537) \quad [\text{resp.}\,(-0.103779,\ 0.204461)\,]. \tag{138}$$

Also in this case, by applying the command *z.test(x, mu=..., sigma.x=..., conf.level=...)*, we can compute directly the above intervals. In fact, we obtain

```
# library(BSDA)
Gauss_1_095_z_test <- z.test(Gauss_1, mu=mean(Gauss_1), sigma.x=sd(Gauss_1), conf.level=0.95)
Gauss_2_095_z_test <- z.test(Gauss_2, mu=mean(Gauss_2), sigma.x=sd(Gauss_2), conf.level=0.95)
show(c(round(Gauss_1_095_z_test$conf.int, digits=6),round(Gauss_2_095_z_test$conf.int, digits=6)))
```

```
## [1] -0.020277  0.338537 -0.103779  0.204461
```

In light of the confidence intervals that we have obtained for the mean of the data sets *Gauss_1* and *Gauss_2*, we can make the hypothesis that the generating distributions of both data sets have mean 0, at the 95% confidence tevel.

Writing X for the random variables whose distributions generate the data sets and setting $\mu \equiv \mathbf{E}[X]$, we know that the statistic

$$T_{n-1} \equiv \frac{\bar{X}_n - \mu}{S_{X,n}/\sqrt{n}}, \tag{139}$$

where $\bar{X}_n$ [resp. $S_{X,n}$] is the sample mean [resp. unbiased standard deviation], given by

$$\bar{X}_n \stackrel{\text{def}}{=} \frac{1}{n}\sum_{k=1}^{n} X_k \qquad [\text{resp.} S_{X,n} \stackrel{\text{def}}{=} \sqrt{\frac{1}{n}\sum_{k=1}^{n}\left(X_k - \bar{X}_n\right)^2}], \tag{140}$$

$X_1, \dots, X_n$ being a simple random sample of size n drawn from X, has the Student's t-distribution with $n-1$ degrees of freedom. Therefore, we can check the rejection of our hypothesis $H_0 : \mu = 0$ against the alternative $H_1 : \mu \neq 0$, at any significance level $\alpha \in (0, 1)$, by checking whether the realization of the test statistic (139) falls within the rejection region, that is

$$T_{n-1}(\omega) \in \left(-\infty, -t_{\alpha/2,n-1}\right) \cup \left(t_{\alpha/2,n-1}, \infty\right), \tag{141}$$

or, in terms of the p-value corresponding to the realization of the test statistic, we obtain equivalently

$$\mathbf{P}\left(T_{n-1} \leq -\left|T_{n-1}(\omega)\right|\right) + \mathbf{P}\left(T_{n-1} \geq \left|T_{n-1}(\omega)\right|\right) = 2\mathbf{P}\left(T_{n-1} \geq \left|T_{n-1}(\omega)\right|\right) \leq \alpha. \tag{142}$$

Approximating to the 6th decimal place, With reference to the data set *Gauss_1* [resp. *Gauss_2*], the null hypothesis $H_0 : \mu = 0$, and the significance level $\alpha = 0.05$, that is $\alpha = 5\%$, we obtain

$$T_{n-1}(\omega) \equiv \frac{\bar{x}_n - \mu}{s_{X,n}/\sqrt{n}} = \frac{0.159130}{1.121082/\sqrt{150}} = 1.738442 \quad [\text{resp.} T_{n-1}(\omega) \equiv \frac{\bar{x}_n - \mu}{s_{X,n}/\sqrt{n}} = \frac{0.050341}{0.963069/\sqrt{150}} = 0.640192]. \tag{143}$$

Therefore, since in both cases we have

$$0 < T_{n-1}(\omega) < t_{\alpha/2,n-1} \equiv 1.976013, \tag{144}$$

we cannot reject the null hypothesis $H_0 : \mu = 0$ at the 0.05 significance level.

In terms of p-value, with reference to the data sets *Gauss_1* and *Gauss_2* we have

$$2\mathbf{P}\left(T_{n-1} \geq \left|T_{n-1}(\omega)\right|\right) = 2\mathbf{P}\left(T_{n-1} \geq 1.738442\right) = 0.084198 > 0.05 \tag{145}$$

and

$$2\mathbf{P}\left(T_{n-1} \geq \left|T_{n-1}(\omega)\right|\right) = 2\mathbf{P}\left(T_{n-1} \geq 0.640192\right) = 0.523032 > 0.05, \tag{146}$$

respectively. These confirm that we cannot reject the null hypothesis $H_0 : \mu = 0$ at the 0.05 significance level for noth the data sets.

Note that, by applying again the R command *t.test(x, mu=. . . ., conf.level=0.95)*, we can perform directly the hypothesis test under the null hypothesis $H_0 : \mu = mu$ at the significance level $\alpha = 1 - conf.level$. In fact, we have

```
Gauss_1_005_t_test <- t.test(Gauss_1, mu=0, conf.level=0.95)
Gauss_2_005_t_test <- t.test(Gauss_2, mu=0, conf.level=0.95)
show(c(round(Gauss_1_005_t_test$statistic, digits=6), round(Gauss_1_005_t_test$p.value, digits=6)))
```

```
##        t
## 1.738441 0.084199
```

```
show(c(round(Gauss_2_005_t_test$statistic, digits=6), round(Gauss_2_005_t_test$p.value, digits=6)))
```

```
##        t
## 0.640193 0.523030
```

Also in this case, thanks to the "large" size of the data sets, by applying the command *z.test(x, alternative=. . . , mu=. . . , sigma.x=. . . , conf.level=0.95)*, we can obtain approximate test of the null hypothesis $H_0 : \mu = mu$ at the significance level $\alpha = 1 - conf.level$. In fact,

```
Gauss_1_005_z_test <- z.test(Gauss_1, mu=0, sigma.x=sd(Gauss_1), conf.level=0.95)
Gauss_2_005_z_test <- z.test(Gauss_2, mu=0, sigma.x=sd(Gauss_2), conf.level=0.95)
show(c(round(Gauss_1_005_z_test$statistic, digits=6), round(Gauss_1_005_z_test$p.value, digits=6)))
```

```
##        z
## 1.738441 0.082133
```

```
show(c(round(Gauss_2_005_z_test$statistic, digits=6), round(Gauss_2_005_z_test$p.value, digits=6)))
```

```
##        z
## 0.640193 0.522047
```

Still because we have to behave as Gauss_1 and Gauss_2 have been generated by independent sampling from a Gaussian distributions, for any $\alpha \in (0, 1)$, the realization of the $(1 - \alpha)\%$ confidence interval for the variance of the data set *Gauss_1* and *Gauss_2* is given by

$$\left(\frac{(n-1)\, s_{X,n}^2}{\chi^2_{n-1,\alpha/2,+}}, \frac{(n-1)\, s_{X,n}^2}{\chi^2_{n-1,\alpha/2,-}}\right) \tag{147}$$

where $s_{X,n}^2$ is the realization of the unbiased sample variance, given by

$$s_{X,n}^2 \equiv \frac{1}{n-1} \sum_{k=1}^{n} (x_k - \bar{x}_n)^2 \,, \tag{148}$$

$\chi^2_{n-1,\alpha/2,-}$ [resp. $\chi^2_{n-1,\alpha/2,+}$] is the lower [resp. upper] tail critical value of level $\alpha/2$ of the chi-square distribution χ^2_{n-1}, with $n - 1$ degrees of freedom, and $n = 150$.

With reference to the data set *Gauss_1* [resp. *Gauss_2*], rounding to the 6th decimal place, we have

$$s_{X,n}^2 \approx 1.256825 \qquad [\text{resp.} s_{X,n}^2 \approx 0.927501]. \tag{149}$$

Furthermore, choosing $\alpha = 0.05$, we have

$$\chi^2_{n-1,\alpha/2,-} \equiv \chi^2_{149,0.025,-} \approx 117.098 \qquad \text{and} \qquad \chi^2_{n-1,\alpha/2,+} \equiv \chi^2_{149,0.025,+} \approx 184.687 \tag{150}$$

In fact,

```
show(c(round(qchisq(p=0.025, df=149, lower.tail=TRUE), digits=6), round(qchisq(p=0.025, df=149, lower.ta
```

```
## [1] 117.098 184.687
```

As a consequence, for the data set *Gauss_1* we have

$$\frac{(n-1)\, s_{X,n}^2}{\chi^2_{n-1,\alpha/2,+}} = \frac{149 * 1.256825}{184.687} \approx 1.013969 \qquad \text{and} \qquad \frac{(n-1)\, s_{X,n}^2}{\chi^2_{n-1,\alpha/2,-}} = \frac{149 * 1.256825}{117.098} \approx 1.599232 \tag{151}$$

In turn, for the data set *Gauss_2* we have

$$\frac{(n-1)\,s_{X,n}^2}{\chi^2_{n-1,\alpha/2,+}} = \frac{149 * 0.927501}{184.687} \approx 0.74828 \qquad \text{and} \qquad \frac{(n-1)\,s_{X,n}^2}{\chi^2_{n-1,\alpha/2,-}} = \frac{149 * 0.927501}{117.098} \approx 1.180188 \tag{152}$$

Therefore, with reference to the *Gauss_1* [resp. *Gauss_2*] data set, the realization of the 95% confidence interval for the variance is given by

$$(1.013969,\, 1.599232) \qquad [\text{resp.}\,(0.74828,\, 1.180188)\,] \tag{153}$$

Note that, by applying the R command *varTest(x, alternative=... sigma.squared=..., conf.level=...)*, we can compute directly the above intervals. In fact,

```
# library(EnvStats)
Gauss_1_chisq_test <- varTest(Gauss_1, alternative="two.sided", sigma.squared=var(Gauss_1), conf.level=
Gauss_2_chisq_test <- varTest(Gauss_2, alternative="two.sided", sigma.squared=var(Gauss_2), conf.level=
show(c(round(Gauss_1_chisq_test$conf.int, digits=6),round(Gauss_2_chisq_test$conf.int, digits=6)))
```

```
##      LCL      UCL      LCL      UCL
## 1.013969 1.599233 0.748280 1.180188
```

Note that the 95% confidence interval of the *Gauss_1* data set does not contain the point 1. On the contrary the 95% confidence interval of the *Gauss_2* data set contains the point 1. On the other hand, the 99% confidence interval of the *Gauss_1* data set, and clearly also the *Gauss_2* data set, contains the point 1. In fact,

```
# library(EnvStats)
Gauss_1_chisq_test <- varTest(Gauss_1, alternative="two.sided", sigma.squared=var(Gauss_1), conf.level=
show(round(Gauss_1_chisq_test$conf.int, digits=6))
```

```
##      LCL      UCL
## 0.949576 1.729290
## attr(,"conf.level")
## [1] 0.99
```

In light of the confidence intervals that we have obtained for the data sets *Gauss_1* and *Gauss_2*, we can make the hypothesis that both the data sets have variance 1. This hypothesis his more appropriate for tha data set *Gauss_2* than *Gauss_1*, though. Hence, writing X for the random variables whose distributions generate the data sets and setting $\sigma \equiv \mathbf{D^2}\left[X\right]$, we know that the statistic

$$\chi^2_{n-1} \equiv \frac{(n-1)\,S_{X,n}^2}{\sigma^2}, \tag{154}$$

where $S_{X,n}^2$ is the sample variance, given by

$$S_{X,n}^2 \stackrel{\text{def}}{=} \frac{1}{n-1}\sum_{k=1}^{n}\left(X_k - \bar{X}_n\right)^2, \tag{155}$$

$X_1, \ldots, X_n$ being a simple random sample of size n drawn from X, has the chi-square distribution with $n-1$ degrees of freedom. Therefore, we can check the rejection of our hypothesis $H_0 : \sigma^2 = 1$ against the alternative $H_1 : \sigma^2 = 1 \neq 0$, at any significance level $\alpha \in (0,1)$, by checking whether the realization of the test statistic (154) falls within the rejection region, that is

$$\chi^2_{n-1}(\omega) \in \left(0,\, \chi^2_{\alpha/2,n-1,-}\right) \cup \left(\chi^2_{\alpha/2,n-1,-},\, +\infty\right), \tag{156}$$

or, in terms of the p-value corresponding to the realization of the test statistic, we reject equivalently the null hypothesis whether we have

$$2\min\left\{\mathbf{P}\left(\chi^2_{n-1} \leq \chi^2_{n-1}(\omega)\right), \mathbf{P}\left(\chi^2_{n-1} \geq \chi^2_{n-1}(\omega)\right)\right\} \leq \alpha/2 \tag{157}$$

or not. Approximating to the 6th decimal place, With reference to the data set *Gauss_1* [resp. *Gauss_2*], the null hypothesis $H_0 : \sigma_X = 1$, and the significance level $\alpha = 0.05$, we obtain

$$\chi^2_{n-1}(\omega) = \frac{(n-1)\, s^2_{X,n}}{\sigma^2_X} \approx \frac{149 * 1.256825}{1} \approx 187.2669 \quad [\text{resp.}\chi^2_{n-1}(\omega) = \frac{(n-1)\, s^2_{X,n}}{\sigma^2_X} \approx \frac{149 * 0.927501}{1} \approx 138.1976] \tag{158}$$

Now, with regard to the *Gauss_1* data set, we have

$$\chi^2_{n-1}(\omega) \approx 187.2669 > 184.687 \approx \chi^2_{149,0.025,+}. \tag{159}$$

Hence, we have to reject the null hypothesis $H_0 : \sigma_X = 1$ at the 0.05 significance level.

On the contrary, with regard to the *Gauss_2* data set, we have

$$\chi^2_{149,0.025,-} \approx 117.098 < \chi^2_{n-1}(\omega) \approx 138.1976 < 184.687 \approx \chi^2_{149,0.025,+}, \tag{160}$$

Hence, we cannot reject the null hypothesis $H_0 : \sigma = 1$ at the 0.05 significance level.

On the other hand, still with regard to the *Gauss_1* data set, we have

$$\chi^2_{149,0.010,-} \approx 108.2912 < \chi^2_{n-1}(\omega) \approx 187.2669 < 197.2112 \approx \chi^2_{149,0.010,+}. \tag{161}$$

Therefore, we cannot reject the null hypothesis $H_0 : \sigma = 1$ at the 0.01 significance level. Note that the 0.01 significance level corresponds to the 99% condidence interval.

In terms of p-value, with reference to the data set *Gauss_1*, we have

$$2\min\left\{\mathbf{P}\left(\chi^2_{n-1} \leq \chi^2_{n-1}(\omega)\right), \mathbf{P}\left(\chi^2_{n-1} \geq \chi^2_{n-1}(\omega)\right)\right\} \approx 2\min\{0.981647, 0.018353\} = 0.036706 < 0.05, \tag{162}$$

Thus, we can reject the null hypothesis at the 0.05 significance level.

On the contrary, With reference to to the data set *Gauss_2*, we have

$$2\min\left\{\mathbf{P}\left(\chi^2_{n-1} \leq \chi^2_{n-1}(\omega)\right), \mathbf{P}\left(\chi^2_{n-1} \geq \chi^2_{n-1}(\omega)\right)\right\} \approx 2\min\{0.273357, 0.726643\} = 0.546714 > 0.05, \tag{163}$$

Thus, we cannot reject the null hypothesis at the 0.05 significance level.

On the other hand, still with reference to the data set *Gauss_1*, we have

$$2\min\left\{\mathbf{P}\left(\chi^2_{n-1} \leq \chi^2_{n-1}(\omega)\right), \mathbf{P}\left(\chi^2_{n-1} \geq \chi^2_{n-1}(\omega)\right)\right\} \approx 2\min\{0.981647, 0.018353\} = 0.036706 > 0.025, \tag{164}$$

which confirms that we cannot reject the null hypothesis at the 0.01 significance level.

Also in this case, by applying the R command *varTest(x, alternative=... sigma.squared=1, conf.level=...)*, we can perform directly the variance test. In fact,

```
# library(EnvStats)
Gauss_1_chisq_test <- varTest(Gauss_1, alternative="two.sided", sigma.squared=1, conf.level=0.95)
Gauss_2_chisq_test <- varTest(Gauss_2, alternative="two.sided", sigma.squared=1, conf.level=0.95)
show(c(round(Gauss_1_chisq_test$statistic, digits=6),round(Gauss_1_chisq_test$p.value, digits=6)))
```

```
## Chi-Squared
##  187.266931    0.036707
```

```
show(c(round(Gauss_2_chisq_test$statistic, digits=6),round(Gauss_2_chisq_test$p.value, digits=6)))
```

```
## Chi-Squared
##  138.197644    0.546715
```

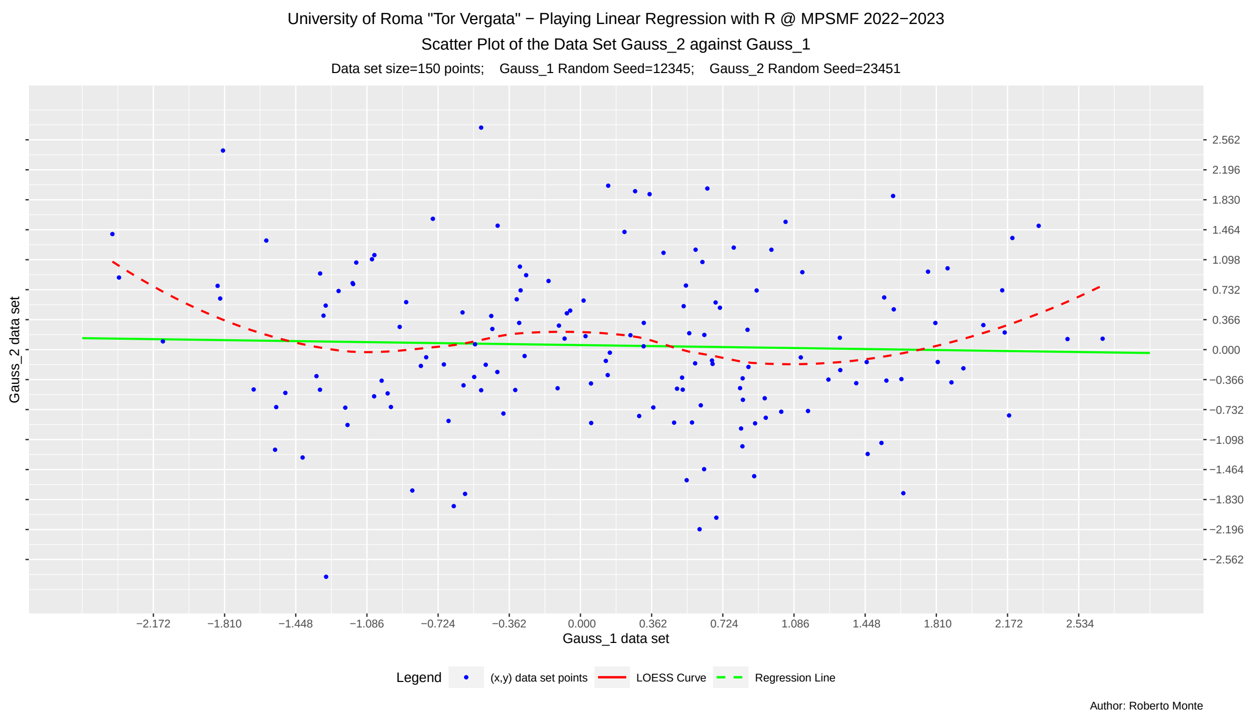
As the last issue, we need to check whether the data sets *Gauss_1* and *Gauss_2* are *cross-correlated* or not. That is to say whether they have been generated by independent standard Gaussian distribution.

We have found significant evidence that each of the data sets has been generated by independent sampling from the standard Gaussian distribution, namely the data sets *Gauss_1* and *Gauss_2* are not *auto-correlated*, but this does not clearly prevent the possibility of *cross-correlation*.

To get a visual evidence of possible cross-correlation, we inspect the scatter plot of the data sets *Gauss_2* against *Gauss_1*. A rather homogeneous cloud of points spread around an horizontal regression line would constitute visual evidence for the lack of cross correlation.

```
Data_df <- Gauss_df
n <- nrow(Data_df)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u0
paste("Scatter Plot of the Data Set Gauss_2 against Gauss_1")))
subtitle_content <- bquote(paste("Data set size=", .(n), " points;    Gauss_1 Random Seed=12345;    Gau
caption_content <- "Author: Roberto Monte"
x_name <- bquote("Gauss_1 data set")
y_name <- bquote("Gauss_2 data set")
x_breaks_num <- 15
x_binwidth <- round((max(Data_df$Z_1)-min(Data_df$Z_2))/x_breaks_num, digits=3)
x_breaks_low <- ceiling((min(Data_df$Z_1)/x_binwidth))*x_binwidth
x_breaks_up <- floor((max(Data_df$Z_1)/x_binwidth))*x_binwidth
x_breaks <- c(round(seq(from=x_breaks_low, to=x_breaks_up, by=x_binwidth),3))
x_labs <- format(x_breaks, scientific=FALSE)
J <- 1
x_lims <- c(x_breaks_low-J*x_binwidth,x_breaks_up+J*x_binwidth)
y_breaks_num <- 15
y_binwidth <- round((max(Data_df$Z_2)-min(Data_df$Z_2))/y_breaks_num, digits=3)
y_breaks_low <- ceiling((min(Data_df$Z_2)/y_binwidth))*y_binwidth
y_breaks_up <- floor((max(Data_df$Z_2)/y_binwidth))*y_binwidth
y_breaks <- c(round(seq(from=y_breaks_low, to=y_breaks_up, by=y_binwidth),3))
y_labs <- format(y_breaks, scientific=FALSE)
K <- 1
y_lims <- c((y_breaks_low-K*y_binwidth), (y_breaks_up+K*y_binwidth))
col_1 <- bquote("(x,y) data set points")
col_2 <- bquote("LOESS Curve")
col_3 <- bquote("Regression Line")
leg_labs <- c(col_1, col_2, col_3)
leg_cols <- c("col_1"="blue", "col_2"="red", "col_3"="green")
leg_ord <- c("col_1", "col_2", "col_3")
Data_df_01_sp <- ggplot(Data_df, aes(x=Z_1, y=Z_2)) +
  geom_smooth(alpha=1, size=0.8, linetype="solid", aes(color="col_3"),
              method="lm" , formula=y~x, se=FALSE, fullrange=TRUE) +
  geom_smooth(alpha=1, size=0.8, linetype="dashed", aes(color="col_2"),
              method="loess", formula=y~x, se=FALSE) +
  geom_point(alpha=1, size=1.0, shape=19, aes(color="col_1")) +
```

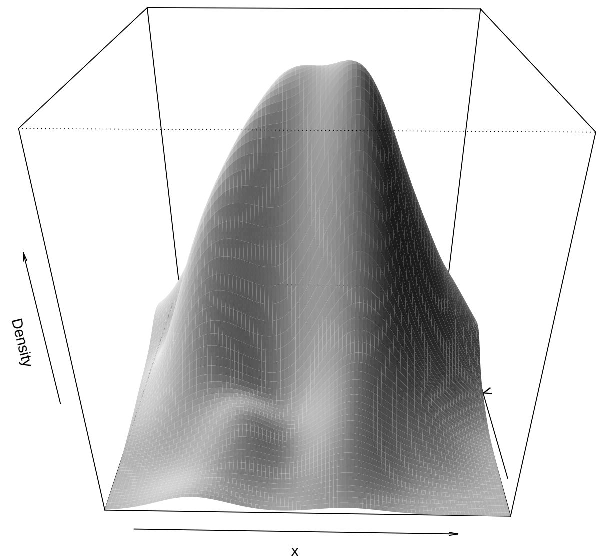
```
  scale_x_continuous(name=x_name, breaks=x_breaks, label=x_labs, limits=x_lims) +
  scale_y_continuous(name=y_name, breaks=y_breaks, labels=NULL, limits=y_lims,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  scale_colour_manual(name="Legend", labels=leg_labs, values=leg_cols, breaks=leg_ord,
                      guide=guide_legend(override.aes=list(shape=c(19,NA,NA),
                                                          linetype=c("blank", "solid", "dashed")))) +
  theme(plot.title=element_text(hjust=0.5), plot.subtitle=element_text(hjust=0.5),
        axis.text.x=element_text(angle=0, vjust=1),
        legend.key.width=unit(1.0,"cm"), legend.position="bottom")
plot(Data_df_01_sp)
```



As expected the scatter plot shows a cloud of points spread around an almost horizontal regression line.

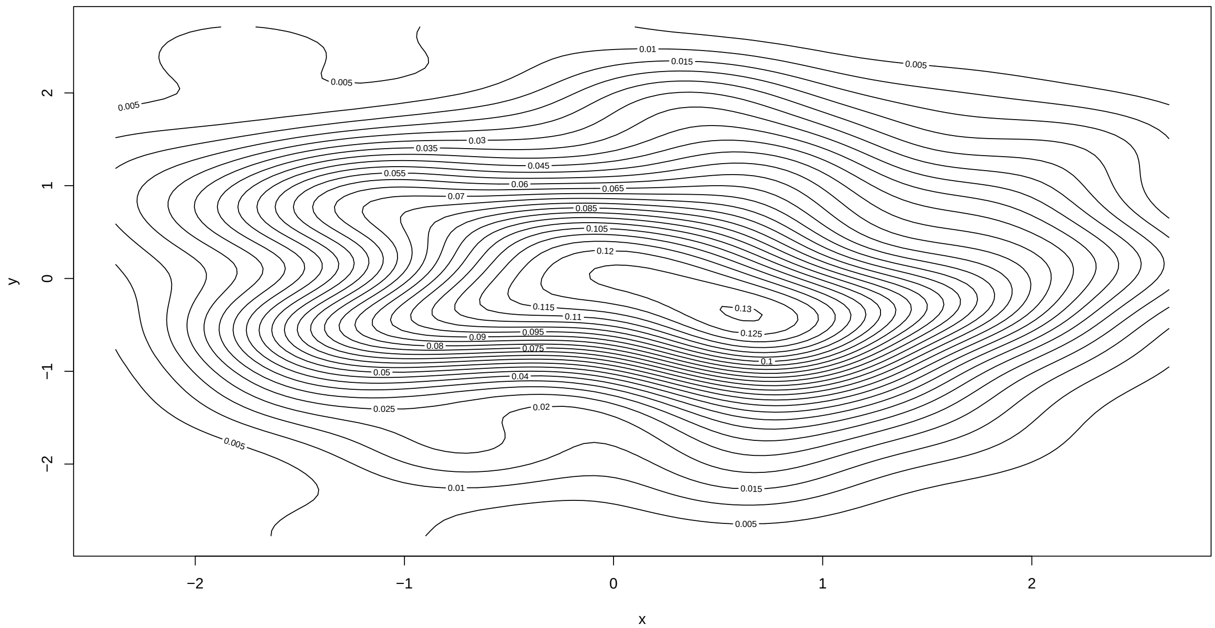
We also consider a graphical representation of the joint distribution of the data sets **x** and **y** by the function *mvn(x, multivariatePlot=...)* in the library *MVN*.

```
# library(MVN)
x <- Gauss_df$Z_1
y <- Gauss_df$Z_2
X_Y_MVN_mat <- cbind(x,y)
mvn(X_Y_MVN_mat, multivariatePlot = "persp") # draw a perspective plot
```




```
## $multivariateNormality
##            Test        HZ   p value MVN
## 1 Henze-Zirkler 0.6191172 0.4295288 YES
##
## $univariateNormality
##               Test  Variable Statistic   p value Normality
## 1 Anderson-Darling     x        0.5118    0.1923    YES
## 2 Anderson-Darling     y        0.3236    0.5225    YES
##
## $Descriptives
##     n      Mean   Std.Dev      Median       Min      Max       25th      75th
## x 150 0.1591299 1.1210821  0.26611244 -2.380358 2.655788 -0.6643145 0.8763541
## y 150 0.0503411 0.9630685 -0.05741142 -2.773087 2.710826 -0.5199724 0.7208519
##          Skew    Kurtosis
## x -0.02140620 -0.70338190
## y  0.03222489  0.08223607
```

```
mvn(X_Y_MVN_mat, multivariatePlot = "contour") # draw a contour plot
```



```
## $multivariateNormality
##            Test        HZ   p value MVN
## 1 Henze-Zirkler 0.6191172 0.4295288 YES
##
## $univariateNormality
##               Test  Variable Statistic   p value Normality
## 1 Anderson-Darling     x        0.5118    0.1923    YES
## 2 Anderson-Darling     y        0.3236    0.5225    YES
##
## $Descriptives
##     n      Mean   Std.Dev      Median       Min      Max       25th      75th
## x 150 0.1591299 1.1210821  0.26611244 -2.380358 2.655788 -0.6643145 0.8763541
## y 150 0.0503411 0.9630685 -0.05741142 -2.773087 2.710826 -0.5199724 0.7208519
##          Skew    Kurtosis
## x -0.02140620 -0.70338190
## y  0.03222489  0.08223607
```

The contour lines of the multivariate plot appear to be somewhat elliptic with horizontal orientation of the major axis. The same orientation to that of the regression line. This confirms the lack of correlation between the distributions generating the data sets *Gauss_1* and *Gauss_2*. Note that we should not expect circular contour lines since the standard deviations of the data sets have been estimated to be different.

Writing X [resp. Y] for the random variable whose distribution generates the *Gauss_1* [resp. *Gauss_2*] data set, the cross-correlation of the data set *Gauss*_1 and *Gauss*_2 is given by

$$r_{X,Y,n} = \frac{\sum_{k=1}^{n} (x_k - \bar{x}_n)(y_k - \bar{y}_n)}{\sqrt{\sum_{k=1}^{n} (x_k - \bar{x}_n)^2}\sqrt{\sum_{k=1}^{n} (y_k - \bar{y}_n)^2}}, \tag{165}$$

where

$$\bar{x}_n = \frac{1}{n}\sum_{k=1}^{n} x_k \quad \text{and} \quad \bar{y}_n = \frac{1}{n}\sum_{k=1}^{n} y_k \tag{166}$$

From a computational point of view, the correlation between the data sets *Gauss_1* and *Gauss_2* can be checked by the Pearson correlation test. This test measures the linear dependence between two data sets. It is a parametric correlation test because it depends on the distribution of the data. It can be used only when the data sets x and y are generated by Gaussian distributions.

Given two random variables X and Y and considering the sample correlation of X and $Y which is the statistic $R_{X,Y,n}$ given by

$$R_{X,Y,n} = \frac{\sum_{k=1}^{n} \left(X_k - \bar{X}_n\right)\left(Y_k - \bar{Y}_n\right)}{\sqrt{\sum_{k=1}^{n} \left(X_k - \bar{X}_n\right)^2}\sqrt{\sum_{k=1}^{n} \left(Y_k - \bar{Y}_n\right)^2}}, \tag{167}$$

where $\bar{X}_n$ [resp. $\bar{Y}_n$] is the sample mean of X [resp. Y] and $X_1, \dots, X_n$ [resp. $Y_1, \dots, Y_n$] is a simple random sample drawn from X [resp. Y], under the null hypothesis that two random variables X and Y are independent and Gaussian distributed, it is possible to prove that the statistic $T_{n-2}(X,Y)$ given by

$$T_{X,Y,n-2} \stackrel{\text{def}}{=} R_{X,Y,n}\sqrt{\frac{n-2}{1-R_{X,Y,n}^2}}, \tag{168}$$

has the Student's t-distribution with $n-2$ degrees of freedom.

We have

```
Corr_Z_1_Z_2 <- cor.test(Gauss_df$Z_1,Gauss_df$Z_2, method="pearson")
show(Corr_Z_1_Z_2)
```

```
##
##  Pearson's product-moment correlation
##
## data:  Gauss_df$Z_1 and Gauss_df$Z_2
## t = -0.47279, df = 148, p-value = 0.6371
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.1978636  0.1221885
## sample estimates:
##         cor
## -0.03883349
```

Therefore, by applying the Pearson correlation test, we cannot reject the null hypothesis that the data sets *Gauss_1* and *Gauss_2* have been generated by the realizations of two independent standard Gaussian random variables Z_1 and Z_2.

In light of what shown above, we are in a position to exploit the data sets *Gauss_1* and *Gauss_2* to generate other data sets that will turn out to be jointly Gaussian distributed.

For instance, choosing the parameters

$$\mu_1 \equiv 5, \quad \mu_2 \equiv 3, \quad a_{1,1} \equiv 1, \quad a_{1,2} \equiv 2, \quad a_{2,1} \equiv -1, \quad a_{2,2} \equiv 1, \tag{169}$$

we are led to think that the vector (X,Y) given by Equation @ref is jointly Gaussian distributed and we have

$$\mathbf{E}[X] = 5, \quad \mathbf{D}^2[X] = 5, \quad \mathbf{E}[Y] = 3, \quad \mathbf{D}^2[Y] = 2, \tag{170}$$

and

$$Cov(X,Y) = 1, \tag{171}$$

In addition,

$$Corr(X,Y) = \frac{Cov(X,Y)}{\mathbf{D}[X]\,\mathbf{D}[Y]} = \frac{1}{\sqrt{2}\sqrt{5}} = 0.316228. \tag{172}$$

From computational point of view, since the data sets *Gauss_1* and *Gauss_2* can be tought as the realizations of simple random samples drawn from the independent standard Gaussian random variables, say Z_1 and Z_2, on the occurring of some random outcome ω, we can exploit *Gauss_1* and *Gauss_2* to build the data sets $\mathbf{x}$ and $\mathbf{y}$ corresponding to the realizations of simple random samples drawn from the jointly Gaussian random variables X and Y given by Equation @ref on the occurring of ω.

```
Z_1 <- Gauss_1
Z_2 <- Gauss_2
mu_1 <- 5
mu_2 <- 3
a_1_1 <- 1
a_1_2 <- 2
a_2_1 <- -1
a_2_2 <- 1
X <- mu_1 + a_1_1*Z_1 + a_1_2*Z_2
Y <- mu_2 + a_2_1*Z_1 + a_2_2*Z_2
```

We add the data sets $\mathbf{x}$ and $\mathbf{y}$ to the *Gauss_df* data frame.

```
Gauss_df <- add_column(Gauss_df, X=X, Y=Y, .after="Z_2")
head(Gauss_df)
```

```
##   k        Z_1        Z_2        X         Y  Shuff_Z_1  Shuff_Z_2
## 1 1  0.5855288  1.2202517 8.026032 3.6347229 -1.2937153 -0.1328113
## 2 2  0.7094660  0.5116104 6.732687 2.8021444 -0.5403861  0.8004549
## 3 3 -0.1093033  0.2933357 5.477368 3.4026390  0.8237953  1.5974201
## 4 4 -0.4534972  0.4111048 5.368712 3.8646020  1.8869469 -0.5700748
## 5 5  0.6058875 -2.1928016 1.220284 0.2013109  1.4027054 -0.3789933
## 6 6 -1.8179560  2.4308397 8.043723 7.2487956  0.6873321  0.5378371
```

We also add the standardization of the data sets $\mathbf{x}$ and $\mathbf{y}$ to the *Gauss_df* data frame. Such standardized data sets will be exploited in place of X and Y for visual and computational tests of the null hypothesis of Gaussianity. Clearly, the rejection [resp. non-rejection] of the null hypothesis of Gaussianity for either of the standardized data sets will imply the rejection [resp. non-rejection] of the same hypothesis for the corresponding non-standardized data set.

```
X_st <- (X-mean(X))/sd(X)
Gauss_df <- add_column(Gauss_df, X_st=X_st, .after="X")
Y_st <- (Y-mean(Y))/sd(Y)
Gauss_df <- add_column(Gauss_df, Y_st=Y_st, .after="Y")
head(Gauss_df)
```

```
##   k        Z_1        Z_2        X        X_st         Y        Y_st  Shuff_Z_1
## 1 1  0.5855288  1.2202517 8.026032  1.26271699 3.6347229  0.49368381 -1.2937153
## 2 2  0.7094660  0.5116104 6.732687  0.67233403 2.8021444 -0.05913944 -0.5403861
## 3 3 -0.1093033  0.2933357 5.477368  0.09930942 3.4026390  0.33958260  0.8237953
## 4 4 -0.4534972  0.4111048 5.368712  0.04971055 3.8646020  0.64632107  1.8869469
## 5 5  0.6058875 -2.1928016 1.220284 -1.84395317 0.2013109 -1.78606513  1.4027054
## 6 6 -1.8179560  2.4308397 8.043723  1.27079259 7.2487956  2.89338959  0.6873321
```

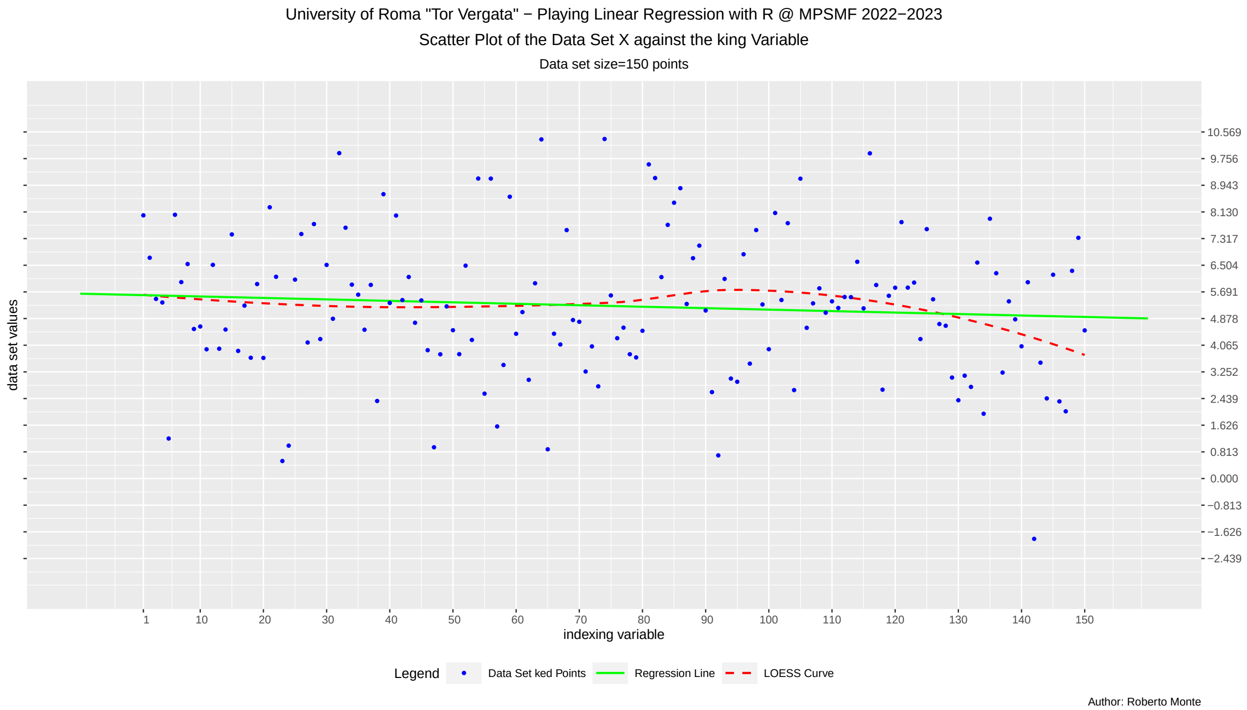
```
##    Shuff_Z_2
## 1 -0.1328113
## 2  0.8004549
## 3  1.5974201
## 4 -0.5700748
## 5 -0.3789933
## 6  0.5378371
```

We draw the scatter plots for the data sets X and Y.

The scatter plot for X

```
Data_df <- Gauss_df
n <- nrow(Data_df)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u00
paste("Scatter Plot of the Data Set X against the king Variable")))
subtitle_content <- bquote(paste("Data set size=", .(n), " points"))
caption_content <- "Author: Roberto Monte"
x_name <- bquote("indexing variable")
y_name <- bquote("data set values")
x_breaks_num <- 15
x_breaks_low <- Data_df$k[1]
x_breaks_up <- Data_df$k[n]
x_binwidth <- ceiling((x_breaks_up-x_breaks_low)/x_breaks_num)
x_breaks <- c(x_breaks_low,seq(from=x_binwidth, to=x_breaks_up, by=x_binwidth))
x_labs <- format(x_breaks, scientific=FALSE)
J <-1
x_lims <- c(x_breaks_low-J*x_binwidth, x_breaks_up+J*x_binwidth)
y_breaks_num <- 15
y_binwidth <- round((max(Data_df$X)-min(Data_df$X))/y_breaks_num, digits=3)
y_breaks_low <- floor((min(Data_df$X)/y_binwidth))*y_binwidth
y_breaks_up <- ceiling((max(Data_df$X)/y_binwidth))*y_binwidth
y_breaks <- c(round(seq(from=y_breaks_low, to=y_breaks_up, by=y_binwidth),3))
y_labs <- format(y_breaks, scientific=FALSE)
K <- 1
y_lims <- c((y_breaks_low-K*y_binwidth), (y_breaks_up+K*y_binwidth))
col_1 <- bquote("Data Set ked Points")
col_2 <- bquote("Regression Line")
col_3 <- bquote("LOESS Curve")
leg_labs <- c(col_1, col_2, col_3)
leg_cols <- c("col_1"="blue", "col_2"="green", "col_3"="red")
leg_ord <- c("col_1", "col_2", "col_3")
X_sp <- ggplot(Data_df, aes(x=k, y=X)) +
  geom_smooth(alpha=1, size=0.8, linetype="dashed", aes(color="col_3"),
              method="loess", formula=y ~ x, se=FALSE) +
  geom_smooth(alpha=1, size=0.8, linetype="solid", aes(color="col_2"),
              method="lm" , formula=y ~ x, se=FALSE, fullrange=TRUE) +
  geom_point(alpha=1, size=1.0, shape=19, aes(color="col_1")) +
  scale_x_continuous(name=x_name, breaks=x_breaks, label=x_labs, limits=x_lims) +
  scale_y_continuous(name=y_name, breaks=y_breaks, labels=NULL, limits=y_lims,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  scale_colour_manual(name="Legend", labels=leg_labs, values=leg_cols, breaks=leg_ord,
```

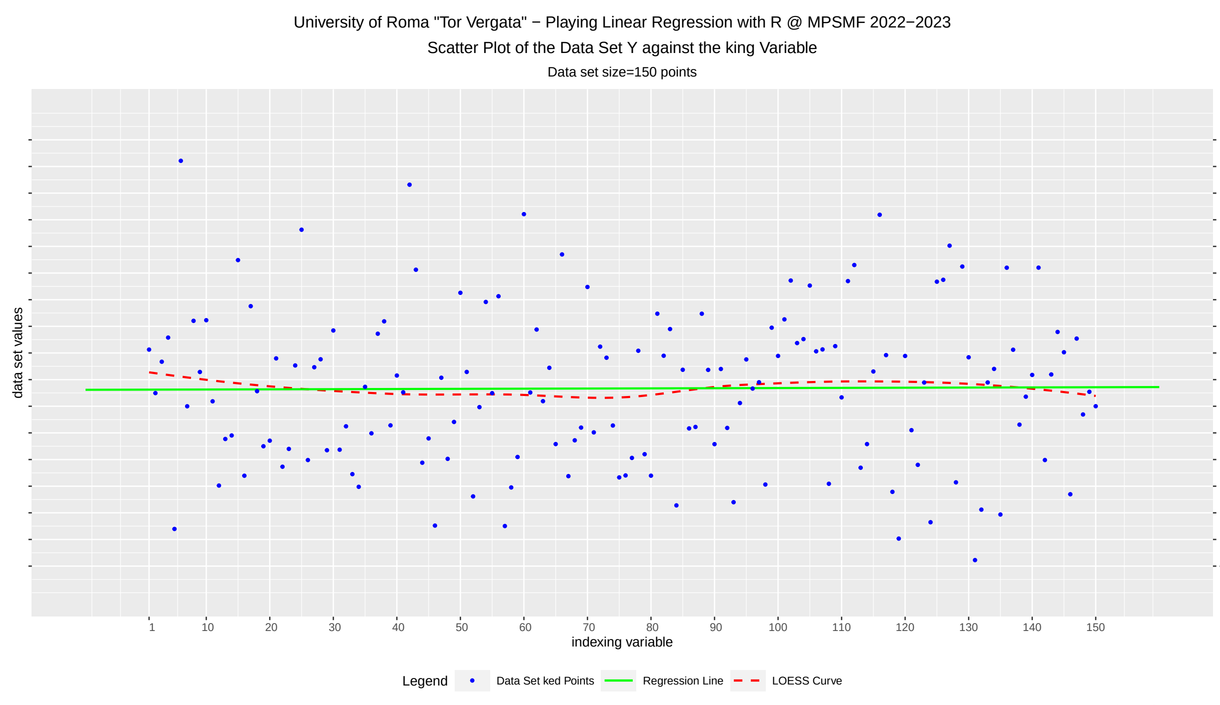
```
                    guide=guide_legend(override.aes=list(shape=c(19,NA,NA),
                                                        linetype=c("blank", "solid", "dashed")))) +
  theme(plot.title=element_text(hjust=0.5), plot.subtitle=element_text(hjust=0.5),
        axis.text.x=element_text(angle=0, vjust=1),
        legend.key.width=unit(1.0,"cm"), legend.position="bottom")
plot(X_sp)
```



The scatter plots of Y data set.

```
Data_df <- Gauss_df
n <- nrow(Data_df)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u00
paste("Scatter Plot of the Data Set Y against the king Variable")))
subtitle_content <- bquote(paste("Data set size=", .(n), " points"))
caption_content <- "Author: Roberto Monte"
x_name <- bquote("indexing variable")
y_name <- bquote("data set values")
x_breaks_num <- 15
x_breaks_low <- Data_df$k[1]
x_breaks_up <- Data_df$k[n]
x_binwidth <- ceiling((x_breaks_up-x_breaks_low)/x_breaks_num)
x_breaks <- c(x_breaks_low,seq(from=x_binwidth, to=x_breaks_up, by=x_binwidth))
x_labs <- format(x_breaks, scientific=FALSE)
J <-1
x_lims <- c(x_breaks_low-J*x_binwidth, x_breaks_up+J*x_binwidth)
y_breaks_num <- 15
y_binwidth <- round((max(Data_df$Y)-min(Data_df$Y))/y_breaks_num, digits=3)
y_breaks_low <- floor((min(Data_df$Y)/y_binwidth))*y_binwidth
y_breaks_up <- ceiling((max(Data_df$Y)/y_binwidth))*y_binwidth
y_breaks <- c(round(seq(from=y_breaks_low, to=y_breaks_up, by=y_binwidth),3))
```

```
y_labs <- format(y_breaks, scientific=FALSE)
K <- 1
y_lims <- c((y_breaks_low-K*y_binwidth), (y_breaks_up+K*y_binwidth))
col_1 <- bquote("Data Set ked Points")
col_2 <- bquote("Regression Line")
col_3 <- bquote("LOESS Curve")
leg_labs <- c(col_1, col_2, col_3)
leg_cols <- c("col_1"="blue", "col_2"="green", "col_3"="red")
leg_ord <- c("col_1", "col_2", "col_3")
Y_sp <- ggplot(Data_df, aes(x=k, y=Y)) +
  geom_smooth(alpha=1, size=0.8, linetype="dashed", aes(color="col_3"),
              method="loess", formula=y ~ x, se=FALSE) +
  geom_smooth(alpha=1, size=0.8, linetype="solid", aes(color="col_2"),
              method="lm" , formula=y ~ x, se=FALSE, fullrange=TRUE) +
  geom_point(alpha=1, size=1.0, shape=19, aes(color="col_1")) +
  scale_x_continuous(name=x_name, breaks=x_breaks, label=x_labs, limits=x_lims) +
  scale_y_continuous(name=y_name, breaks=y_breaks, labels=NULL, limits=y_lims,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  scale_colour_manual(name="Legend", labels=leg_labs, values=leg_cols, breaks=leg_ord,
                      guide=guide_legend(override.aes=list(shape=c(19,NA,NA),
                                                            linetype=c("blank", "solid", "dashed")))) +
  theme(plot.title=element_text(hjust=0.5), plot.subtitle=element_text(hjust=0.5),
        axis.text.x=element_text(angle=0, vjust=1),
        legend.key.width=unit(1.0,"cm"), legend.position="bottom")
plot(Y_sp)
```



Comparing the scatter plots of both data sets X and Y with the scatter plots of the data sets *Gauss_1* and *Gauss_2*, the main, perhaps only, visual difference is in the intercept of the regression lines, However, we proceed by applying the ADF test.

```
# library(urca) # The library for this vesion of the test.
z <- Gauss_df$X   # Choosing the data set to be tested.
no_lags <- 0      # Setting the lag parameter for the test.

X_DF_none <- ur.df(z, type="none", lags=no_lags, selectlags="Fixed")
# Applying the form of the DF test which takes as the null hypothesis that the data set is generated by
# a process with a random walk component, while the alternative hypothesis is that the data set is gene
# by an autoregressive process with no drift and trend.
summary(X_DF_none) # Showing the result of the test
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression none
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 - 1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.8378 -0.9534  0.5515  2.6764  7.9821
##
## Coefficients:
##         Estimate Std. Error t value Pr(>|t|)
## z.lag.1 -0.15599    0.04338  -3.596  0.00044 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.02 on 148 degrees of freedom
## Multiple R-squared:  0.08033,    Adjusted R-squared:  0.07412
## F-statistic: 12.93 on 1 and 148 DF,  p-value: 0.0004404
##
##
## Value of test-statistic is: -3.5955
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau1 -2.58 -1.95 -1.62
```

```
# library(urca)# The library for this vesion of the test.
z <- Gauss_df$Y    # Choosing the data set to be tested.
no_lags <- 0       # Setting the lag parameter for the test.

Y_DF_none <- ur.df(z, type="none", lags=no_lags, selectlags="Fixed")
# Applying the form of the DF test which takes as the null hypothesis that the data set is generated by
# a process with a random walk component, while the alternative hypothesis is that the data set is gene
# by an autoregressive process with no drift and trend.
summary(Y_DF_none) # Showing the result of the test
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression none
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 - 1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.3471 -0.6131  0.6724  1.8987  7.0930
##
## Coefficients:
##         Estimate Std. Error t value Pr(>|t|)
## z.lag.1 -0.22596    0.05177  -4.365 2.38e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.061 on 148 degrees of freedom
## Multiple R-squared:  0.114,  Adjusted R-squared:  0.1081
## F-statistic: 19.05 on 1 and 148 DF,  p-value: 2.375e-05
##
##
## Value of test-statistic is: -4.3649
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau1 -2.58 -1.95 -1.62
```

Note that for both data sets X and Y we have the rejection of the null hypothesis of unit root at at the significance level $\alpha = 0.01$ or $\alpha = 1\%$. However, on one side in the item "Call:" the formula notifies that the linear regression exploited in the test is given by

$$Z_k - Z_{k-1} = \beta_1 Z_{k-1} + U_k,$$

where Z_k [resp. U_k] is the kth sample component of the random variable Z which generates the data set [the error term U], on varying of the indexing variable $k = 1, \dots, n$ (the meaning of the "-1" is that the intercept β_0 is forced to be 0); on the other hand the items "…R-squared:" show in both cases a value rather close to zero. In addition, the item "Residuals:" shows a velue for the median somewhat far from 0. These findings suggests that the linear regression exploited in the test is not entirely appropriate. Therefore, we re-apply the test modifying the options in the function *ur.df*. In particular, we change the *type* opion from *none* to *drift*, to account for a possible non null intercept β_0.

```
# library(urca) # The library for this vesion of the test.
z <- Gauss_df$X   # Choosing the data set to be tested.
no_lags <- 0      # Setting the lag parameter for the test.

X_DF_none <- ur.df(z, type="drift", lags=no_lags, selectlags="Fixed")
# Applying the form of the DF test which takes as the null hypothesis that the data set is generated by
# a process with a random walk component, while the alternative hypothesis is that the data set is gene
# by an autoregressive process with no drift and trend.
summary(X_DF_none) # Showing the result of the test
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression drift
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -7.058 -1.320  0.079  1.232  5.123
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.40843    0.46770   11.56   <2e-16 ***
## z.lag.1     -1.03176    0.08202  -12.58   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.193 on 147 degrees of freedom
## Multiple R-squared:  0.5184, Adjusted R-squared:  0.5151
## F-statistic: 158.2 on 1 and 147 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic is: -12.5788 79.1221
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau2 -3.46 -2.88 -2.57
## phi1  6.52  4.63  3.81
```

```
# library(urca)# The library for this vesion of the test.
z <- Gauss_df$Y    # Choosing the data set to be tested.
no_lags <- 0       # Setting the lag parameter for the test.

Y_DF_none <- ur.df(z, type="drift", lags=no_lags, selectlags="Fixed")
# Applying the form of the DF test which takes as the null hypothesis that the data set is generated by
# a process with a random walk component, while the alternative hypothesis is that the data set is gene
# by an autoregressive process with no drift and trend.
summary(Y_DF_none) # Showing the result of the test
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression drift
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.2507 -1.1379  0.0305  0.9790  4.2251
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.03393    0.26851   11.30   <2e-16 ***
## z.lag.1     -1.05105    0.08232  -12.77   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.513 on 147 degrees of freedom
## Multiple R-squared:  0.5259, Adjusted R-squared:  0.5226
## F-statistic:   163 on 1 and 147 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic is: -12.7683 81.516
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau2 -3.46 -2.88 -2.57
## phi1  6.52  4.63  3.81
```

In this form the ADF test produces more sounding results. In fact, in the item "Call:" the formula notifies that the linear regression exploited in the test is given by

$$Z_k - Z_{k-1} = \beta_0 + \beta_1 Z_{k-1} + U_k,$$

where Z_k [resp. U_k] is the kth sample component of the random variable Z which generates the data set [the error term U], on varying of the indexing variable $k = 1, \dots, n$ (the meaning of the "+1" is that the intercept β_0 is not forced to be 0). Now, for both the data sets X and Y, the items "…R-squared:" show a significantly higher value than the former linear regression and the item "Residuals:" shows a value of the Median closer to zero. These findings suggest that the linear regression exploited in this form of the test is much more appropriate than the former. Referring to the X data set the item "Coefficients:" notifies us that we have the estimate

$$\hat{\beta}_0 = 5.40843 \quad \text{and} \quad \hat{\beta}_1 = -1.03176$$

This means that it is more significant that the data sets X is generated by a random sample $(X_k)_{k=1}^n$ drawn from X which satisfies the equation

$$X_k = 5.40843 - 0.03176 X_{k-1} + U_k$$

where U_k is the kth sample component of the error term U, on varying of the indexing variable $k = 1, \dots, n$. Similarly, referring to the Y data set the item "Coefficients:" notifies us that we have the estimate

$$\hat{\beta}_0 = 3.03393 \quad \text{and} \quad \hat{\beta}_1 = -1.05105$$

This means that it is more significant that the data sets Y is generated by a random sample $(Y_k)_{k=1}^n$ drawn from Y which satisfies the equation

$$Y_k = 3.03393 - 0.05105 Y_{k-1} + V_k$$

where V_k is the kth sample component of the error term V, on varying of the indexing variable $k = 1, \dots, n$.

The above equations suggest that the data sets $\mathbf{x}$ and $\mathbf{y}$ are essentially estimated as noises with a drift.

In this form of the ADF test the rejection of the null hypothesis of unit root at the significance level $\alpha = 1\%$ in favor of the alternative is even stronger.

We then apply the $KPSS$ test to the X data set

```
# library(urca)# The library for this version of the test
z <- Gauss_df$X    # Choosing the data set to be tested

X_KPSS_mu <- ur.kpss(z, type="mu", lags="nil", use.lag=NULL)
# Applying the simplest form of the KPSS test which considers the hull hypothesis that the data set is 
# by an autoregressive process with constant mean, while the alternative hypothesis is that the data se
# generated a process with a random walk component.

summary(X_KPSS_mu)# Showing the result of the test
```

```
##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: mu with 0 lags.
##
## Value of test-statistic is: 0.1782
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

and to the Y data set.

```
# library(urca)# The library for this version of the test
z <- Gauss_df$Y    # Choosing the data set to be tested

Y_KPSS_mu <- ur.kpss(z, type="mu", lags="nil", use.lag=NULL)
# Applying the simplest form of the KPSS test which considers the hull hypothesis that the data set is 
# by an autoregressive process with constant mean, while the alternative hypothesis is that the data se
# generated a process with a random walk component.

summary(Y_KPSS_mu)# Showing the result of the test
```

```
##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: mu with 0 lags.
##
## Value of test-statistic is: 0.0387
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

For both the data sets **x** and **y** the null hypothesis of generation by a stationary process cannot rejected at the significance level $\alpha = 1\%$. Therefore we can reject the null hypothesis in favor of the alternative.

The rejection of the null hypothesis of random walk component in the random sample generating the data sets **x** and **y** in favor of the alternative hypothesis of autoregression with drift but no trend, by the *ADF* test jointly with the lack of rejection of the null hypothesis of autoregression without trend against the alternative hypothesis of random walk component in the random sample generating the data sets **x** and **y** by the *KPSS* test constitute a significant computational evidence that the data sets **x** and **y** have been generated by processes with constant mean.

To deal with the issue of constant variance, besides the visual inspection of the scatterplots of the data sets X and Y, we apply the *Breusch-Pagan* (*BP*) and *White* (W) test.

The (unstudentized) *BP* test on the data set **x**.

```
# Unstudentized Breusch-Pagan test
x <- Gauss_df$k    # The independent variable in the test
y <- Gauss_df$X        # The dependent variable in the test
# Checking the empirical kurtosis of the data set according to which to select the option Studentize
# of the BP and W test (TRUE if the kurtosis is >> 0).
# library(EnvStats)# The library for the kurtosis function.
EnvStats::kurtosis(y, method="moment", excess=TRUE)
```

```
## [1] 0.1675436
```

```
# library(lmtest)# The library for this version of the test.
# The function for the BP test, which stores the result in the Gauss_1_BP list.
X_BP <- bptest(formula=y~x, varformula=NULL, studentize=FALSE)
show(X_BP) # The summary of the X_BP list.
```

```
##
##  Breusch-Pagan test
##
## data:  y ~ x
## BP = 0.041485, df = 1, p-value = 0.8386
```

The (unstudentized) W test on the data set **x**.

```
# library(lmtest) # The library for this version of the test.
# White test
x <- Gauss_df$k
y <- Gauss_df$X
var.formula <- ~ x+I(x^2) # The formula which allows to switch from *BP* to *W* test.
# The function for the W test, which stores the result in the Gauss_1_W list.
X_W <- bptest(formula=y~x, varformula=var.formula, studentize=FALSE)
show(X_W) # The summary of the Gauss_1_W list.
```

```
##
##  Breusch-Pagan test
##
## data:  y ~ x
## BP = 0.49445, df = 2, p-value = 0.781
```

Both the BP and W test cannot reject the null hypothesis of homoskedasticity at any of the standard levels. In light of the visual inspections, and the results of the BP and W test, we have significant evidences to not reject the null hypothesis that the data set **x** has been generated by a process with constant variance.

The same result holds true for the data set **y**.

The (unstudentized) BP test on the data set **y**.

```
# Unstudentized Breusch-Pagan test
x <- Gauss_df$k    # The independent variable in the test
y <- Gauss_df$Y        # The dependent variable in the test
# Checking the empirical kurtosis of the data set according to which to select the option Studentize
# of the BP and W test (TRUE if the kurtosis is >> 0).
# library(EnvStats)# The library for the kurtosis function.
EnvStats::kurtosis(y, method="moment", excess=TRUE)
```

```
## [1] -0.2589339
```

```
# library(lmtest)# The library for this version of the test.
# The function for the BP test, which stores the result in the Gauss_1_BP list.
Y_BP <- bptest(formula=y~x, varformula=NULL, studentize=FALSE)
show(Y_BP) # The summary of the Gauss_1_BP list.
```

```
##
##  Breusch-Pagan test
##
## data:  y ~ x
## BP = 0.00090234, df = 1, p-value = 0.976
```

The (unstudentized) W test on the data set **y**.

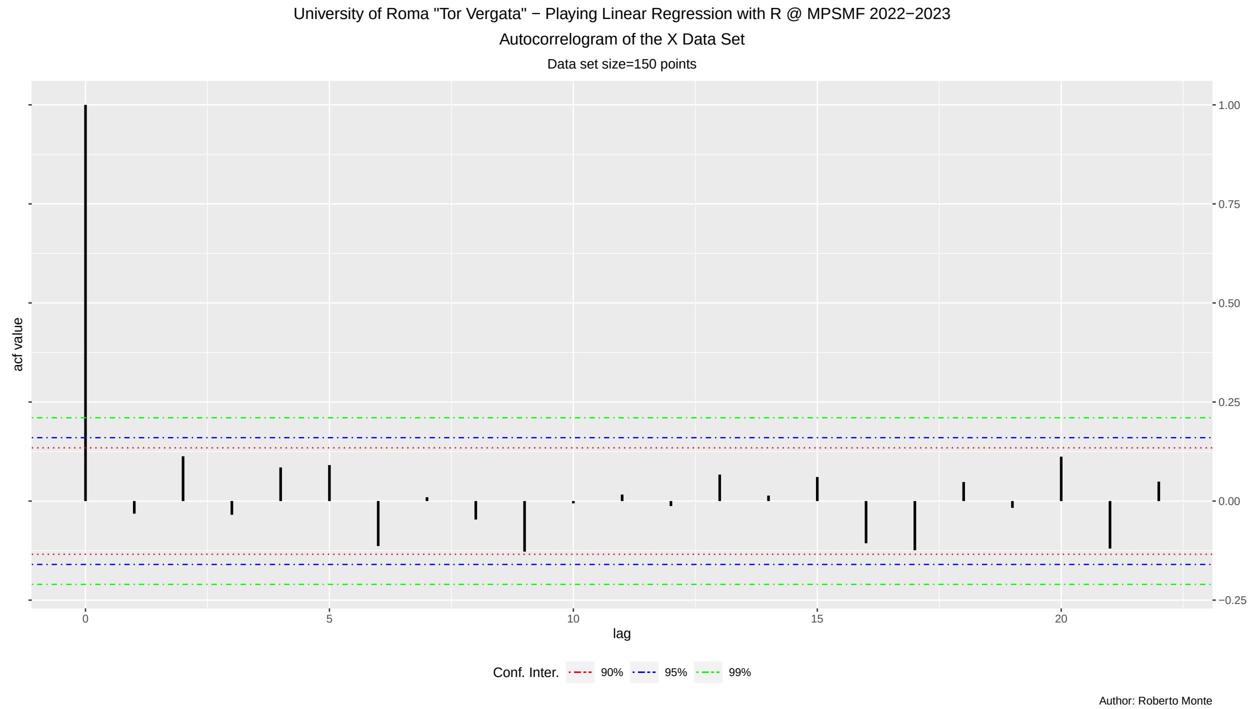
```
# library(lmtest) # The library for this version of the test.
# White test
x <- Gauss_df$k
y <- Gauss_df$Y
var.formula <- ~ x+I(x^2)# The formula which allows to switch from *BP* to *W* test.
# The function for the W test, which stores the result in the Gauss_1_W list.
Y_W <- bptest(formula=y~x, varformula=var.formula, studentize=FALSE)
show(Y_W) # The summary of the Gauss_1_W list.
```

```
##
##  Breusch-Pagan test
##
## data:  y ~ x
## BP = 0.26363, df = 2, p-value = 0.8765
```

We now consider the issue that the data sets **x** and **y** have been generated by independent random sampling from the same distribution (not necessarily Gaussian). Also in this case we consider a visual check by plotting the autocorrelogram and the partial autocorrelogram of the data sets.

The autocorrelogram of the X data set.

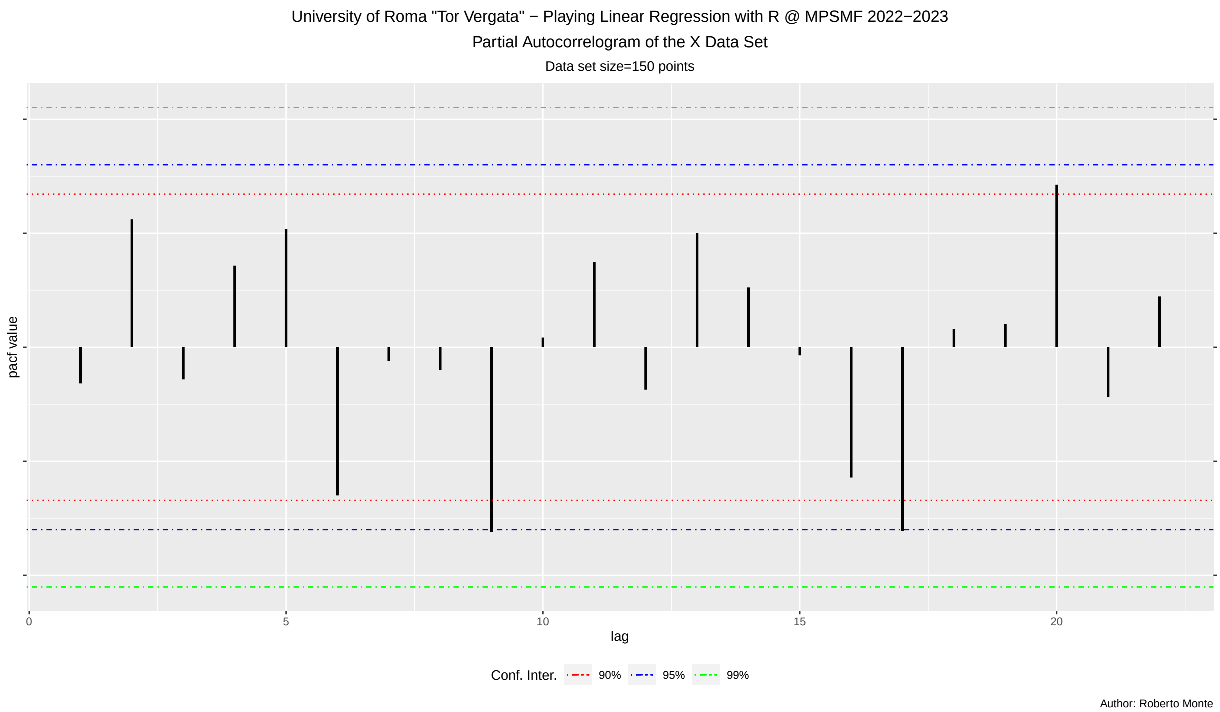
```
z <- Gauss_df$X
n <- length(z)
maxlag <- ceiling(10*log10(n))
Aut_Fun_z <- acf(z, lag.max=maxlag, type="correlation", plot=FALSE)
ci_90 <- qnorm((1+0.90)/2)/sqrt(n)
ci_95 <- qnorm((1+0.95)/2)/sqrt(n)
ci_99 <- qnorm((1+0.99)/2)/sqrt(n)
Plot_Aut_Fun_z <- data.frame(lag=Aut_Fun_z$lag, acf=Aut_Fun_z$acf)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u0
paste("Autocorrelogram of the X Data Set")))
subtitle_content <- bquote(paste("Data set size=", .(n), " points"))
caption_content <- "Author: Roberto Monte"
ggplot(Plot_Aut_Fun_z, aes(x=lag, y=acf)) +
  geom_segment(aes(x=lag, y=rep(0,length(lag)), xend=lag, yend=acf), size=1, col="black") +
  geom_hline(aes(yintercept=-ci_90, color="CI_90"), show.legend=TRUE, lty=3) +
  geom_hline(aes(yintercept=ci_90, color="CI_90"), lty=3) +
  geom_hline(aes(yintercept=ci_95, color="CI_95"), show.legend=TRUE, lty=4) +
  geom_hline(aes(yintercept=-ci_95, color="CI_95"), lty=4) +
  geom_hline(aes(yintercept=-ci_99, color="CI_99"), show.legend=TRUE, lty=4) +
  geom_hline(aes(yintercept=ci_99, color="CI_99"), lty=4) +
  scale_x_continuous(name="lag", breaks=waiver(), label=waiver()) +
  scale_y_continuous(name="acf value", breaks=waiver(), labels=NULL,
                     sec.axis=sec_axis(~., breaks=waiver(), labels=waiver())) +
  scale_color_manual(name="Conf. Inter.", labels=c("90%","95%","99%"),
                     values=c(CI_90="red", CI_95="blue", CI_99="green")) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  theme(plot.title=element_text(hjust=0.5),
        plot.subtitle=element_text(hjust= 0.5),
        plot.caption=element_text(hjust=1.0),
        legend.key.width=unit(0.8,"cm"), legend.position="bottom")
```



The partial autocorrelogram of the data set *Gauss_1*.

```
z <- Gauss_df$X
n <- length(z)
maxlag <- ceiling(10*log10(n))
P_Aut_Fun_z <- pacf(z, lag.max=maxlag, type="correlation", plot=FALSE)
ci_90 <- qnorm((1+0.90)/2)/sqrt(n)
ci_95 <- qnorm((1+0.95)/2)/sqrt(n)
ci_99 <- qnorm((1+0.99)/2)/sqrt(n)
Plot_P_Aut_Fun_z <- data.frame(lag=P_Aut_Fun_z$lag, pacf=P_Aut_Fun_z$acf)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u00
paste("Partial Autocorrelogram of the X Data Set")))
subtitle_content <- bquote(paste("Data set size=", .(n), " points"))
caption_content <- "Author: Roberto Monte"
ggplot(Plot_P_Aut_Fun_z, aes(x=lag, y=pacf)) +
  geom_segment(aes(x=lag, y=rep(0,length(lag)), xend=lag, yend=pacf), size=1, col="black") +
  geom_hline(aes(yintercept=-ci_90, color="CI_90"), show.legend=TRUE, lty=3) +
  geom_hline(aes(yintercept=ci_90, color="CI_90"), lty=3) +
  geom_hline(aes(yintercept=ci_95, color="CI_95"), show.legend=TRUE, lty=4) +
  geom_hline(aes(yintercept=-ci_95, color="CI_95"), lty=4) +
  geom_hline(aes(yintercept=-ci_99, color="CI_99"), show.legend=TRUE, lty=4) +
  geom_hline(aes(yintercept=ci_99, color="CI_99"), lty=4) +
  scale_x_continuous(name="lag", breaks=waiver(), label=waiver()) +
  scale_y_continuous(name="pacf value", breaks=waiver(), labels=NULL,
                     sec.axis=sec_axis(~., breaks=waiver(), labels=waiver())) +
  scale_color_manual(name="Conf. Inter.", labels=c("90%","95%","99%"),
                     values=c(CI_90="red", CI_95="blue", CI_99="green")) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  theme(plot.title=element_text(hjust=0.5),
        plot.subtitle=element_text(hjust= 0.5),
```

```
        plot.caption=element_text(hjust=1.0),
        legend.key.width=unit(0.8,"cm"), legend.position="bottom")
```

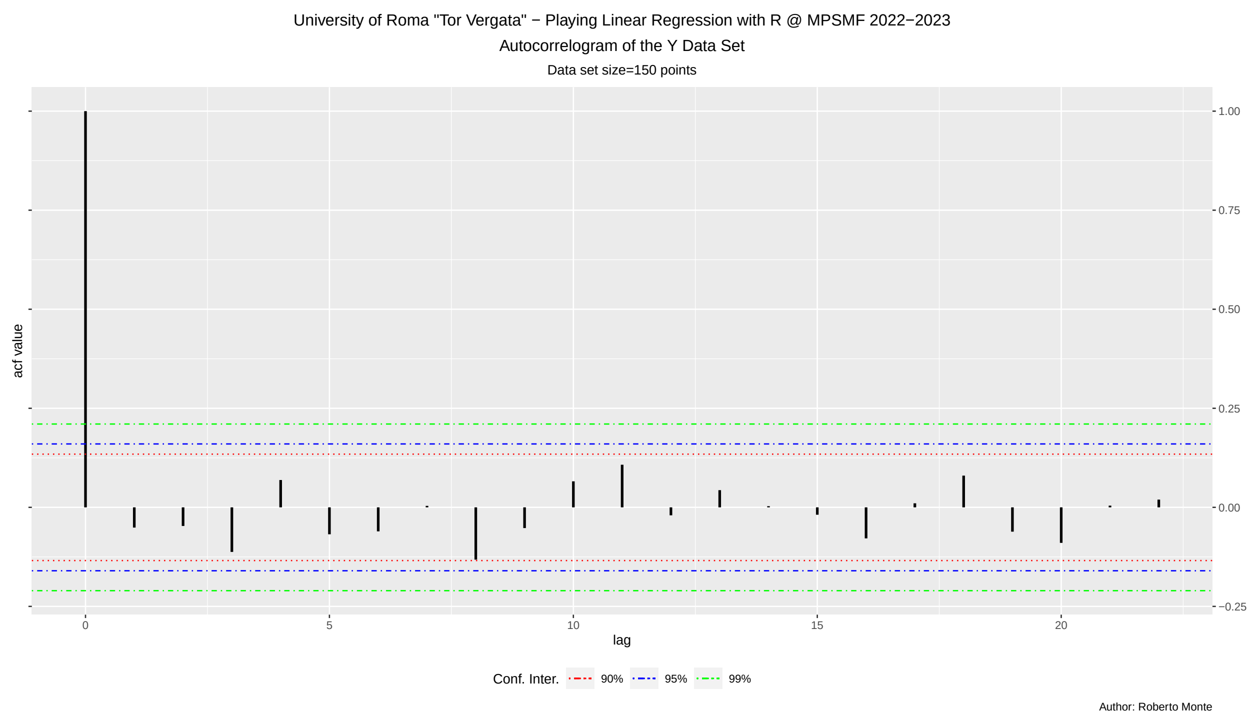


With reference to the X autocorrelogram, the number of peaks corresponding to positive lags crossing the confidence lines is within the statistical tolerance. In fact, we have no peaks crossing both the 95% confidence lines (the tolerance is *floor(maxlag* $*0.05$*)=floor(*$22 * 0.05$*)*=1 and the 90% confidence lines (the tolerance is *floor(maxlag* $*0.10$*)=floor(*$22 * 0.10$*)*=2. With reference to the X partial autocorrelogram the visual evidence is not clear as well, we still have no peaks clearly crossing the 95% confidence line (the tolerance is still *floor(maxlag* $*0.05$*)*=1*)*, but two peaks almost do it. In addition, three peaks clearly cross the 90% confidence line (the tolerance is still *floor(maxlag* $*0.10$*)*=2*)*. Summariziong, we have visual evidence that the data set X_d has been generated by independent random sampling from the same distribution at the nearly 95% confidence level.

The autocorrelogram of the Y data set.

```
z <- Gauss_df$Y
n <- length(z)
maxlag <- ceiling(10*log10(n))
Aut_Fun_z <- acf(z, lag.max=maxlag, type="correlation", plot=FALSE)
ci_90 <- qnorm((1+0.90)/2)/sqrt(n)
ci_95 <- qnorm((1+0.95)/2)/sqrt(n)
ci_99 <- qnorm((1+0.99)/2)/sqrt(n)
Plot_Aut_Fun_z <- data.frame(lag=Aut_Fun_z$lag, acf=Aut_Fun_z$acf)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u0(
paste("Autocorrelogram of the Y Data Set")))
subtitle_content <- bquote(paste("Data set size=", .(n), " points"))
caption_content <- "Author: Roberto Monte"
ggplot(Plot_Aut_Fun_z, aes(x=lag, y=acf)) +
  geom_segment(aes(x=lag, y=rep(0,length(lag)), xend=lag, yend=acf), size=1, col="black") +
  geom_hline(aes(yintercept=-ci_90, color="CI_90"), show.legend=TRUE, lty=3) +
  geom_hline(aes(yintercept=ci_90, color="CI_90"), lty=3) +
```

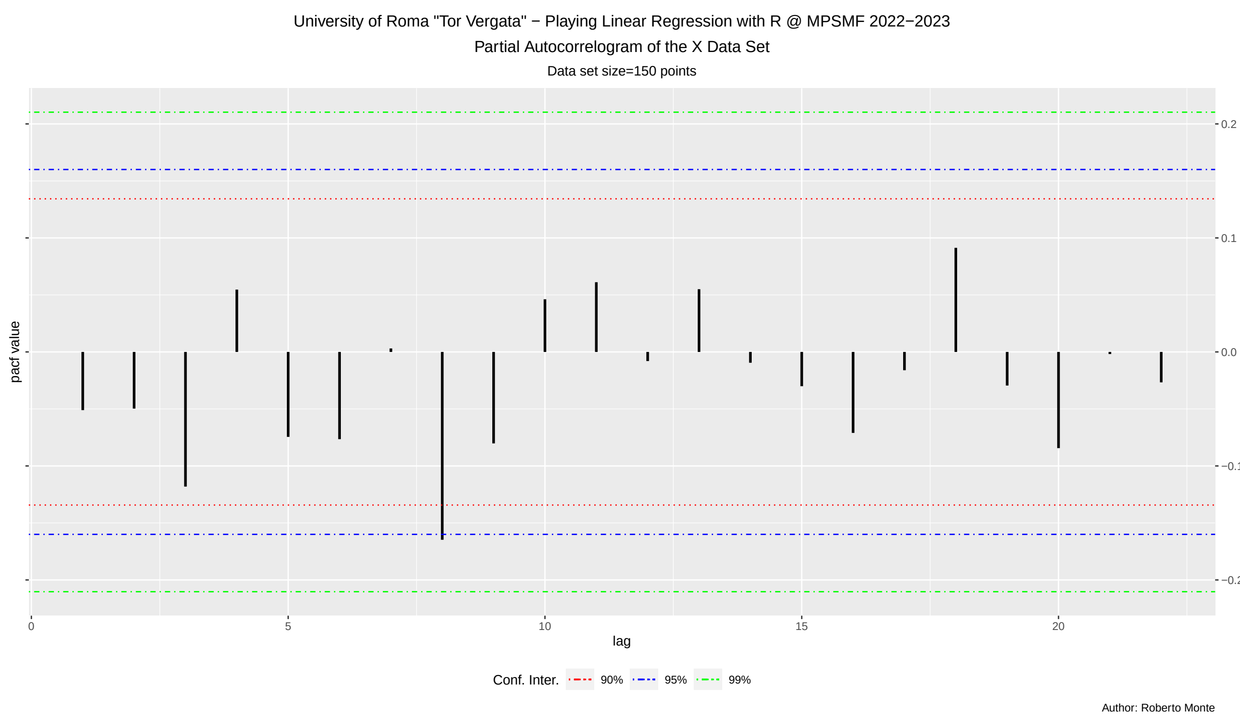
```
  geom_hline(aes(yintercept=ci_95, color="CI_95"), show.legend=TRUE, lty=4) +
  geom_hline(aes(yintercept=-ci_95, color="CI_95"), lty=4) +
  geom_hline(aes(yintercept=-ci_99, color="CI_99"), show.legend=TRUE, lty=4) +
  geom_hline(aes(yintercept=ci_99, color="CI_99"), lty=4) +
  scale_x_continuous(name="lag", breaks=waiver(), label=waiver()) +
  scale_y_continuous(name="acf value", breaks=waiver(), labels=NULL,
                     sec.axis=sec_axis(~., breaks=waiver(), labels=waiver())) +
  scale_color_manual(name="Conf. Inter.", labels=c("90%","95%","99%"),
                     values=c(CI_90="red", CI_95="blue", CI_99="green")) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  theme(plot.title=element_text(hjust=0.5),
        plot.subtitle=element_text(hjust= 0.5),
        plot.caption=element_text(hjust=1.0),
        legend.key.width=unit(0.8,"cm"), legend.position="bottom")
```



The partial autocorrelogram of the *Gauss_2* data set.

```
z <- Gauss_df$Y
n <- length(z)
maxlag <- ceiling(10*log10(n))
P_Aut_Fun_z <- pacf(z, lag.max=maxlag, type="correlation", plot=FALSE)
ci_90 <- qnorm((1+0.90)/2)/sqrt(n)
ci_95 <- qnorm((1+0.95)/2)/sqrt(n)
ci_99 <- qnorm((1+0.99)/2)/sqrt(n)
Plot_P_Aut_Fun_z <- data.frame(lag=P_Aut_Fun_z$lag, pacf=P_Aut_Fun_z$acf)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u00
paste("Partial Autocorrelogram of the X Data Set")))
subtitle_content <- bquote(paste("Data set size=", .(n), " points"))
caption_content <- "Author: Roberto Monte"
```

```
ggplot(Plot_P_Aut_Fun_z, aes(x=lag, y=pacf)) +
  geom_segment(aes(x=lag, y=rep(0,length(lag)), xend=lag, yend=pacf), size=1, col="black") +
  geom_hline(aes(yintercept=-ci_90, color="CI_90"), show.legend=TRUE, lty=3) +
  geom_hline(aes(yintercept=ci_90, color="CI_90"), lty=3) +
  geom_hline(aes(yintercept=ci_95, color="CI_95"), show.legend=TRUE, lty=4) +
  geom_hline(aes(yintercept=-ci_95, color="CI_95"), lty=4) +
  geom_hline(aes(yintercept=-ci_99, color="CI_99"), show.legend=TRUE, lty=4) +
  geom_hline(aes(yintercept=ci_99, color="CI_99"), lty=4) +
  scale_x_continuous(name="lag", breaks=waiver(), label=waiver()) +
  scale_y_continuous(name="pacf value", breaks=waiver(), labels=NULL,
                     sec.axis=sec_axis(~., breaks=waiver(), labels=waiver())) +
  scale_color_manual(name="Conf. Inter.", labels=c("90%","95%","99%"),
                     values=c(CI_90="red", CI_95="blue", CI_99="green")) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  theme(plot.title=element_text(hjust=0.5),
        plot.subtitle=element_text(hjust= 0.5),
        plot.caption=element_text(hjust=1.0),
        legend.key.width=unit(0.8,"cm"), legend.position="bottom")
```



With reference to the Y autocorrelogram and partial autocorrelogram, we ahave visual evidence visual evidence that the data set has been generated by independent random sampling from the same distribution at the 90% confidence level.

Note that the lack of autocorrelation in the data sets $\mathbf{x}$ and $\mathbf{y}$ is clearly an expected finding, due to the mechanism we used to generate them.

We can also apply the Ljung-Box (LB) test. We have

```
z <- Gauss_df$X
n <- length(z)
```

```
maxlag <- ceiling(10*log10(n))
X_LB <- Box.test(z, lag=maxlag, type="Ljung-Box")
show(X_LB)
```

```
##
##  Box-Ljung test
##
## data:  z
## X-squared = 21.435, df = 22, p-value = 0.494
```

and

```
z <- Gauss_df$Y
n <- length(z)
maxlag <- ceiling(10*log10(n))
Y_LB <- Box.test(z, lag=maxlag, type="Ljung-Box")
show(Y_LB)
```

```
##
##  Box-Ljung test
##
## data:  z
## X-squared = 15.333, df = 22, p-value = 0.8476
```

As a consequence, from the application of the LB test to both the data sets X_LB and Y_LB, we cannot reject the null hypothesis that each them has been generated by independent random sampling from the same distribution, at the significance level of 95%.

We also apply the runs test to the X data set

```
# library(randtests)
z <- Gauss_df$X
X_median_WW <- randtests::runs.test(z, alternative="two.sided", threshold=median(z), pvalue='normal', pl
show(X_median_WW)
```

```
##
##  Runs Test
##
## data:  z
## statistic = 0, runs = 76, n1 = 75, n2 = 75, n = 150, p-value = 1
## alternative hypothesis: nonrandomness
```

```
X_mean_WW <- randtests::runs.test(z, alternative="two.sided", threshold=mean(z), pvalue='normal', plot=
show(X_mean_WW)
```

```
##
##  Runs Test
##
## data:  z
## statistic = 0.33668, runs = 78, n1 = 77, n2 = 73, n = 150, p-value =
## 0.7364
## alternative hypothesis: nonrandomness
```

which does not reject the null hypothesis at the signinficance level $\alpha = 0.10$, and to the Y data set.

```
# library(randtests)
z <- Gauss_df$Y
Y_median_WW <- randtests::runs.test(z, alternative="two.sided", threshold=median(z), pvalue='normal', pl
show(Y_median_WW)
```

```
##
##  Runs Test
##
## data:  z
## statistic = 1.9662, runs = 88, n1 = 75, n2 = 75, n = 150, p-value =
## 0.04928
## alternative hypothesis: nonrandomness
```

```
Y_mean_WW <- randtests::runs.test(z, alternative="two.sided", threshold=mean(z), pvalue='normal', plot=
show(Y_mean_WW)
```

```
##
##  Runs Test
##
## data:  z
## statistic = 1.9662, runs = 88, n1 = 75, n2 = 75, n = 150, p-value =
## 0.04928
## alternative hypothesis: nonrandomness
```

which, somewhat surprisingly, rejects the null hypothesis only at the signinficance level $\alpha = 0.01$.

However, summarizing the above results, we cannot reject the hypothesis that the data sets $\mathbf{x}$ and $\mathbf{y}$ have been generated by random sampling from the same distribution.

Now, we perform a visual test about the alleged Gaussianity of the data set $\mathbf{x}$ and Y.

As a first step we consider the statistical summary of the two standardized data sets.

```
z <- Gauss_df$X
z_st <- (z-mean(z))/sd(z)
summary(z_st)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -3.24085 -0.61150  0.02567  0.00000  0.57312  2.32548
```

We also consider the empirical standard deviation, skewness and kurtosis which are not included in the summary.

```
X_sd <- sd(z)
X_skew <- EnvStats::skewness(z, method="moment")
X_kurt <- EnvStats::kurtosis(z, method="moment", excess=TRUE)
show(c(round(X_sd,4), round(X_skew,4), round(X_kurt,4)))
```

```
## [1]  2.1907 -0.0385  0.1675
```

The six summary points of the X data set do not fit perfectly the corresponding summary points of the centered Gaussian distribution with standard deviation *X_sd*. However, the fit is not bad either. In fact, for such a Gaussian distribution we have the following summary points (to be compared to the *Gauss_1* summary points in the last row):

$$\begin{array}{cccccc} Min\,(99.73\%) & 1stQ & Median & Mean & 3rdQ & Max\,(99.73\%) \\ -3.0000 & -0.6745 & 0.0000 & 0.0000 & 0.6745 & 3.0000 \\ -3.2408 & -0.6115 & 0.0257 & 0.0000 & 0.5731 & 2.3255 \end{array} \tag{173}$$

where

$$\begin{aligned} &Min\,(99.73\%) = Mean - 3*sd, \qquad Max\,(99.73\%) = Mean + 3*sd, \\ &1stQ = qnorm(0.25,\ mean = 0,\ sd = 1,\ lower.tail = TRUE), \\ &33dQ = qnorm(0.75,\ mean = 0,\ sd = 1,\ lower.tail = TRUE), \end{aligned} \tag{174}$$

In addition, the skewness and the excess kurtosis of the standard Gaussian distribution are both equal to 0.

A similar result we have for the *Gauss_2* data set.

```
z <- Gauss_df$Y
z_st <- (z-mean(z))/sd(z)
summary(z_st)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -2.181910 -0.760183  0.009278  0.000000  0.614440  2.893390
```

```
Y_sd <- sd(z)
Y_skew <- EnvStats::skewness(z, method="moment")
Y_kurt <- EnvStats::kurtosis(z, method="moment", excess=TRUE)
show(c(round(Y_sd,4), round(Y_skew,4), round(Y_kurt,4)))
```

```
## [1]  1.5060  0.2818 -0.2589
```

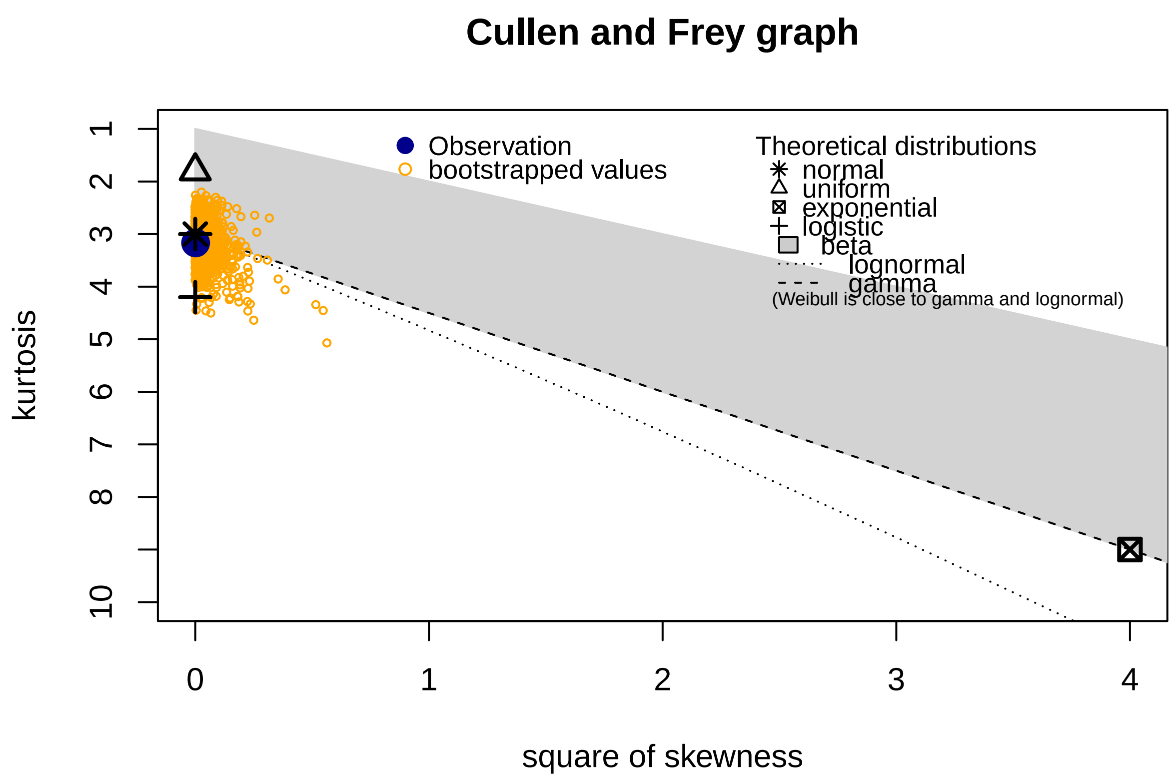
Note that also the six summary points of the *Gauss_2* data set do not fit perfectly the corresponding summary points of the standard Gaussian distribution but the fit is not bad either.

$$\begin{array}{cccccc} Min\,(99.73\%) & 1stQ & Median & Mean & 3rdQ & Max\,(99.73\%) \\ -3.0000 & -0.6745 & 0.0000 & 0.0000 & 0.6745 & 3.0000 \\ -2.1819 & -0.7602 & 0.0093 & 0.0000 & 0.6144 & 2.8934 \end{array} \tag{175}$$

A Cullen-Frey graph can help to understand better the relationship between the distributions of the data sets X, Y and a theoretical distribution of reference.

The Cullen-Frey graph for the data set **x**.

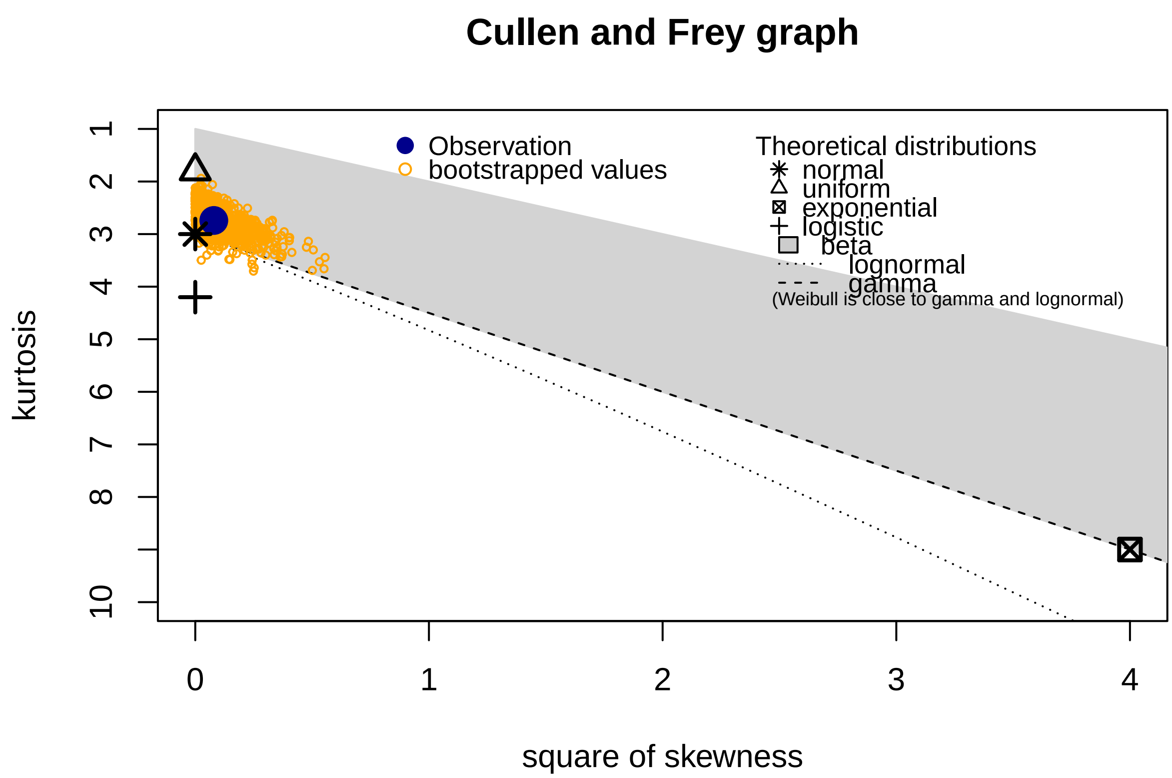
```
# library(fitdistrplus)
z <- Gauss_df$X
descdist(z, discrete = FALSE, method = "sample", graph = TRUE, boot=1000)
```




```
## summary statistics
## ------
## min:  -1.839889   max:  10.35422
## median:  5.316054
## mean:  5.259812
## sample sd:  2.183374
## sample skewness:  -0.03852825
## sample kurtosis:  3.167544
```

The Cullen-Frey graph for the data set **y**.

```
# library(fitdistrplus)
z <- Gauss_df$Y
descdist(z, discrete = FALSE, method = "sample", graph = TRUE, boot=1000)
```




```
## summary statistics
## ------
## min:  -0.3948503   max:  7.248796
## median:  2.905184
## mean:  2.891211
## sample sd:  1.50102
## sample skewness:  0.2818103
## sample kurtosis:  2.741066
```

From the inspection of the Cullen-Frey graphs the suspect arises that the standardized data sets **x** and **y** might be Gaussian distributed. In particular, for the former the suspect is rather strong.

We draw the *Q–Q* and the *P–P* plot for the data sets **x** and **y** against the standard Gaussian distribution. WE follow the same procedure exploited in case of the data sets *Gauss_1* and *Gauss_2*.

```
z <- Gauss_df$X                                                    # X data set.
z_cent <- z - mean(z)                                  # Centered X data set.
z_cent_qemp <- qemp(ppoints(length(z)), z_cent) # Empirical quantiles of the centered X data set.
Gauss_quants <- qnorm(ppoints(length(z)), mean=0, sd=1) # Quantiles of the standard Gaussian distributi
                                                           # corresponding to the empirical quantiles
X_QQ_plot_df <- data.frame(k=1:length(z), S=z_cent, X=Gauss_quants, Y=z_cent_qemp)
head(X_QQ_plot_df)
```

```
##   k          S         X         Y
## 1 1  2.7662202 -2.713052 -7.099701
## 2 2  1.4728747 -2.326348 -5.016239
```

```
## 3 3  0.2175560 -2.128045 -4.575491
## 4 4  0.1089003 -1.989313 -4.390783
## 5 5 -4.0395279 -1.880794 -4.314110
## 6 6  2.7839113 -1.790751 -4.262998
```

```
Data_df <- X_QQ_plot_df
n <- nrow(Data_df)
quart_probs <- c(0.25,0.75)
quart_Y <- as.vector(quantile(Data_df$Y, quart_probs))
quart_X <- qnorm(quart_probs, mean=0, sd=1)
slope <- diff(quart_Y)/diff(quart_X)
intercept <- quart_Y[1]-slope*quart_X[1]
distr <- "norm"
distr_pars <- list(mean=0, sd=1)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u0
paste("Q-Q plot (Normal Confidence Bands) of the Data Set X Against the Standard Gaussian Distribution")
subtitle_content <- bquote(paste("Data set size ", .(n), " sample points."))
caption_content <- "Author: Roberto Monte"
x_name <- bquote("Theoretical Quantiles")
y_name <- bquote("Sample Quantiles")
x_breaks_num <- 15 # (deduced from primeFactors(n))
# x_breaks_num <- ceiling(n^(1/2)) # Tukey & Mosteller square-root rule
# x_breaks_num <- ceiling(1+log2(n)) # Sturges rule
# x_breaks_num <- ceiling((2*n)^(1/3)) # Teller & Scott rice rule
x_binwidth <- round((max(Data_df$X)-min(Data_df$X))/x_breaks_num, digits=1)
x_breaks_low <- floor((min(Data_df$X)/x_binwidth))*x_binwidth
x_breaks_up <- ceiling((max(Data_df$X)/x_binwidth))*x_binwidth
x_breaks <- c(seq(from=x_breaks_low, to=x_breaks_up, by=x_binwidth))
x_labs <- format(x_breaks, scientific=FALSE)
J <- 1.0
x_lims <- c((x_breaks_low-J*x_binwidth), (x_breaks_up+J*x_binwidth))
y_breaks_num <- length(x_breaks)
y_binwidth <- round((max(Data_df$Y)-min(Data_df$Y))/y_breaks_num, digits=3)
y_breaks_low <- floor((min(Data_df$Y)/y_binwidth))*y_binwidth
y_breaks_up <- ceiling((max(Data_df$Y)/y_binwidth))*y_binwidth
y_breaks <- c(round(seq(from=y_breaks_low, to=y_breaks_up, by=y_binwidth),3))
y_labs <- format(y_breaks, scientific=FALSE)
K <- 1.5
y_lims <- c((y_breaks_low-K*y_binwidth), (y_breaks_up+K*y_binwidth))
y1_shape <- bquote("Q-Q plot")
y1_fill <- bquote("95% confidence bands")
y2_fill <- bquote("99% confidence bands")
col_1 <- bquote("interquartile line")
col_2 <- bquote("regression line")
col_3 <- bquote("y=x line")
leg_shape_labs <- y1_shape
leg_fill_labs <- c(y1_fill, y2_fill)
leg_col_labs <- c(col_1, col_2, col_3)
leg_shape_cols <- c("y1_shape" = 19)
leg_fill_cols <- c("y1_fill"="gold", "y2_fill"="green")
leg_col_cols <- c("col_1"="cyan", "col_2"="red", "col_3"="black")
leg_shape_sort <- "y1_shape"
leg_fill_sort <- c("y1_fill", "y2_fill")
leg_col_sort <- c("col_1", "col_2", "col_3")
```

```
X_QQ_norm_plot <- ggplot(Data_df, aes(sample=S)) +
  stat_qq_band(aes(fill="y2_fill"), distribution=distr, dparams=distr_pars, conf = 0.99, bandType = "poi
  stat_qq_band(aes(fill="y1_fill"), distribution=distr, dparams=distr_pars, conf = 0.95, bandType = "poi
#  stat_qq_line(aes(colour="col_1"), distribution=distr, dparams=distr_pars) +
  geom_abline(aes(slope=slope, intercept=intercept, colour="col_1"), size=0.8, linetype="solid", show.le
#  geom_segment(aes(x=X[1], xend=-X[1], y=X[1], yend=-X[1], colour="col_3"),
#               size=0.8, linetype="solid", show.legend=FALSE) +
  geom_abline(aes(slope=1, intercept=0, colour="col_3"), size=0.8, linetype="solid", show.legend=FALSE)
  stat_smooth(alpha=1, size=0.8, linetype="solid", aes(x=X, y=Y, colour="col_2"),
              method="lm" , formula=y~x, se=FALSE, fullrange=FALSE) +
  stat_qq_point(aes(shape="y1_shape"), colour="blue", alpha=1, size=1.0,
                distribution=distr, dparams=distr_pars) +
  scale_x_continuous(name=x_name, breaks=x_breaks, label=x_labs, limits=NULL) +
  scale_y_continuous(name=y_name, breaks=y_breaks, labels=NULL, limits=NULL,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  scale_shape_manual(name="Legend", labels=leg_shape_labs, values=leg_shape_cols, breaks=leg_shape_sort)
  scale_fill_manual(name="", labels=leg_fill_labs, values=leg_fill_cols, breaks=leg_fill_sort) +
  scale_colour_manual(name="", labels=leg_col_labs, values=leg_col_cols, breaks=leg_col_sort) +
  guides(shape=guide_legend(order=1), fill=guide_legend(order=2), colour=guide_legend(order=3)) +
  theme(plot.title=element_text(hjust=0.5, size=13.5), plot.subtitle=element_text(hjust=0.5),
        axis.text.x=element_text(angle=0, vjust=1),
        legend.key.width=unit(0.8,"cm"), legend.position="bottom")
plot(X_QQ_norm_plot)
```

```
## Warning: The following aesthetics were dropped during statistical transformation: sample
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a `group` aesthetic or to convert a numerical
##   variable into a factor?
```

We draw also the *P–P* plot with bootstrap confidence bands.

```
z <- Gauss_df$X_st                                        # Standardized X data set.
z_qemp <- qemp(ppoints(length(z)), z) # Empirical quantiles of the standardized X data set.
z_pemp <- pemp(z_qemp, z)           # Empirical probabilities of the standardized X data set
                                        # corresponding to the empirical quantiles.
Gauss_quants <- qnorm(ppoints(length(z)), mean=0, sd=1) # Quantiles of the standard Gaussian distributi
                                                # corresponding to the empirical quantiles.
Gauss_probs <- pnorm(Gauss_quants, mean=0, sd=1)# Probabilities of the standard Gaussian distribution
                                                # corresponding to the empirical probabilities.
X_PP_plot_df <- data.frame(k=1:length(z), S=z, X=Gauss_probs, Y=z_pemp)
head(X_PP_plot_df)
```

```
##   k           S           X           Y
## 1 1  1.26271699 0.003333333 0.004159734
## 2 2  0.67233403 0.010000000 0.010000000
## 3 3  0.09930942 0.016666667 0.016666667
## 4 4  0.04971055 0.023333333 0.023333333
## 5 5 -1.84395317 0.030000000 0.030000000
## 6 6  1.27079259 0.036666667 0.036666667
```

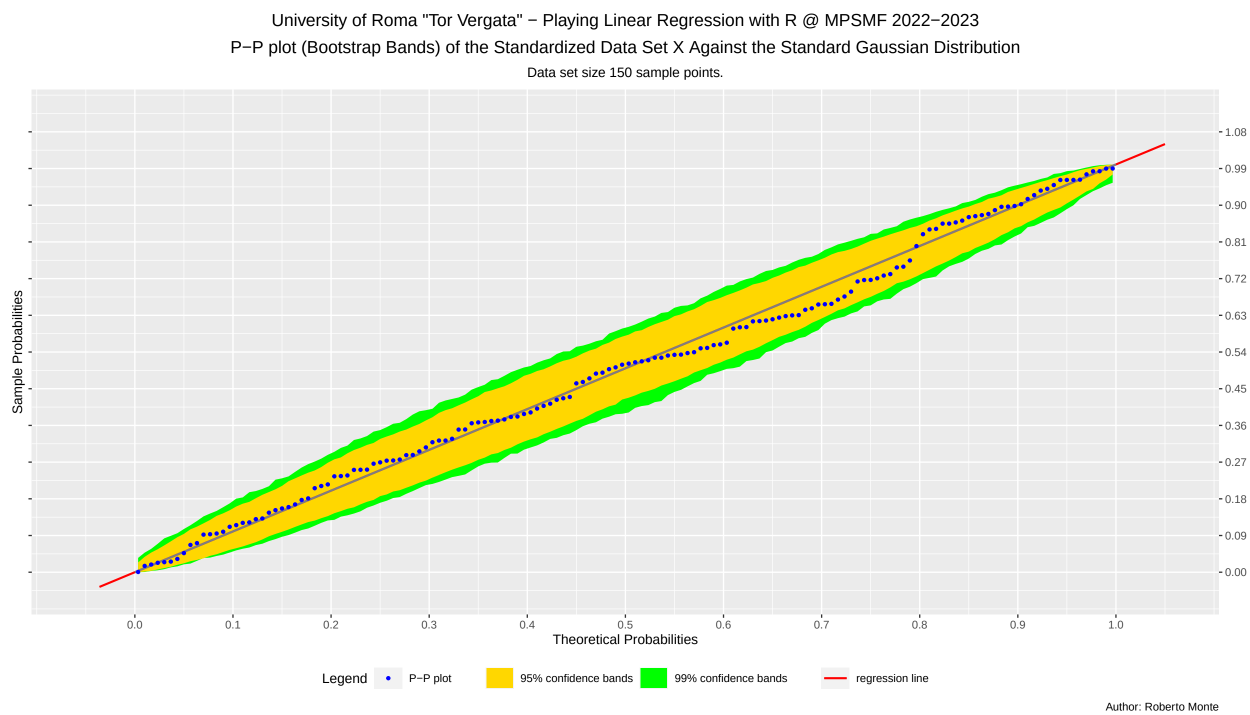
Second we draw the *P–P* plot of the residuals with boot bands (the only option available).

```
Data_df <- X_PP_plot_df
n <- nrow(Data_df)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u0
paste("P-P plot (Bootstrap Bands) of the Standardized Data Set X Against the Standard Gaussian Distribu
subtitle_content <- bquote(paste("Data set size ", .(n), " sample points."))
caption_content <- "Author: Roberto Monte"
x_name <- bquote("Theoretical Probabilities")
y_name <- bquote("Sample Probabilities")
```

```
x_breaks_num <- 15 # (deduced from primeFactors(n))
# x_breaks_num <- ceiling(n^(1/2)) # Tukey & Mosteller square-root rule
# x_breaks_num <- ceiling(1+log2(n)) # Sturges rule
# x_breaks_num <- ceiling((2*n)^(1/3)) # Teller & Scott rice rule
x_binwidth <- round((max(Data_df$X)-min(Data_df$X))/x_breaks_num, digits=1)
x_breaks_low <- floor((min(Data_df$X)/x_binwidth))*x_binwidth
x_breaks_up <- ceiling((max(Data_df$X)/x_binwidth))*x_binwidth
x_breaks <- c(seq(from=x_breaks_low, to=x_breaks_up, by=x_binwidth))
# x_breaks <- c(1,round(seq(from=x_breaks_low, to=x_breaks_up, by=x_binwidth),3),n)
x_labs <- format(x_breaks, scientific=FALSE)
J <- 0.5
x_lims <- c(x_breaks_low-J*x_binwidth, x_breaks_up+J*x_binwidth)
y_breaks_num <- length(x_breaks)
y_binwidth <- round((max(Data_df$Y)-min(Data_df$Y))/y_breaks_num, digits=3)
y_breaks_low <- floor((min(Data_df$Y)/y_binwidth))*y_binwidth
y_breaks_up <- ceiling((max(Data_df$Y)/y_binwidth))*y_binwidth
y_breaks <- c(round(seq(from=y_breaks_low, to=y_breaks_up, by=y_binwidth),3))
y_labs <- format(y_breaks, scientific=FALSE)
K <- 0.5
y_lims <- c(y_breaks_low-K*y_binwidth, y_breaks_up+K*y_binwidth)
y1_shape <- bquote("P-P plot")
y1_fill <- bquote("95% confidence bands")
y2_fill <- bquote("99% confidence bands")
col_1 <- bquote("y=x line")
col_2 <- bquote("regression line")
leg_shape_labs <- y1_shape
leg_fill_labs <- c(y1_fill, y2_fill)
leg_col_labs <- c(col_1, col_2)
leg_shape_cols <- c("y1_shape" = 19)
leg_fill_cols <- c("y1_fill"="gold", "y2_fill"="green")
leg_col_cols <- c("col_1"="black", "col_2"="red")
leg_shape_sort <- "y1_shape"
leg_fill_sort <- c("y1_fill", "y2_fill")
leg_col_sort <- c("col_1", "col_2")
distr <- "norm"
distr_pars <- list(mean=0, sd=1)
X_PP_boot_plot <- ggplot(Data_df, aes(sample=S)) +
  stat_pp_band(aes(fill="y2_fill"), distribution=distr, dparams=distr_pars, conf = 0.99, bandType = "boo
  stat_pp_band(aes(fill="y1_fill"), distribution=distr, dparams=distr_pars, conf = 0.95, bandType = "boo
  stat_smooth(alpha=1, size=0.8, linetype="solid", aes(x=X, y=Y, colour="col_2"),
              method="lm" , formula=y~x, se=FALSE, fullrange=TRUE) +
  stat_pp_line(aes(colour="col_1")) +
  stat_pp_point(aes(shape="y1_shape"), colour="blue", alpha=1, size=1.0,
                distribution=distr, dparams=distr_pars) +
  scale_x_continuous(name=x_name, breaks=x_breaks, label=x_labs, limits=x_lims) +
  scale_y_continuous(name=y_name, breaks=y_breaks, labels=NULL, limits=y_lims,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  scale_shape_manual(name="Legend", labels=leg_shape_labs, values=leg_shape_cols, breaks=leg_shape_sort)
  scale_fill_manual(name="", labels=leg_fill_labs, values=leg_fill_cols, breaks=leg_fill_sort) +
  scale_colour_manual(name="", labels=leg_col_labs, values=leg_col_cols, breaks=leg_col_sort) +
  guides(shape=guide_legend(order=1), fill=guide_legend(order=2), colour=guide_legend(order=3)) +
```

```
  theme(plot.title=element_text(hjust=0.5, size=13.5), plot.subtitle=element_text(hjust=0.5),
        axis.text.x=element_text(angle=0, vjust=1),
        legend.key.width=unit(0.8,"cm"), legend.position="bottom")
plot(X_PP_boot_plot)
```

```
## Warning: The following aesthetics were dropped during statistical transformation: sample
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a `group` aesthetic or to convert a numerical
##   variable into a factor?

## Warning: Removed 1 rows containing missing values (`geom_smooth()`).
```



From the inspection of both the *Q–Q* and *P–P* plots for the **x** data set we have visual evidence that the data set has been drawn from a Gaussian distribution. Note also that the higer slope of the interquatile and regression line with restect to the $y = x$ line corresponds to the circumstance that the empirical variance of the data set **x** is estimated larger than 1, while the pattern of the scatter plot which is nor *S-shaped* neither *reverse S-shaped* corresponds to the circumstance that the empirical excess of kurtosis of the data sets **x** has been estimated close to 0.

We now consider the *Q–Q* and *P–P* plots for the Y data set.

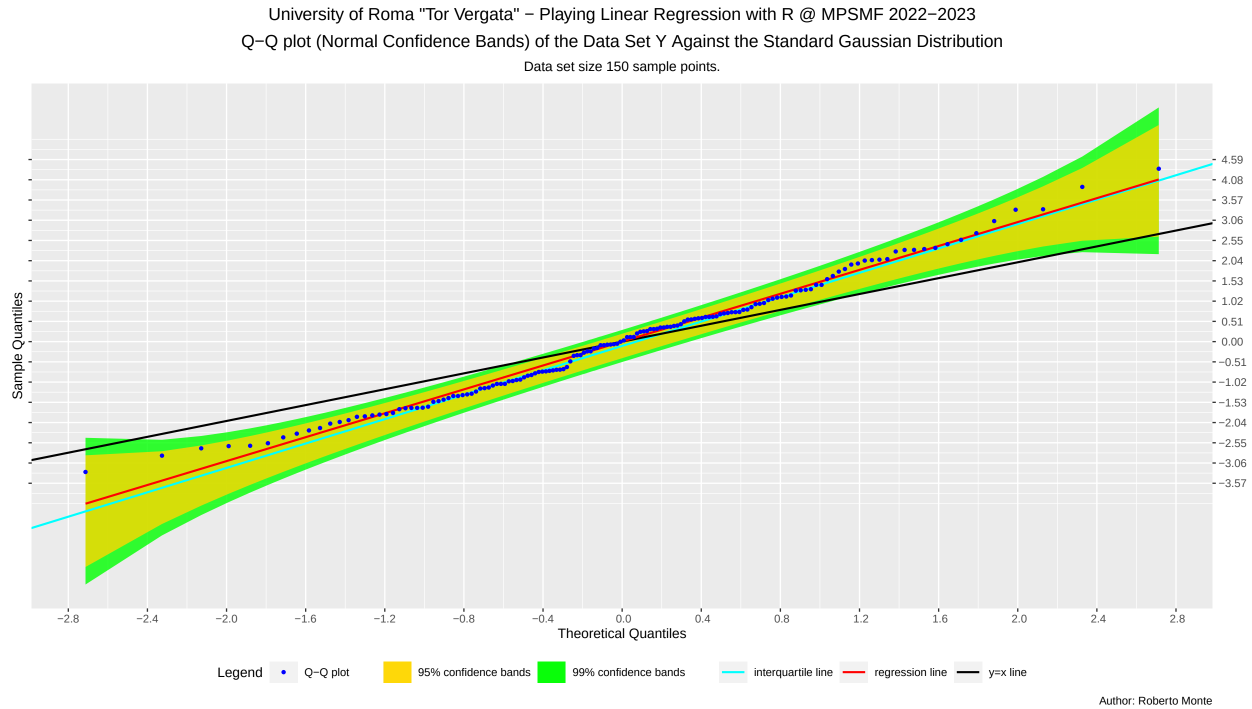
```
z <- Gauss_df$Y                                         # Y data set.
z_cent <- z - mean(z)                             # Centered Y data set.
z_cent_qemp <- qemp(ppoints(length(z)), z_cent) # Empirical quantiles of the centered X data set.
Gauss_quants <- qnorm(ppoints(length(z)), mean=0, sd=1) # Quantiles of the standard Gaussian distributi
                                                        # corresponding to the empirical quantiles
Y_QQ_plot_df <- data.frame(k=1:length(z), S=z_cent, X=Gauss_quants, Y=z_cent_qemp)
head(Y_QQ_plot_df)
```

```
##   k           S         X         Y
## 1 1  0.74351166 -2.713052 -3.286061
## 2 2 -0.08906686 -2.326348 -2.924951
## 3 3  0.51142779 -2.128045 -2.712211
## 4 4  0.97339073 -1.989313 -2.640805
## 5 5 -2.68990031 -1.880794 -2.626663
## 6 6  4.35758439 -1.790751 -2.568628
```

```
Data_df <- Y_QQ_plot_df
n <- nrow(Data_df)
quart_probs <- c(0.25,0.75)
quart_Y <- as.vector(quantile(Data_df$Y, quart_probs))
quart_X <- qnorm(quart_probs, mean=0, sd=1)
slope <- diff(quart_Y)/diff(quart_X)
intercept <- quart_Y[1]-slope*quart_X[1]
distr <- "norm"
distr_pars <- list(mean=0, sd=1)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u00
paste("Q-Q plot (Normal Confidence Bands) of the Data Set Y Against the Standard Gaussian Distribution")
subtitle_content <- bquote(paste("Data set size ", .(n), " sample points."))
caption_content <- "Author: Roberto Monte"
x_name <- bquote("Theoretical Quantiles")
y_name <- bquote("Sample Quantiles")
x_breaks_num <- 15 # (deduced from primeFactors(n))
# x_breaks_num <- ceiling(n^(1/2)) # Tukey & Mosteller square-root rule
# x_breaks_num <- ceiling(1+log2(n)) # Sturges rule
# x_breaks_num <- ceiling((2*n)^(1/3)) # Teller & Scott rice rule
x_binwidth <- round((max(Data_df$X)-min(Data_df$X))/x_breaks_num, digits=1)
x_breaks_low <- floor((min(Data_df$X)/x_binwidth))*x_binwidth
x_breaks_up <- ceiling((max(Data_df$X)/x_binwidth))*x_binwidth
x_breaks <- c(seq(from=x_breaks_low, to=x_breaks_up, by=x_binwidth))
x_labs <- format(x_breaks, scientific=FALSE)
J <- 1.0
x_lims <- c((x_breaks_low-J*x_binwidth), (x_breaks_up+J*x_binwidth))
y_breaks_num <- length(x_breaks)
y_binwidth <- round((max(Data_df$Y)-min(Data_df$Y))/y_breaks_num, digits=3)
y_breaks_low <- floor((min(Data_df$Y)/y_binwidth))*y_binwidth
y_breaks_up <- ceiling((max(Data_df$Y)/y_binwidth))*y_binwidth
y_breaks <- c(round(seq(from=y_breaks_low, to=y_breaks_up, by=y_binwidth),3))
y_labs <- format(y_breaks, scientific=FALSE)
K <- 1.5
y_lims <- c((y_breaks_low-K*y_binwidth), (y_breaks_up+K*y_binwidth))
y1_shape <- bquote("Q-Q plot")
y1_fill <- bquote("95% confidence bands")
y2_fill <- bquote("99% confidence bands")
col_1 <- bquote("interquartile line")
col_2 <- bquote("regression line")
col_3 <- bquote("y=x line")
leg_shape_labs <- y1_shape
leg_fill_labs <- c(y1_fill, y2_fill)
leg_col_labs <- c(col_1, col_2, col_3)
leg_shape_cols <- c("y1_shape" = 19)
leg_fill_cols <- c("y1_fill"="gold", "y2_fill"="green")
leg_col_cols <- c("col_1"="cyan", "col_2"="red", "col_3"="black")
```

```
leg_shape_sort <- "y1_shape"
leg_fill_sort <- c("y1_fill", "y2_fill")
leg_col_sort <- c("col_1", "col_2", "col_3")
Y_QQ_norm_plot <- ggplot(Data_df, aes(sample=S)) +
  stat_qq_band(aes(fill="y2_fill"), distribution=distr, dparams=distr_pars, conf = 0.99, bandType = "po
  stat_qq_band(aes(fill="y1_fill"), distribution=distr, dparams=distr_pars, conf = 0.95, bandType = "po
#  stat_qq_line(aes(colour="col_1"), distribution=distr, dparams=distr_pars) +
  geom_abline(aes(slope=slope, intercept=intercept, colour="col_1"), size=0.8, linetype="solid", show.le
#  geom_segment(aes(x=X[1], xend=-X[1], y=X[1], yend=-X[1], colour="col_3"),
#               size=0.8, linetype="solid", show.legend=FALSE) +
  geom_abline(aes(slope=1, intercept=0, colour="col_3"), size=0.8, linetype="solid", show.legend=FALSE)
  stat_smooth(alpha=1, size=0.8, linetype="solid", aes(x=X, y=Y, colour="col_2"),
              method="lm" , formula=y~x, se=FALSE, fullrange=FALSE) +
  stat_qq_point(aes(shape="y1_shape"), colour="blue", alpha=1, size=1.0,
                distribution=distr, dparams=distr_pars) +
  scale_x_continuous(name=x_name, breaks=x_breaks, label=x_labs, limits=NULL) +
  scale_y_continuous(name=y_name, breaks=y_breaks, labels=NULL, limits=NULL,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  scale_shape_manual(name="Legend", labels=leg_shape_labs, values=leg_shape_cols, breaks=leg_shape_sort
  scale_fill_manual(name="", labels=leg_fill_labs, values=leg_fill_cols, breaks=leg_fill_sort) +
  scale_colour_manual(name="", labels=leg_col_labs, values=leg_col_cols, breaks=leg_col_sort) +
  guides(shape=guide_legend(order=1), fill=guide_legend(order=2), colour=guide_legend(order=3)) +
  theme(plot.title=element_text(hjust=0.5, size=13.5), plot.subtitle=element_text(hjust=0.5),
        axis.text.x=element_text(angle=0, vjust=1),
        legend.key.width=unit(0.8,"cm"), legend.position="bottom")
plot(Y_QQ_norm_plot)
```

```
## Warning: The following aesthetics were dropped during statistical transformation: sample
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a `group` aesthetic or to convert a numerical
##   variable into a factor?
```



We draw also the P–P plot with bootstrap confidence bands.

```
z <- Gauss_df$Y_st                                          # Standardized Y_st data set.
z_qemp <- qemp(ppoints(length(z)), z) # Empirical quantiles of the Y_st data set.
z_pemp <- pemp(z_qemp, z)            # Empirical probabilities of the Y_st data set
                                                            # corresponding to the empirical quantiles.
Gauss_quants <- qnorm(ppoints(length(z)), mean=0, sd=1) # Quantiles of the standard Gaussian distributi
                                                            # corresponding to the empirical quantiles.
Gauss_probs <- pnorm(Gauss_quants, mean=0, sd=1)# Probabilities of the standard Gaussian distribution
                                                            # corresponding to the empirical probabilities.
Y_PP_plot_df <- data.frame(k=1:length(z), S=z, X=Gauss_probs, Y=z_pemp)
head(Y_PP_plot_df)
```

```
##   k           S           X           Y
## 1 1  0.49368381 0.003333333 0.004159734
## 2 2 -0.05913944 0.010000000 0.010000000
## 3 3  0.33958260 0.016666667 0.016666667
## 4 4  0.64632107 0.023333333 0.023333333
## 5 5 -1.78606513 0.030000000 0.030000000
## 6 6  2.89338959 0.036666667 0.036666667
```

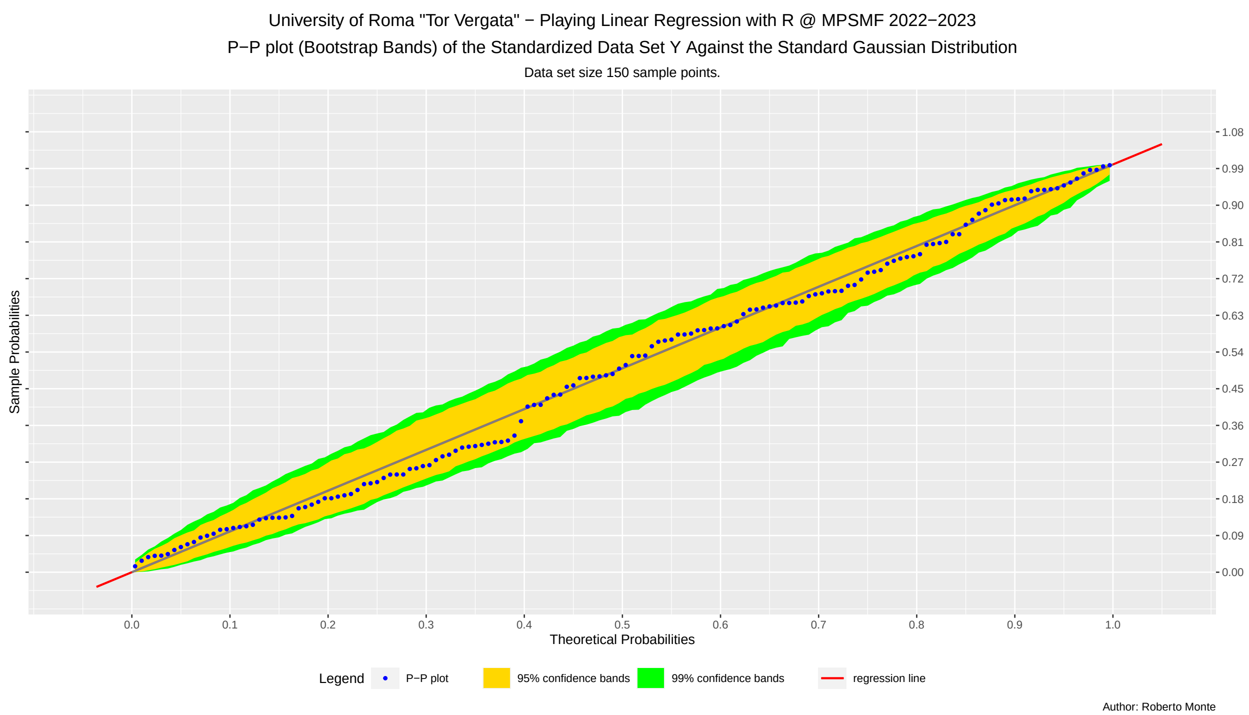
Second we draw the P–P plot of the residuals with boot bands (the only option available).

```
Data_df <- Y_PP_plot_df
n <- nrow(Data_df)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u0
paste("P-P plot (Bootstrap Bands) of the Standardized Data Set Y Against the Standard Gaussian Distribu
subtitle_content <- bquote(paste("Data set size ", .(n), " sample points."))
caption_content <- "Author: Roberto Monte"
x_name <- bquote("Theoretical Probabilities")
y_name <- bquote("Sample Probabilities")
```

```
x_breaks_num <- 15 # (deduced from primeFactors(n))
# x_breaks_num <- ceiling(n^(1/2)) # Tukey & Mosteller square-root rule
# x_breaks_num <- ceiling(1+log2(n)) # Sturges rule
# x_breaks_num <- ceiling((2*n)^(1/3)) # Teller & Scott rice rule
x_binwidth <- round((max(Data_df$X)-min(Data_df$X))/x_breaks_num, digits=1)
x_breaks_low <- floor((min(Data_df$X)/x_binwidth))*x_binwidth
x_breaks_up <- ceiling((max(Data_df$X)/x_binwidth))*x_binwidth
x_breaks <- c(seq(from=x_breaks_low, to=x_breaks_up, by=x_binwidth))
# x_breaks <- c(1,round(seq(from=x_breaks_low, to=x_breaks_up, by=x_binwidth),3),n)
x_labs <- format(x_breaks, scientific=FALSE)
J <- 0.5
x_lims <- c(x_breaks_low-J*x_binwidth, x_breaks_up+J*x_binwidth)
y_breaks_num <- length(x_breaks)
y_binwidth <- round((max(Data_df$Y)-min(Data_df$Y))/y_breaks_num, digits=3)
y_breaks_low <- floor((min(Data_df$Y)/y_binwidth))*y_binwidth
y_breaks_up <- ceiling((max(Data_df$Y)/y_binwidth))*y_binwidth
y_breaks <- c(round(seq(from=y_breaks_low, to=y_breaks_up, by=y_binwidth),3))
y_labs <- format(y_breaks, scientific=FALSE)
K <- 0.5
y_lims <- c(y_breaks_low-K*y_binwidth, y_breaks_up+K*y_binwidth)
y1_shape <- bquote("P-P plot")
y1_fill <- bquote("95% confidence bands")
y2_fill <- bquote("99% confidence bands")
col_1 <- bquote("y=x line")
col_2 <- bquote("regression line")
leg_shape_labs <- y1_shape
leg_fill_labs <- c(y1_fill, y2_fill)
leg_col_labs <- c(col_1, col_2)
leg_shape_cols <- c("y1_shape" = 19)
leg_fill_cols <- c("y1_fill"="gold", "y2_fill"="green")
leg_col_cols <- c("col_1"="black", "col_2"="red")
leg_shape_sort <- "y1_shape"
leg_fill_sort <- c("y1_fill", "y2_fill")
leg_col_sort <- c("col_1", "col_2")
distr <- "norm"
distr_pars <- list(mean=0, sd=1)
Y_PP_boot_plot <- ggplot(Data_df, aes(sample=S)) +
  stat_pp_band(aes(fill="y2_fill"), distribution=distr, dparams=distr_pars, conf = 0.99, bandType = "boo
  stat_pp_band(aes(fill="y1_fill"), distribution=distr, dparams=distr_pars, conf = 0.95, bandType = "boo
  stat_smooth(alpha=1, size=0.8, linetype="solid", aes(x=X, y=Y, colour="col_2"),
              method="lm" , formula=y~x, se=FALSE, fullrange=TRUE) +
  stat_pp_line(aes(colour="col_1")) +
  stat_pp_point(aes(shape="y1_shape"), colour="blue", alpha=1, size=1.0,
                distribution=distr, dparams=distr_pars) +
  scale_x_continuous(name=x_name, breaks=x_breaks, label=x_labs, limits=x_lims) +
  scale_y_continuous(name=y_name, breaks=y_breaks, labels=NULL, limits=y_lims,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  scale_shape_manual(name="Legend", labels=leg_shape_labs, values=leg_shape_cols, breaks=leg_shape_sort)
  scale_fill_manual(name="", labels=leg_fill_labs, values=leg_fill_cols, breaks=leg_fill_sort) +
  scale_colour_manual(name="", labels=leg_col_labs, values=leg_col_cols, breaks=leg_col_sort) +
  guides(shape=guide_legend(order=1), fill=guide_legend(order=2), colour=guide_legend(order=3)) +
```

```
  theme(plot.title=element_text(hjust=0.5, size=13.5), plot.subtitle=element_text(hjust=0.5),
        axis.text.x=element_text(angle=0, vjust=1),
        legend.key.width=unit(0.8,"cm"), legend.position="bottom")
plot(Y_PP_boot_plot)
```

```
## Warning: The following aesthetics were dropped during statistical transformation: sample
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a `group` aesthetic or to convert a numerical
##   variable into a factor?

## Warning: Removed 1 rows containing missing values (`geom_smooth()`).
```



From the inspection of both the *Q-Q* and *P-P* plots for the $\mathbf{y}$ data set we have visual evidence that the data set has been drawn from a Gaussian distribution. Note also that the higher slope of the interquatile and regression line with respect to the $y = x$ line corresponds to the circumstance that the empirical variance of the data set $\mathbf{y}$ is estimated to be larger than 1, while the pattern of the scatter plot which is nor *S-shaped* neither *reverse S-shaped* corresponds to the circumstance that the empirical excess of kurtosis of the data sets $\mathbf{y}$ is estimated to be close to 0.

We plot the relative frequency and the density histograms of the standardized data sets $\mathbf{x}$ and $\mathbf{y}$,

First, we build a table to compare the value taken by standard statistics on the X_st data set with theoretical values.

Standard statistics on X_st data set.

```
mode <- function(x) {
  d <- density(x)
  d$x[which.max(d$y)]
}
```

```r
Samp_Data <-  Gauss_df$X_st
Statistics=c("mean", "median", "mode", "min. (99.73%)", "max. (99.73%)", "1st quart.", "3rd quart.", "s

Teor_Stats <- rep(0,10)
Teor_Stats[1] <- as.character(formatC(0, digits=3, format="f"))
Teor_Stats[2] <- as.character(formatC(qnorm(0.50, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE), di
Teor_Stats[3] <- as.character(formatC(0, digits=3, format="f"))
Teor_Stats[4] <- as.character(formatC(qnorm(0.00135, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE),
Teor_Stats[5] <- as.character(formatC(qnorm(0.99865, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE),
Teor_Stats[6] <- as.character(formatC(qnorm(0.25, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE), di
Teor_Stats[7] <- as.character(formatC(qnorm(0.75, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE), di
Teor_Stats[8] <- as.character(formatC(1, digits=3, format="f"))
Teor_Stats[9] <- as.character(formatC(0, digits=3, format="f"))
Teor_Stats[10] <- as.character(formatC(0, digits=3, format="f"))

Samp_Stats <- rep(0,10)
Samp_Stats[1] <- as.character(formatC(mean(Samp_Data), digits=3, format="f"))
Samp_Stats[2] <- as.character(formatC(median(Samp_Data), digits=3, format="f"))
Samp_Stats[3] <- as.character(formatC(mode(Samp_Data), digits=3, format="f"))
Samp_Stats[4] <- as.character(formatC(min(Samp_Data), digits=3, format="f"))
Samp_Stats[5] <- as.character(formatC(max(Samp_Data), digits=3, format="f"))
Samp_Stats[6] <- as.character(formatC(quantile(Samp_Data,0.25), digits=3, format="f"))
Samp_Stats[7] <- as.character(formatC(quantile(Samp_Data,0.75), digits=3, format="f"))
Samp_Stats[8] <- as.character(formatC(sd(Samp_Data), digits=3, format="f"))
Samp_Stats[9] <- as.character(formatC(as.numeric(timeDate::skewness(Samp_Data, method="moment")), digit
Samp_Stats[10] <- as.character(formatC(as.numeric(timeDate::kurtosis(Samp_Data, method="excess")), digi

Table_Stats <- data.frame(Samp_Stats,Teor_Stats)
rownames(Table_Stats) <- Statistics
colnames(Table_Stats) <- c("Samp. Stats", "Teor. Stats")
```

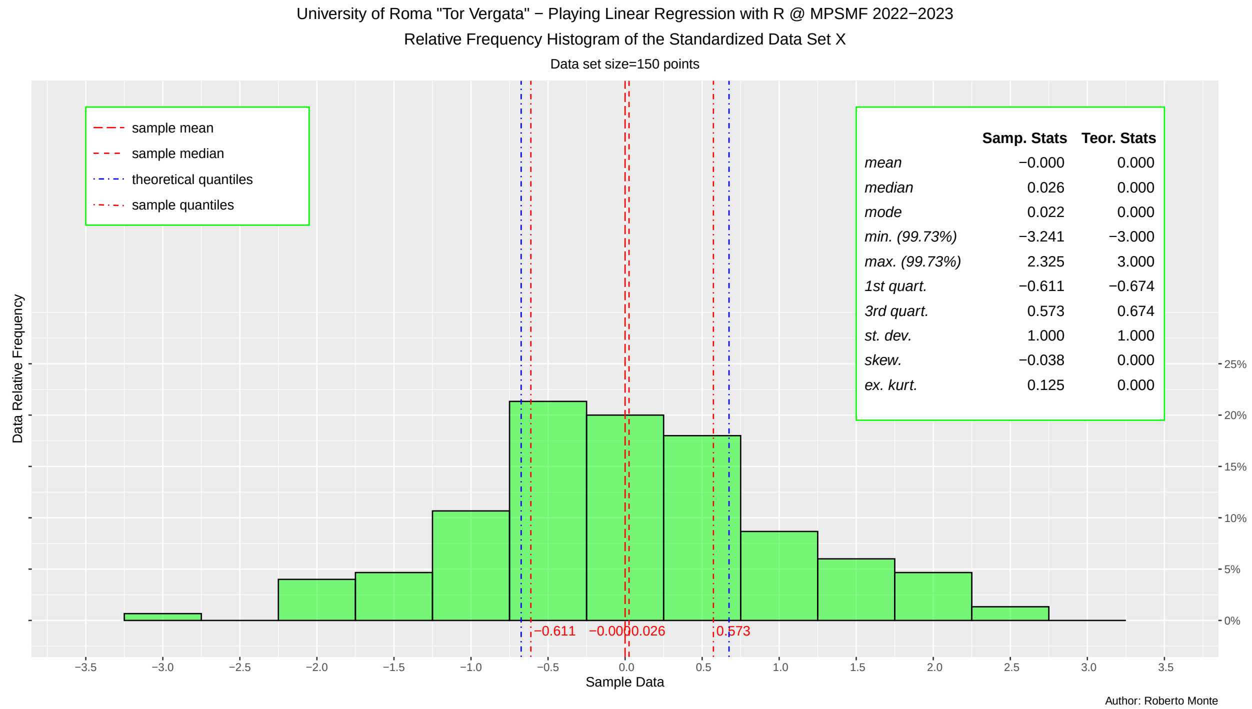
Then, we plot relative the frequency and density histograms of the standardized data set **x**.

The relative frequency histograms.

```r
# library(gridExtra)
#### Relative Frequency Histogram + Sample Statistics
Data_df <- Gauss_df
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u0
subtitle_content <- bquote(paste("Data set size=", .(n), " points"))
caption_content <- "Author: Roberto Monte"
x_binwidth <- 0.5
x_breaks <- seq(from=-3.5, to=3.5, by=x_binwidth)
x_labs <- format(x_breaks, scientific=FALSE)
x_lims <- c(-3.5,3.5)
y_breaks <- seq(from=0, to=0.25, by=0.05)
y_labs <- format(percent(y_breaks), scientific=FALSE)
y_lims <- c(-0.010,0.50)
tt3 <- ttheme_minimal(core=list(fg_params=list(hjust=1, x=0.90)),
                      rowhead=list(fg_params=list(hjust=0, x=0)))
Table_Stats_Grob <- tableGrob(Table_Stats, theme=tt3)
X_st_rel_freq_hist <- ggplot(Data_df, aes(x=X_st)) +
  geom_histogram(binwidth=x_binwidth , aes(y=stat(count)/sum(count)), color="black", fill="green", alph
  scale_x_continuous(name="Sample Data", breaks=x_breaks, labels=x_labs, limits=x_lims) +
```

```
  scale_y_continuous(name="Data Relative Frequency", breaks=y_breaks, labels=NULL, limits=y_lims,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  theme(plot.title=element_text(hjust=0.5),
        plot.subtitle=element_text(hjust=0.5),
        plot.caption=element_text(hjust=1.0)) +
  geom_vline(aes(xintercept=as.numeric(quantile(Samp_Data,0.25))),
             colour="red", linetype="dotdash", size=0.5) +
  annotate("text", x=as.numeric(quantile(Samp_Data,0.25))+0.020, y=-0.01, colour="red",
           label=Samp_Stats[6], hjust=0) +
  geom_vline(aes(xintercept=as.numeric(quantile(Samp_Data,0.75))),
             colour="red", linetype="dotdash", size=0.5) +
  annotate("text", x=as.numeric(quantile(Samp_Data,0.75))+0.020, y=-0.01, colour="red",
           label=Samp_Stats[7], hjust=0) +
  geom_vline(aes(xintercept=mean(Samp_Data)), colour="red", linetype="longdash", size=0.5) +
  annotate("text", x=mean(Samp_Data)-0.235, y=-0.01, colour="red",
           label=Samp_Stats[1], hjust=0) +
  geom_vline(aes(xintercept=median(Samp_Data)), colour="red", linetype="dashed", size=0.5) +
  annotate("text", x=median(Samp_Data)+0.015, y=-0.01, colour="red",
           label=Samp_Stats[2], hjust=0) +
  geom_vline(aes(xintercept=qnorm(0.25, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE)),
             colour="blue", linetype="dotdash", size=0.5) +
  geom_vline(aes(xintercept=qnorm(0.75, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE)),
             colour="blue", linetype="dotdash", size=0.5) +
  annotate("rect", xmin=1.50, xmax=3.50, ymin=0.195, ymax=0.5, colour="green", fill="white") +
  annotation_custom(Table_Stats_Grob, xmin=1.75, xmax=3.30, ymin=0.3, ymax=0.4) +
  annotate("rect", xmin=-3.50, xmax=-2.05, ymin=0.385, ymax=0.500, colour="green", fill="white") +
  annotate("segment", x=-3.45, xend=-3.25, y=0.480, yend=0.480, colour="red", lty="longdash") +
  annotate("text", x=-3.20, y=0.480, colour="black", label="sample mean", hjust=0) +
  annotate("segment", x=-3.45, xend=-3.25, y=0.455, yend=0.455, colour="red", lty="dashed") +
  annotate("text", x=-3.20, y=0.455, colour="black", label="sample median", hjust=0) +
  annotate("segment", x=-3.45, xend=-3.25, y=0.430, yend=0.430, colour="blue", lty="dotdash") +
  annotate("text", x=-3.20, y=0.430, colour="black", label="theoretical quantiles", hjust=0) +
  annotate("segment", x=-3.45, xend=-3.25, y=0.405, yend=0.404, colour="red", lty="dotdash") +
  annotate("text", x=-3.20, y=0.405, colour="black", label="sample quantiles", hjust=0)
plot(X_st_rel_freq_hist)
```

```
## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```

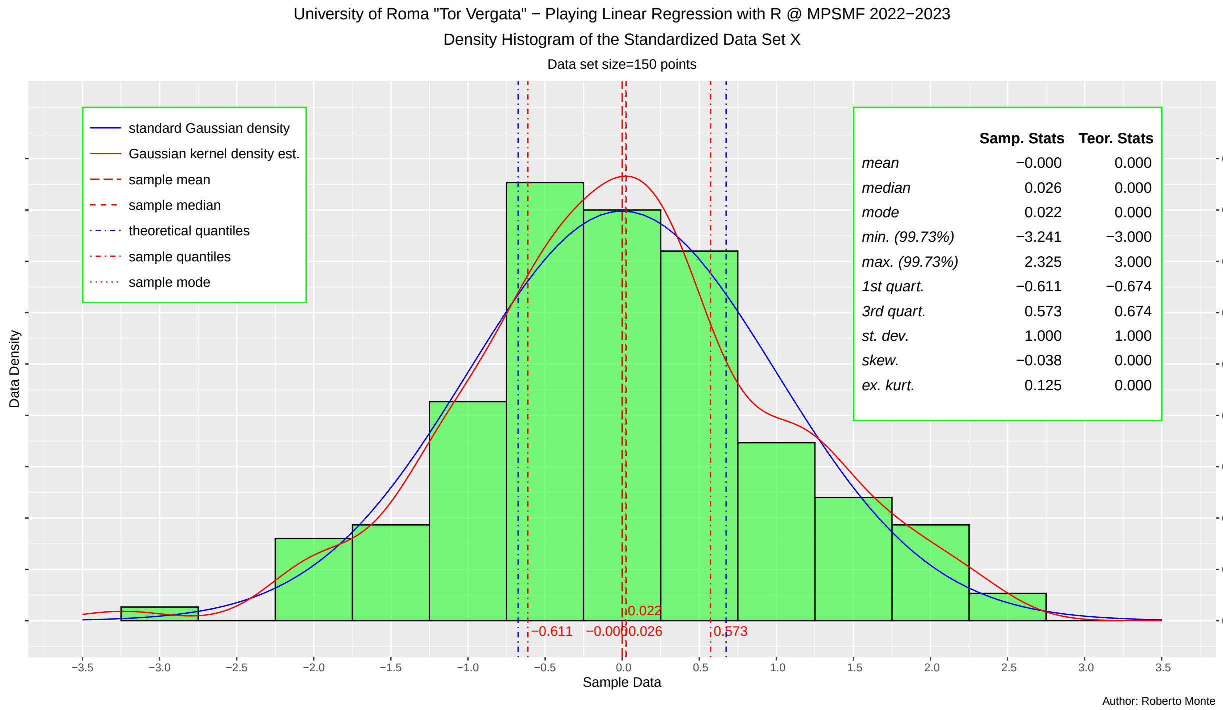


The density histograms.

```
# library(gridExtra)
#### Density Histogram + Sample Statistics + Density Kernel Estimation
Data_df <- Gauss_df
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u0
subtitle_content <- bquote(paste("Data set size=", .(n), " points"))
caption_content <- "Author: Roberto Monte"
x_binwidth <- 0.5
x_breaks <- seq(from=-3.5, to=3.5, by=x_binwidth)
x_labs <- format(x_breaks, scientific=FALSE)
x_lims <- c(-3.5,3.5)
y_breaks <- seq(from=0, to=0.45, by=0.05)
y_labs <- format(y_breaks, scientific=FALSE)
y_lims <- c(-0.010,0.50)
tt3 <- ttheme_minimal(core=list(fg_params=list(hjust=1, x=0.90)),
                      rowhead=list(fg_params=list(hjust=0, x=0)))
Table_Stats_Grob <- tableGrob(Table_Stats, theme=tt3)
X_st_dens_hist <- ggplot(Data_df, aes(x=X_st)) +
  geom_histogram(binwidth=x_binwidth, aes(y=..density..), # binwidth=0.5,  # Density Histogram
                 color="black", fill="green", alpha=0.5) +
  scale_x_continuous(name="Sample Data", breaks=x_breaks, labels=x_labs, limits=x_lims) +
  scale_y_continuous(name="Data Density", breaks=y_breaks, labels=NULL, limits=y_lims,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  theme(plot.title=element_text(lineheight=0.6, face="bold", hjust=0.5),
        plot.subtitle=element_text(hjust= 0.5),
        plot.caption=element_text(hjust=1.0)) +
  stat_function(fun=dnorm, colour="blue", args=list(mean=0, sd=1)) +
  geom_vline(aes(xintercept=as.numeric(quantile(Samp_Data,0.25))),
```

```
              colour="red", linetype="dotdash", size=0.5) +
  annotate("text", x=as.numeric(quantile(Samp_Data,0.25))+0.020, y=-0.01, colour="red",
            label=Samp_Stats[6], hjust=0) +
  geom_vline(aes(xintercept=as.numeric(quantile(Samp_Data,0.75))),
              colour="red", linetype="dotdash", size=0.5) +
  annotate("text", x=as.numeric(quantile(Samp_Data,0.75))+0.020, y=-0.01, colour="red",
            label=Samp_Stats[7], hjust=0) +
  geom_vline(aes(xintercept=mean(Samp_Data)), colour="red", linetype="longdash", size=0.5) +
  annotate("text", x=mean(Samp_Data)-0.235, y=-0.01, colour="red",
            label=Samp_Stats[1], hjust=0) +
  geom_vline(aes(xintercept=median(Samp_Data)), colour="red", linetype="dashed", size=0.5) +
  annotate("text", x=median(Samp_Data)+0.015, y=-0.01, colour="red",
            label=Samp_Stats[2], hjust=0) +
  geom_vline(aes(xintercept=mode(Samp_Data)), colour="red", linetype="dotted", size=0.5) +
  annotate("text", x=mode(Samp_Data)+0.015, y=0.01, colour="red",
            label=Samp_Stats[3], hjust=0) +
  geom_vline(aes(xintercept=qnorm(0.25, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE)),
              colour="blue", linetype="dotdash", size=0.5) +
  geom_vline(aes(xintercept=qnorm(0.75, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE)),
              colour="blue", linetype="dotdash", size=0.5) +
  geom_density(alpha=.2, colour="red") +
  annotate("rect", xmin=1.50, xmax=3.50, ymin=0.195, ymax=0.5, colour="green", fill="white") +
  annotation_custom(Table_Stats_Grob, xmin=1.75, xmax=3.30, ymin=0.3, ymax=0.4) +
  annotate("rect", xmin=-3.50, xmax=-2.05, ymin=0.310, ymax=0.500, colour="green", fill="white") +
  annotate("segment", x= -3.45, xend=-3.25, y=0.480, yend=0.480, colour="blue") +
  annotate("text", x=-3.20, y=0.480, colour="black", label="standard Gaussian density", hjust=0) +
  annotate("segment", x= -3.45, xend=-3.25, y=0.455, yend=0.455, colour="red") +
  annotate("text", x=-3.20, y=0.455, colour="black", label="Gaussian kernel density est.", hjust=0) +
  annotate("segment", x= -3.45, xend=-3.25, y=0.430, yend=0.430, colour="red", lty="longdash") +
  annotate("text", x=-3.20, y=0.430, colour="black", label="sample mean", hjust=0) +
  annotate("segment", x= -3.45, xend=-3.25, y=0.405, yend=0.405, colour="red", lty="dashed") +
  annotate("text", x=-3.20, y=0.405, colour="black", label="sample median", hjust=0) +
  annotate("segment", x= -3.45, xend=-3.25, y=0.380, yend=0.380, colour="blue", lty="dotdash") +
  annotate("text", x=-3.20, y=0.380, colour="black", label="theoretical quantiles", hjust=0) +
  annotate("segment", x= -3.45, xend=-3.25, y=0.355, yend=0.355, colour="red", lty="dotdash") +
  annotate("text", x=-3.20, y=0.355, colour="black", label="sample quantiles", hjust=0) +
  annotate("segment", x= -3.45, xend=-3.25, y=0.330, yend=0.330, colour="red", lty="dotted") +
  annotate("text", x=-3.20, y=0.330, colour="black", label="sample mode", hjust=0)
plot(X_st_dens_hist)
```

```
## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```



We also plot frequency and density histograms of the standardized data set $\mathbf{y}^*$.

Standard statistics on Y_st data set.

```
mode <- function(x) {
  d <- density(x)
  d$x[which.max(d$y)]
}
Samp_Data <-  Gauss_df$Y_st
Statistics=c("mean", "median", "mode", "min. (99.73%)", "max. (99.73%)", "1st quart.", "3rd quart.", "s

Teor_Stats <- rep(0,10)
Teor_Stats[1] <- as.character(formatC(0, digits=3, format="f"))
Teor_Stats[2] <- as.character(formatC(qnorm(0.50, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE), dig
Teor_Stats[3] <- as.character(formatC(0, digits=3, format="f"))
Teor_Stats[4] <- as.character(formatC(qnorm(0.00135, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE),
Teor_Stats[5] <- as.character(formatC(qnorm(0.99865, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE),
Teor_Stats[6] <- as.character(formatC(qnorm(0.25, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE), dig
Teor_Stats[7] <- as.character(formatC(qnorm(0.75, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE), dig
Teor_Stats[8] <- as.character(formatC(1, digits=3, format="f"))
Teor_Stats[9] <- as.character(formatC(0, digits=3, format="f"))
Teor_Stats[10] <- as.character(formatC(0, digits=3, format="f"))

Samp_Stats <- rep(0,10)
Samp_Stats[1] <- as.character(formatC(mean(Samp_Data), digits=3, format="f"))
Samp_Stats[2] <- as.character(formatC(median(Samp_Data), digits=3, format="f"))
Samp_Stats[3] <- as.character(formatC(mode(Samp_Data), digits=3, format="f"))
Samp_Stats[4] <- as.character(formatC(min(Samp_Data), digits=3, format="f"))
Samp_Stats[5] <- as.character(formatC(max(Samp_Data), digits=3, format="f"))
Samp_Stats[6] <- as.character(formatC(quantile(Samp_Data,0.25), digits=3, format="f"))
Samp_Stats[7] <- as.character(formatC(quantile(Samp_Data,0.75), digits=3, format="f"))
```

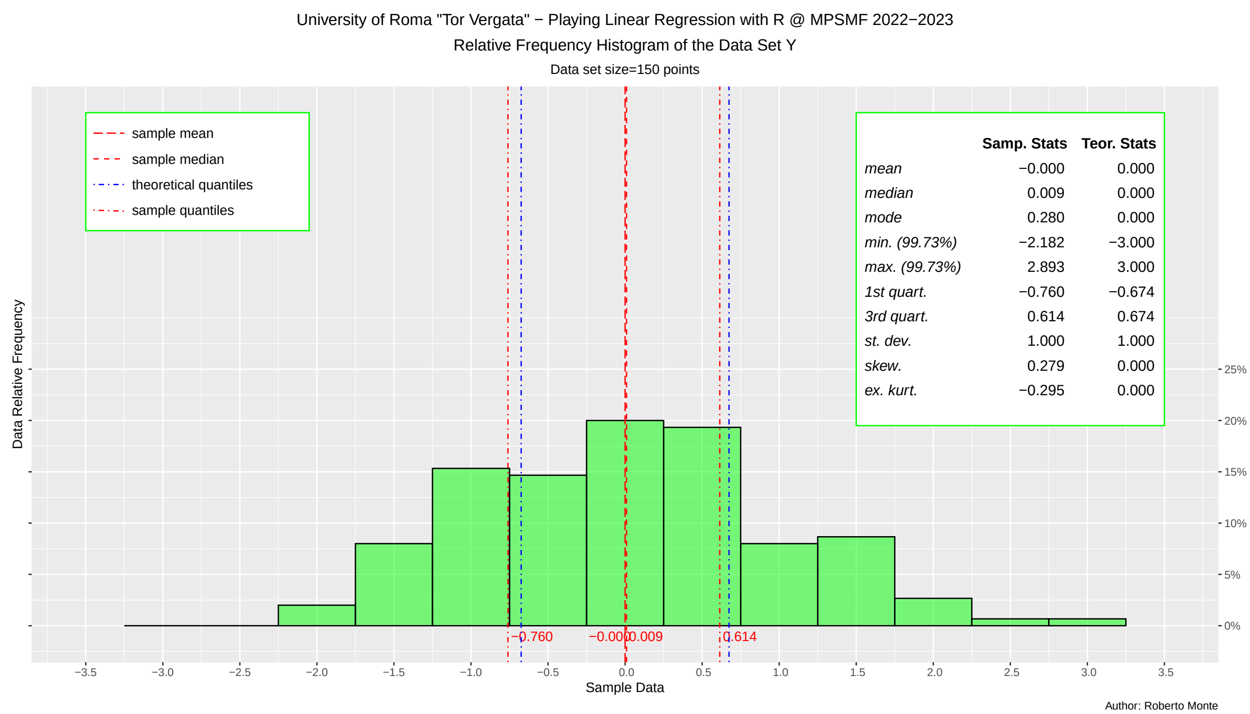
```
Samp_Stats[8] <- as.character(formatC(sd(Samp_Data), digits=3, format="f"))
Samp_Stats[9] <- as.character(formatC(as.numeric(timeDate::skewness(Samp_Data, method="moment")), digits
Samp_Stats[10] <- as.character(formatC(as.numeric(timeDate::kurtosis(Samp_Data, method="excess")), digi

Table_Stats <- data.frame(Samp_Stats,Teor_Stats)
rownames(Table_Stats) <- Statistics
colnames(Table_Stats) <- c("Samp. Stats", "Teor. Stats")
```

The relative frequency histograms

```
# library(gridExtra)
#### Relative Frequency Histogram + Sample Statistics
Data_df <- Gauss_df
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u0
subtitle_content <- bquote(paste("Data set size=", .(n), " points"))
caption_content <- "Author: Roberto Monte"
x_binwidth <- 0.5
x_breaks <- seq(from=-3.5, to=3.5, by=x_binwidth)
x_labs <- format(x_breaks, scientific=FALSE)
x_lims <- c(-3.5,3.5)
y_breaks <- seq(from=0, to=0.25, by=0.05)
y_labs <- format(percent(y_breaks), scientific=FALSE)
y_lims <- c(-0.010,0.50)
tt3 <- ttheme_minimal(core=list(fg_params=list(hjust=1, x=0.90)),
                      rowhead=list(fg_params=list(hjust=0, x=0)))
Table_Stats_Grob <- tableGrob(Table_Stats, theme=tt3)
Data_df_rel_freq_hist <- ggplot(Data_df, aes(x=Y_st)) +
  geom_histogram(binwidth=x_binwidth , aes(y=stat(count)/sum(count)), color="black", fill="green", alpha
  scale_x_continuous(name="Sample Data", breaks=x_breaks, labels=x_labs, limits=x_lims) +
  scale_y_continuous(name="Data Relative Frequency", breaks=y_breaks, labels=NULL, limits=y_lims,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  theme(plot.title=element_text(hjust=0.5),
        plot.subtitle=element_text(hjust=0.5),
        plot.caption=element_text(hjust=1.0)) +
  geom_vline(aes(xintercept=as.numeric(quantile(Samp_Data,0.25))),
             colour="red", linetype="dotdash", size=0.5) +
  annotate("text", x=as.numeric(quantile(Samp_Data,0.25))+0.020, y=-0.01, colour="red",
           label=Samp_Stats[6], hjust=0) +
  geom_vline(aes(xintercept=as.numeric(quantile(Samp_Data,0.75))),
             colour="red", linetype="dotdash", size=0.5) +
  annotate("text", x=as.numeric(quantile(Samp_Data,0.75))+0.020, y=-0.01, colour="red",
           label=Samp_Stats[7], hjust=0) +
  geom_vline(aes(xintercept=mean(Samp_Data)), colour="red", linetype="longdash", size=0.5) +
  annotate("text", x=mean(Samp_Data)-0.235, y=-0.01, colour="red",
           label=Samp_Stats[1], hjust=0) +
  geom_vline(aes(xintercept=median(Samp_Data)), colour="red", linetype="dashed", size=0.5) +
  annotate("text", x=median(Samp_Data)+0.015, y=-0.01, colour="red",
           label=Samp_Stats[2], hjust=0) +
  geom_vline(aes(xintercept=qnorm(0.25, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE)),
             colour="blue", linetype="dotdash", size=0.5) +
  geom_vline(aes(xintercept=qnorm(0.75, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE)),
```

```
              colour="blue", linetype="dotdash", size=0.5) +
  annotate("rect", xmin=1.50, xmax=3.50, ymin=0.195, ymax=0.5, colour="green", fill="white") +
  annotation_custom(Table_Stats_Grob, xmin=1.75, xmax=3.30, ymin=0.3, ymax=0.4) +
  annotate("rect", xmin=-3.50, xmax=-2.05, ymin=0.385, ymax=0.500, colour="green", fill="white") +
  annotate("segment", x=-3.45, xend=-3.25, y=0.480, yend=0.480, colour="red", lty="longdash") +
  annotate("text", x=-3.20, y=0.480, colour="black", label="sample mean", hjust=0) +
  annotate("segment", x=-3.45, xend=-3.25, y=0.455, yend=0.455, colour="red", lty="dashed") +
  annotate("text", x=-3.20, y=0.455, colour="black", label="sample median", hjust=0) +
  annotate("segment", x=-3.45, xend=-3.25, y=0.430, yend=0.430, colour="blue", lty="dotdash") +
  annotate("text", x=-3.20, y=0.430, colour="black", label="theoretical quantiles", hjust=0) +
  annotate("segment", x=-3.45, xend=-3.25, y=0.405, yend=0.404, colour="red", lty="dotdash") +
  annotate("text", x=-3.20, y=0.405, colour="black", label="sample quantiles", hjust=0)
plot(Data_df_rel_freq_hist)
```

```
## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```



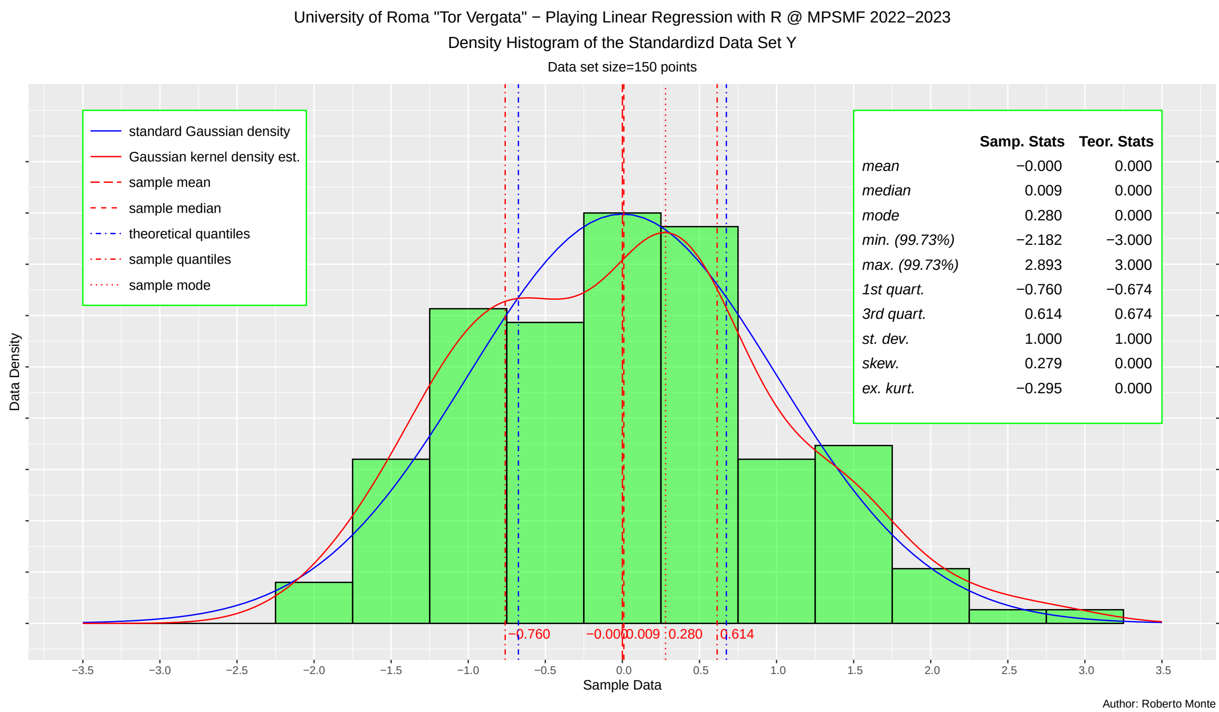
The density histograms

```
# library(gridExtra)
#### Density Histogram + Sample Statistics + Density Kernel Estimation
Data_df <- Gauss_df
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u0
subtitle_content <- bquote(paste("Data set size=", .(n), " points"))
caption_content <- "Author: Roberto Monte"
x_binwidth <- 0.5
x_breaks <- seq(from=-3.5, to=3.5, by=x_binwidth)
x_labs <- format(x_breaks, scientific=FALSE)
x_lims <- c(-3.5,3.5)
y_breaks <- seq(from=0, to=0.45, by=0.05)
y_labs <- format(y_breaks, scientific=FALSE)
```

```
y_lims <- c(-0.010,0.50)
tt3 <- ttheme_minimal(core=list(fg_params=list(hjust=1, x=0.90)),
                      rowhead=list(fg_params=list(hjust=0, x=0)))
Table_Stats_Grob <- tableGrob(Table_Stats, theme=tt3)
Y_st_dens_hist <- ggplot(Data_df, aes(x=Y_st)) +
  geom_histogram(binwidth=x_binwidth, aes(y=..density..), # binwidth=0.5,  # Density Histogram
                 color="black", fill="green", alpha=0.5) +
  scale_x_continuous(name="Sample Data", breaks=x_breaks, labels=x_labs, limits=x_lims) +
  scale_y_continuous(name="Data Density", breaks=y_breaks, labels=NULL, limits=y_lims,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  theme(plot.title=element_text(lineheight=0.6, face="bold", hjust=0.5),
        plot.subtitle=element_text(hjust= 0.5),
        plot.caption=element_text(hjust=1.0)) +
  stat_function(fun=dnorm, colour="blue", args=list(mean=0, sd=1)) +
  geom_vline(aes(xintercept=as.numeric(quantile(Samp_Data,0.25))),
             colour="red", linetype="dotdash", size=0.5) +
  annotate("text", x=as.numeric(quantile(Samp_Data,0.25))+0.020, y=-0.01, colour="red",
           label=Samp_Stats[6], hjust=0) +
  geom_vline(aes(xintercept=as.numeric(quantile(Samp_Data,0.75))),
             colour="red", linetype="dotdash", size=0.5) +
  annotate("text", x=as.numeric(quantile(Samp_Data,0.75))+0.020, y=-0.01, colour="red",
           label=Samp_Stats[7], hjust=0) +
  geom_vline(aes(xintercept=mean(Samp_Data)), colour="red", linetype="longdash", size=0.5) +
  annotate("text", x=mean(Samp_Data)-0.235, y=-0.01, colour="red",
           label=Samp_Stats[1], hjust=0) +
  geom_vline(aes(xintercept=median(Samp_Data)), colour="red", linetype="dashed", size=0.5) +
  annotate("text", x=median(Samp_Data)+0.015, y=-0.01, colour="red",
           label=Samp_Stats[2], hjust=0) +
  geom_vline(aes(xintercept=mode(Samp_Data)), colour="red", linetype="dotted", size=0.5) +
  annotate("text", x=mode(Samp_Data)+0.020, y=-0.01, colour="red",
           label=Samp_Stats[3], hjust=0) +
  geom_vline(aes(xintercept=qnorm(0.25, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE)),
             colour="blue", linetype="dotdash", size=0.5) +
  geom_vline(aes(xintercept=qnorm(0.75, mean=0.00, sd=1.00, lower.tail=TRUE, log.p=FALSE)),
             colour="blue", linetype="dotdash", size=0.5) +
  geom_density(alpha=.2, colour="red") +
  annotate("rect", xmin=1.50, xmax=3.50, ymin=0.195, ymax=0.5, colour="green", fill="white") +
  annotation_custom(Table_Stats_Grob, xmin=1.75, xmax=3.30, ymin=0.3, ymax=0.4) +
  annotate("rect", xmin=-3.50, xmax=-2.05, ymin=0.310, ymax=0.500, colour="green", fill="white") +
  annotate("segment", x= -3.45, xend=-3.25, y=0.480, yend=0.480, colour="blue") +
  annotate("text", x=-3.20, y=0.480, colour="black", label="standard Gaussian density", hjust=0) +
  annotate("segment", x= -3.45, xend=-3.25, y=0.455, yend=0.455, colour="red") +
  annotate("text", x=-3.20, y=0.455, colour="black", label="Gaussian kernel density est.", hjust=0) +
  annotate("segment", x= -3.45, xend=-3.25, y=0.430, yend=0.430, colour="red", lty="longdash") +
  annotate("text", x=-3.20, y=0.430, colour="black", label="sample mean", hjust=0) +
  annotate("segment", x= -3.45, xend=-3.25, y=0.405, yend=0.405, colour="red", lty="dashed") +
  annotate("text", x=-3.20, y=0.405, colour="black", label="sample median", hjust=0) +
  annotate("segment", x= -3.45, xend=-3.25, y=0.380, yend=0.380, colour="blue", lty="dotdash") +
  annotate("text", x=-3.20, y=0.380, colour="black", label="theoretical quantiles", hjust=0) +
  annotate("segment", x= -3.45, xend=-3.25, y=0.355, yend=0.355, colour="red", lty="dotdash") +
  annotate("text", x=-3.20, y=0.355, colour="black", label="sample quantiles", hjust=0) +
```

```r
  annotate("segment", x= -3.45, xend=-3.25, y=0.330, yend=0.330, colour="red", lty="dotted") +
  annotate("text", x=-3.20, y=0.330, colour="black", label="sample mode", hjust=0)
plot(Y_st_dens_hist)
```

```
## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```



On the computational side, we apply the Shapiro-Wilks (SW), D'agostino-Pearson (DP), Anderson-Darling (AD), and Jarque-Bera (JB) test to the data sets X_st and Y_st. We recall that the above normality tests (and others) rely on the assumption that the data sets have been generated by means of independent random sampling from some distribution.

The SW test

```r
# Shapiro-Wilks (*SW*) test.
# library(stats)
z <- Gauss_df$X_st
X_SW <- shapiro.test(z)
show(X_SW)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  z
## W = 0.99159, p-value = 0.5196
```

```r
z <- Gauss_df$Y_st
Y_SW <- shapiro.test(z)
show(Y_SW)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  z
## W = 0.98868, p-value = 0.2663
```

The SW test does not allow to reject the null hypothesis of Gaussianity for both the data sets X and Y.

The DP test

```
# D'Agostino-Pearson (*DP*) test.
# library(fBasics)
z <- Gauss_df$X_st
X_DP <- dagoTest(z)
show(X_DP)
```

```
##
## Title:
##  D'Agostino Normality Test
##
## Test Results:
##   STATISTIC:
##     Chi2 | Omnibus: 0.5096
##     Z3  | Skewness: -0.2012
##     Z4  | Kurtosis: 0.6849
##   P VALUE:
##     Omnibus  Test: 0.7751
##     Skewness Test: 0.8405
##     Kurtosis Test: 0.4934
```

```
z <- Gauss_df$Y_st
Y_DP <- dagoTest(z)
show(Y_DP)
```

```
##
## Title:
##  D'Agostino Normality Test
##
## Test Results:
##   STATISTIC:
##     Chi2 | Omnibus: 2.3491
##     Z3  | Skewness: 1.4481
##     Z4  | Kurtosis: -0.502
##   P VALUE:
##     Omnibus  Test: 0.309
##     Skewness Test: 0.1476
##     Kurtosis Test: 0.6157
```

The DP test does not allow to reject the null hypothesis of Gaussianity for both the data sets X and Y.

The AD test

```r
# Anderson-Darling (*AD*) test.
# library(nortest)
z <- Gauss_df$X_st
X_AD <- ad.test(z)
show(X_AD)
```

```
##
##  Anderson-Darling normality test
##
## data:  z
## A = 0.37705, p-value = 0.4057
```

```r
z <- Gauss_df$Y_st
Y_AD <- ad.test(z)
show(Y_AD)
```

```
##
##  Anderson-Darling normality test
##
## data:  z
## A = 0.43486, p-value = 0.2964
```

The AD test does not allow to reject the null hypothesis of Gaussianity for both the data sets X and Y.

The JB test

```r
# Jarque-Bera (*JB*) test.
# library(tseries)
z <- Gauss_df$X
X_JB <- jarque.bera.test(z)
show(X_JB)
```

```
##
##  Jarque Bera Test
##
## data:  z
## X-squared = 0.21255, df = 2, p-value = 0.8992
```

```r
z <- Gauss_df$Y
Y_JB <- jarque.bera.test(z)
show(Y_JB)
```

```
##
##  Jarque Bera Test
##
## data:  z
## X-squared = 2.4045, df = 2, p-value = 0.3005
```

The JB test does not allow to reject the null hypothesis of Gaussianity for both the data sets X and Y.

So far, we have collected a highly significant evidence that the data sets $\mathbf{x}$ and $\mathbf{y}$ have been generated by independent random sampling from a Gaussian distributions. Also in this case, note that the computational tests perform somewhat better in case of the X data set. This should be interpreted in light of the difference of the Cullen-Frey graphs of X and Y and the difference between the graph of the empirical density function of the X [resp. Y] data set and the graph of the standard Gaussian density.

Since we cannot reject the null hypothesis of independent sampling from a Gaussian distribution for the generation of both the data sets X and Y, in tackling the problem of determining confidence intervals for the *location* (mean) and *scale* (standard deviation) parameters of the data sets and performing hypothesis tests for the true values of these parameters we have to assume that X and Y have actually been generated by independent sampling from a Gaussian distributions.

From the t-test for the mean referred to the X data set we obtain the realization of the 90% confidence interval

```
z <- Gauss_df$X
X_090_t_test <- t.test(z, mu=mean(z), conf.level=0.90)
show(round(X_090_t_test$conf.int, digits=6))
```

```
## [1] 4.963758 5.555866
## attr(,"conf.level")
## [1] 0.9
```

and we cannot reject at the significance level 10% the null hypothesis $H_0 : \mu = 5$

```
z <- Gauss_df$X
X_01_t_test <- t.test(z, mu=5, conf.level=0.90)
show(c(round(X_01_t_test$statistic, digits=6), round(X_01_t_test$p.value, digits=6)))
```

```
##        t
## 1.452527 0.148458
```

From the t-test for the mean referred to the Y data set we obtain the realization of the 90% confidence interval

```
z <- Gauss_df$Y
Y_090_t_test <- t.test(z, mu=mean(z), conf.level=0.90)
show(round(Y_090_t_test$conf.int, digits=6))
```

```
## [1] 2.687681 3.094742
## attr(,"conf.level")
## [1] 0.9
```

and we cannot reject at the significance level 10% the null hypothesis $H_0 : \mu = 3$

```
z <- Gauss_df$Y
Y_01_t_test <- t.test(z, mu=3, conf.level=0.90)
show(c(round(Y_01_t_test$statistic, digits=6), round(Y_01_t_test$p.value, digits=6)))
```

```
##         t
## -0.884689  0.377750
```

From the chi-square test for the variance referred to the X data set we obtain the realization of the 90% confidence interval

```r
z <- Gauss_df$X
X_chisq_test <- varTest(z, alternative="two.sided", sigma.squared=var(z), conf.level=0.90)
show(round(X_chisq_test$conf.int, digits=6))
```

```
##      LCL      UCL
## 4.006315 5.871467
## attr(,"conf.level")
## [1] 0.9
```

and we cannot reject at the significance level 10% the null hypothesis $H_0 : \sigma^2 = 5$

```r
z <- Gauss_df$X
X_chisq_test <- varTest(z, alternative="two.sided", sigma.squared=5, conf.level=0.90)
show(c(round(X_chisq_test$statistic, digits=6),round(X_chisq_test$p.value, digits=6)))
```

```
## Chi-Squared
##  143.013720    0.754456
```

From the chi-square test for the variance referred to the Y data set we obtain the realization of the 90% confidence interval

```r
z <- Gauss_df$Y
Y_chisq_test <- varTest(z, alternative="two.sided", sigma.squared=var(y), conf.level=0.90)
show(round(Y_chisq_test$conf.int, digits=6))
```

```
##      LCL      UCL
## 1.893483 2.775000
## attr(,"conf.level")
## [1] 0.9
```

and we cannot reject at the significance level 10% the null hypothesis $H_0 : \sigma^2 = 2$

```r
z <- Gauss_df$Y
Y_chisq_test <- varTest(z, alternative="two.sided", sigma.squared=2, conf.level=0.90)
show(c(round(Y_chisq_test$statistic, digits=6),round(Y_chisq_test$p.value, digits=6)))
```

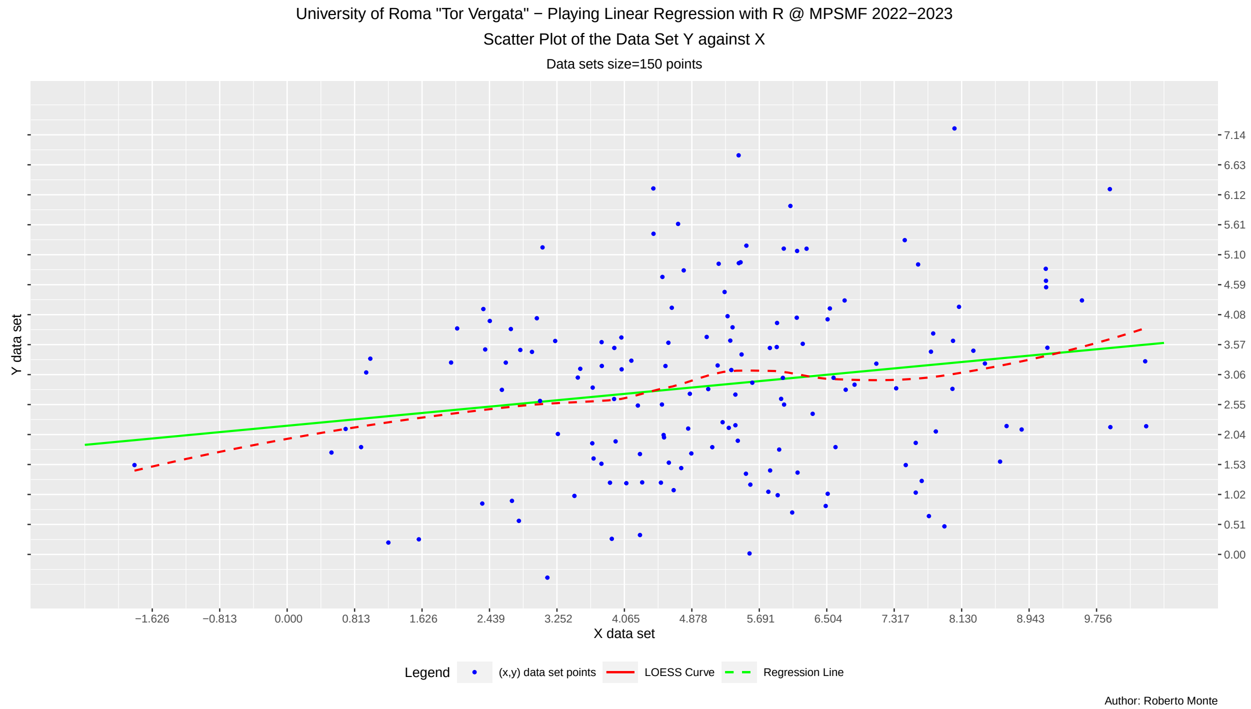
```
## Chi-Squared
##  168.979514    0.251145
```

In the end, we have collected a significant evidence that the data set $\mathbf{x}$ [resp. Y] has been generated by independent random sampling from a Gaussia distribution of location parameter $\mu_X = 5$ [resp. $\mu_Y = 3$] and scale parameter $\sigma_X = \sqrt{5}$ [resp. $\sigma_Y = \sqrt{2}$].

Now, we check whether the data sets $\mathbf{x}$ and $\mathbf{y}$ are cross-correlated or not. That is to say whether they have been generated by independent standard Gaussian distribution. To get a visual evidence of possible cross-correlation, we inspect the scatter plot of the data sets Y against X.

```r
Data_df <- Gauss_df
n <- nrow(Data_df)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - Playing Linear Regression with R \u00
paste("Scatter Plot of the Data Set Y against X")))
subtitle_content <- bquote(paste("Data sets size=", .(n), " points"))
caption_content <- "Author: Roberto Monte"
```

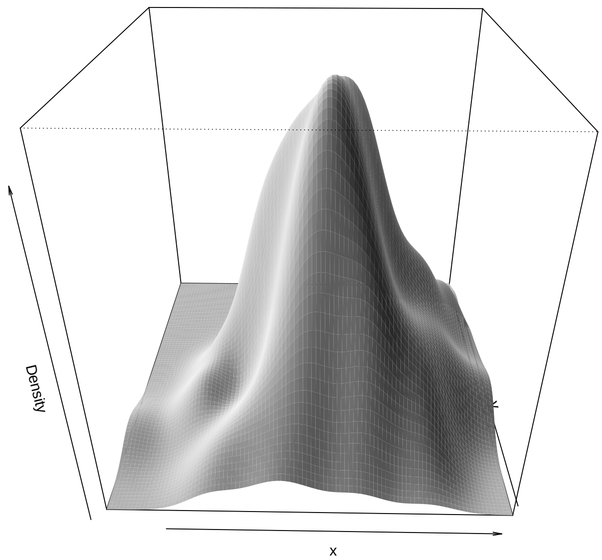
```r
x_name <- bquote("X data set")
y_name <- bquote("Y data set")
x_breaks_num <- 15
x_binwidth <- round((max(Data_df$X)-min(Data_df$X))/x_breaks_num, digits=3)
x_breaks_low <- ceiling((min(Data_df$X)/x_binwidth))*x_binwidth
x_breaks_up <- floor((max(Data_df$X)/x_binwidth))*x_binwidth
x_breaks <- c(round(seq(from=x_breaks_low, to=x_breaks_up, by=x_binwidth),3))
x_labs <- format(x_breaks, scientific=FALSE)
J <- 1
x_lims <- c(x_breaks_low-J*x_binwidth,x_breaks_up+J*x_binwidth)
y_breaks_num <- 15
y_binwidth <- round((max(Data_df$Y)-min(Data_df$Y))/y_breaks_num, digits=3)
y_breaks_low <- ceiling((min(Data_df$Y)/y_binwidth))*y_binwidth
y_breaks_up <- floor((max(Data_df$Y)/y_binwidth))*y_binwidth
y_breaks <- c(round(seq(from=y_breaks_low, to=y_breaks_up, by=y_binwidth),3))
y_labs <- format(y_breaks, scientific=FALSE)
K <- 1
y_lims <- c((y_breaks_low-K*y_binwidth), (y_breaks_up+K*y_binwidth))
col_1 <- bquote("(x,y) data set points")
col_2 <- bquote("LOESS Curve")
col_3 <- bquote("Regression Line")
leg_labs <- c(col_1, col_2, col_3)
leg_cols <- c("col_1"="blue", "col_2"="red", "col_3"="green")
leg_ord <- c("col_1", "col_2", "col_3")
Y_vs_X_sp <- ggplot(Data_df, aes(x=X, y=Y)) +
  geom_smooth(alpha=1, size=0.8, linetype="solid", aes(color="col_3"),
              method="lm" , formula=y~x, se=FALSE, fullrange=TRUE) +
  geom_smooth(alpha=1, size=0.8, linetype="dashed", aes(color="col_2"),
              method="loess", formula=y~x, se=FALSE) +
  geom_point(alpha=1, size=1.0, shape=19, aes(color="col_1")) +
  scale_x_continuous(name=x_name, breaks=x_breaks, label=x_labs, limits=x_lims) +
  scale_y_continuous(name=y_name, breaks=y_breaks, labels=NULL, limits=y_lims,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  scale_colour_manual(name="Legend", labels=leg_labs, values=leg_cols, breaks=leg_ord,
                      guide=guide_legend(override.aes=list(shape=c(19,NA,NA),
                                                           linetype=c("blank", "solid", "dashed")))) +
  theme(plot.title=element_text(hjust=0.5), plot.subtitle=element_text(hjust=0.5),
        axis.text.x=element_text(angle=0, vjust=1),
        legend.key.width=unit(1.0,"cm"), legend.position="bottom")
plot(Y_vs_X_sp)
```



A non-horizontal regression line and and an almost flat LOESS curve intertwined with the regression line constitute a visual evidence for linear dependence in the Gaussian distributions generating the data sets X and Y.

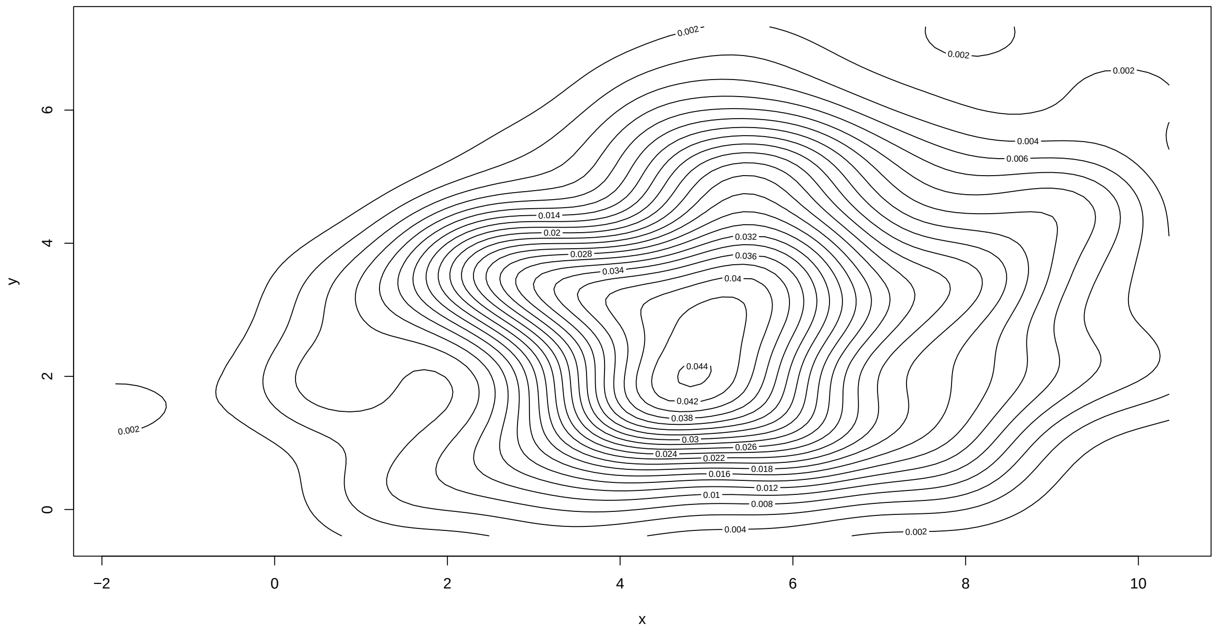
We also consider a graphical representation of the joint distribution of the data sets **x** and **y** by the function *mvn(x, multivariatePlot=...)* in the library *MVN*.

```
# library(MVN)
x <- Gauss_df$X
y <- Gauss_df$Y
X_Y_MVN_mat <- cbind(x,y)
mvn(X_Y_MVN_mat, multivariatePlot = "persp") # draw a perspective plot
```




```
## $multivariateNormality
##            Test        HZ   p value MVN
## 1 Henze-Zirkler 0.6191172 0.4295288 YES
##
## $univariateNormality
##               Test  Variable Statistic   p value Normality
## 1 Anderson-Darling     x        0.3771    0.4057    YES
## 2 Anderson-Darling     y        0.4349    0.2964    YES
##
## $Descriptives
##     n     Mean  Std.Dev   Median        Min       Max     25th     75th
## x 150 5.259812 2.190689 5.316054 -1.8398886 10.354218 3.920216 6.515333
## y 150 2.891211 1.506048 2.905184 -0.3948503  7.248796 1.746339 3.816588
##          Skew   Kurtosis
## x -0.03814361  0.1254504
## y  0.27899692 -0.2953596
```

```
mvn(X_Y_MVN_mat, multivariatePlot = "contour") # draw a contour plot
```




```
## $multivariateNormality
##           Test        HZ   p value MVN
## 1 Henze-Zirkler 0.6191172 0.4295288 YES
##
## $univariateNormality
##               Test  Variable Statistic   p value Normality
## 1 Anderson-Darling     x        0.3771    0.4057    YES
## 2 Anderson-Darling     y        0.4349    0.2964    YES
##
## $Descriptives
##     n     Mean  Std.Dev   Median        Min       Max     25th     75th
## x 150 5.259812 2.190689 5.316054 -1.8398886 10.354218 3.920216 6.515333
## y 150 2.891211 1.506048 2.905184 -0.3948503  7.248796 1.746339 3.816588
##          Skew   Kurtosis
## x -0.03814361  0.1254504
## y  0.27899692 -0.2953596
```

The contour lines of the multivariate plot appear to be somewhat elliptic with the orientation of the major axis similar to that of the regression line. This confirms the possible correlation between the Gaussian distributions generating the data sets X and Y.

Now, we consider the Pearson correlation test on the random vector (X, Y).

```
x <- Gauss_df$X
y <- Gauss_df$Y
Corr_X_Y <- cor.test(x, y, alternative="two.sided", conf.level = 0.95, method="pearson")
show(Corr_X_Y)
```

```
##
##  Pearson's product-moment correlation
##
```

```
## data:  x and y
## t = 2.406, df = 148, p-value = 0.01736
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.03483481 0.34359151
## sample estimates:
##       cor
## 0.1940132
```

In light of the Pearson correlation test, we can reject at the 5% significance level the null assumption of independence of the random variables generating the data sets X and Y.

In the end, we can check whether the random variables generating the data sets **x** and **y** are jointly Gaussian distributed by applying the Shapiro-Wilk (SWn) test for multivariate distribution or the Dormick-Hansen (DH) test.

The Shapiro-Wilk multivariate test.

```r
# library(mvnormtest)
x <- Gauss_df$X
y <- Gauss_df$Y
X_Y_SWn_mat <- rbind(x,y)
X_Y_SWn_test <- mshapiro.test(X_Y_SWn_mat)
show(X_Y_SWn_test)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  Z
## W = 0.99185, p-value = 0.5483
```

The Dormick-Hansen test.

```r
# library(mvnTest)
x <- Gauss_df$X
y <- Gauss_df$Y
X_Y_DH_mat <- cbind(x,y)
DH.test(X_Y_DH_mat)
```

```
##             Doornik-Hansen test for Multivariate Normality
##
##   data : X_Y_DH_mat
##
##   DH              : 3.573766
##   p-value         : 0.466751
##
##   Result  : Data are multivariate normal (sig.level = 0.05)
```

The application of the multivariate Shapiro-Wilk and Dormick-Hansen test shows we cannot reject the null hypothesis that the random variables generating the data sets **x** and **y** are jointly Gaussian distributed.

Summarizing, we cannot reject the hypothesis that the data sets **x** and **y** have been generated by jointly Gaussian distributed with non null correlation.

Before studying the linear regression of Y against X we recall some simple property of the linear regressions.

Assume we consider the simple regression of the regressand Y against the regressor X with error U

$$Y = f(X) + U, \tag{176}$$

where the function $f : \mathbb{R} \to \mathbb{R}$ is given by

$$f \stackrel{\text{def}}{=} \beta_0 + \beta_1 x, \quad \forall x \in \mathbb{R}, \tag{177}$$

for some $\beta_0, \beta_1 \in \mathbb{R}$. We have the following estimates for the mean and the variance of the random variable X generating the data sets $\mathbf{x} \equiv (x_k)_{k=1}^{n}$.

$$\hat{\mu}_X = \bar{x}_n \quad \text{and} \quad \hat{\sigma}_X^2 = s_{X,n}^2, \tag{178}$$

where

$$\bar{x}_n \equiv \frac{1}{n}\sum_{k=1}^{n} x_k \quad \text{and} \quad s_{X,n}^2 \equiv \frac{1}{n-1}\left(\sum_{k=1}^{n}\left(x_k^2 - \bar{x}_n\right)\right) = \frac{1}{n-1}\left(\sum_{k=1}^{n} x_k^2 - n\bar{x}_n^2\right). \tag{179}$$

Clearly equations analogous to (178) and (179) hold true for the the random variable Y generating the data sets $\mathbf{y} \equiv (y_k)_{k=1}^{n}$. Furthermore, we have the following estimate for the cross-covariance [resp. cross-correlation] of the random variable X and Y

$$\hat{\gamma}_{X,Y} = g_{X,Y,n} \qquad [\text{resp.} \qquad \hat{\rho}_{X,Y} = r_{X,Y,n}\ ], \tag{180}$$

where

$$g_{X,Y,n} = \frac{1}{n-1}\left(\sum_{k=1}^{n}\left(x_k - \bar{x}_n\right)\left(y_k - \bar{y}_n\right)\right) = \frac{1}{n-1}\sum_{k=1}^{n} x_k y_k - n\bar{x}_n^2\bar{y}_n^2 \tag{181}$$

$$[\text{resp.} r_{X,Y,n} = \frac{g_{X,Y,n}}{s_{X,n}s_{Y,n}} = \frac{\sum_{k=1}^{n} x_k y_k - n\bar{x}_n^2\bar{y}_n^2}{\sqrt{\sum\limits_{k=1}^{n} x_k^2 - n\bar{x}_n^2}\sqrt{\sum\limits_{k=1}^{n} y_k^2 - n\bar{y}_n^2}}\ ]. \tag{182}$$

For example,

```
x <- Gauss_df$X
n <- length(x)
round(mean(x),digits=14)==round(sum(x)/n,digits=14)
```

```
## [1] TRUE
```

```
x <- Gauss_df$X
n <- length(x)
round(sd(x),digits=14)==round(sqrt((sum(x^2)-n*mean(x)^2)/(n-1)),digits=14)
```

```
## [1] TRUE
```

```
y <- Gauss_df$Y
n <- length(y)
round(mean(y),digits=14)==round(sum(y)/n,digits=14)
```

```
## [1] TRUE
```

```
y <- Gauss_df$Y
n <- length(y)
round(sd(y),digits=14)==round(sqrt((sum(y^2)-n*mean(y)^2)/(n-1)),digits=14)
```

```
## [1] TRUE
```

```
x <- Gauss_df$X
y <- Gauss_df$Y
n <- length(x)
round(cov(x,y, method="pearson"), digits=14)==round(((1/(n-1))*(sum(x*y)-n*mean(x)*mean(y))), digits=14
```

```
## [1] TRUE
```

```
x <- Gauss_df$X
y <- Gauss_df$Y
n <- length(x)
round(cor(x,y, method="pearson"), digits=14)==round((1/(n-1))*((sum(x*y)-n*mean(x)*mean(y))/(sd(x)*sd(y
```

```
## [1] TRUE
```

and

```
x <- Gauss_df$X
y <- Gauss_df$Y
n <- length(x)
round(cor(x,y, method="pearson"), digits=14)==round((sum(x*y)-n*mean(x)*mean(y))/(sqrt(sum(x^2)-n*mean(
```

```
## [1] TRUE
```

In turn, the estimates of the regression coefficients $\hat{\beta}_0$ and $\hat{\beta}_1$ are given by

$$\hat{\beta}_1 = \frac{\hat{\gamma}_{X,Y}}{\hat{\sigma}_X^2} = \hat{\rho}_{X,Y}\frac{\hat{\sigma}_Y}{\hat{\sigma}_X} = \frac{g_{X,Y,n}}{s_{X,n}^2} = r_{X,Y,n}\frac{s_{Y,n}}{s_{X,n}} = \frac{\sum_{k=1}^n x_k y_k - n\bar{x}_n\bar{y}_n}{\sum_{k=1}^n x_k^2 - n\bar{x}_n^2} \tag{183}$$

and

$$\hat{\beta}_0 = \hat{\mu}_Y - \hat{\beta}_1\hat{\mu}_X = \frac{\bar{y}_n \sum_{k=1}^n x_k^2 - \bar{x}_n \sum_{k=1}^n x_k y_k}{\sum_{k=1}^n x_k^2 - n\bar{x}_n^2} \tag{184}$$

Now, we consider the linear regression of the data set $\mathbf{y}$ against $\mathbf{x}$.

```
Y_X_lm <- lm(data=Gauss_df, Y~X)
class(Y_X_lm)
```

```
## [1] "lm"
```

```
structure(Y_X_lm)
```

```
##
## Call:
## lm(formula = Y ~ X, data = Gauss_df)
##
## Coefficients:
## (Intercept)            X
##      2.1897       0.1334
```

```
x <- Gauss_df$X
all(x==Y_X_lm$model$X)
```

```
## [1] TRUE
```

```
y <- Gauss_df$Y
all(y==Y_X_lm$model$Y)
```

```
## [1] TRUE
```

The coefficients in the *structure* output are the estimates of the intercept β_0 and the slope β_1 of the regression line.

```
x <- Y_X_lm$model$X
y <- Y_X_lm$model$Y
beta_1 <- as.numeric(Y_X_lm$coefficients[2])
n <- length(x)
round(beta_1,digits=14)==round(((sum(x*y)-n*mean(x)*mean(y))/(sum(x^2)-n*mean(x)^2)),digits=14)
```

```
## [1] TRUE
```

and

```
x <- Y_X_lm$model$X
y <- Y_X_lm$model$Y
beta_0 <- as.numeric(Y_X_lm$coefficients[1])
n <- length(x)
round(beta_0,digits=14)==round(((mean(y)*sum(x^2)-mean(x)*sum(x*y))/(sum(x^2)-n*mean(x)^2)),digits=14)
```

```
## [1] TRUE
```

We recall that the *fitted values* [resp. *residuals*] of the linear regression are defined by

$$\hat{y}_k \stackrel{\text{def}}{=} \hat{\beta}_0 + \hat{\beta}_1 x_k, \quad [\text{resp.} \hat{u}_k \stackrel{\text{def}}{=} y_k - \hat{y}_k], \quad \forall k = 1, \dots, n. \tag{185}$$

Recall also that we have

$$\frac{1}{n}\sum_{k=1}^{n} \hat{y}_k = \bar{y}_n, \quad \frac{1}{n}\sum_{k=1}^{n} \hat{u}_k = 0, \quad \frac{1}{n}\sum_{k=1}^{n} x_k \hat{u}_k = 0 \tag{186}$$

For example, we check that also Equations (185) and (186) hold true. Note that in the following chunks of code, in order to make the desired comparisons, we found appropriate to reduce some objects retrieved from the linear model *Y_X_lm* to numbers and vectors.

```
x <- Y_X_lm$model$X
y_hat <- as.vector(Y_X_lm$fitted)
beta_0 <- as.numeric(Y_X_lm$coefficients[1])
beta_1 <- as.numeric(Y_X_lm$coefficients[2])
round(y_hat,digits=14)==round(beta_0+beta_1*x,digits=14)
```

```
##  [1] FALSE  TRUE  TRUE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [13]  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE
## [25] FALSE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE
## [37]  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [49]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE  TRUE  TRUE  TRUE  TRUE
## [61]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE
## [73]  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE  TRUE FALSE  TRUE  TRUE  TRUE
```

```
##  [85]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [97]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE FALSE
## [109]  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [121]  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE FALSE  TRUE FALSE  TRUE
## [133]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE FALSE
## [145]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
```

However,

```
x <- Y_X_lm$model$X
y_hat <- as.vector(Y_X_lm$fitted)
beta_0 <- as.numeric(Y_X_lm$coefficients[1])
beta_1 <- as.numeric(Y_X_lm$coefficients[2])
all(round(y_hat,digits=12)==round(beta_0+beta_1*x,digits=12))
```

```
## [1] TRUE
```

and

```
x <- Y_X_lm$model$X
y_hat <- as.vector(Y_X_lm$fitted)
beta_0 <- as.numeric(Y_X_lm$coefficients[1])
beta_1 <- as.numeric(Y_X_lm$coefficients[2])
all.equal(y_hat, beta_0+beta_1*x)
```

```
## [1] TRUE
```

Similarly,

```
y <- Y_X_lm$model$Y
y_hat <- as.vector(Y_X_lm$fitted)
u_hat <- as.vector(Y_X_lm$residuals)
round(u_hat,digits=14)==round(y-y_hat,digits=14)
```

```
##   [1]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [13]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [25]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [37]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [49]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [61]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [73]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [85]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [97]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [109]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [121]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE
## [133]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [145]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
```

However,

```r
y <- Y_X_lm$model$Y
y_hat <- as.vector(Y_X_lm$fitted)
u_hat <- Y_X_lm$residuals
all(round(u_hat,digits=13)==round(y-y_hat,digits=13))
```

```
## [1] TRUE
```

and

```r
y <- Y_X_lm$model$Y
y_hat <- as.vector(Y_X_lm$fitted)
u_hat <- as.vector(Y_X_lm$residuals)
all.equal(u_hat, y-y_hat)
```

```
## [1] TRUE
```

Furthermore,

```r
y <- Y_X_lm$model$Y
y_hat <- as.vector(Y_X_lm$fitted)
round(mean(y),digits=14)==round(mean(y_hat),digits=14)
```

```
## [1] TRUE
```

```r
u_hat <- as.vector(Y_X_lm$residuals)
round(mean(u_hat),digits=14)
```

```
## [1] 0
```

```r
x <- Y_X_lm$model$X
u_hat <- as.vector(Y_X_lm$residuals)
round(mean(x*u_hat),digits=14)
```

```
## [1] 0
```

More detailed information on the linear regression of the data set $\mathbf{y}$ against $\mathbf{x}$ can be obtained by the function *summary*.

```r
summary(Y_X_lm)
```

```
##
## Call:
## lm(formula = Y ~ X, data = Gauss_df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.0029 -1.1879 -0.0344  0.9608  3.9863
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.18966    0.31571   6.936 1.17e-10 ***
## X            0.13338    0.05544   2.406   0.0174 *
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.482 on 148 degrees of freedom
## Multiple R-squared:  0.03764,    Adjusted R-squared:  0.03114
## F-statistic: 5.789 on 1 and 148 DF,  p-value: 0.01736
```

The Residuals section of the model output presents 5 summary points. These points express the symmetry of the residuals of the linear regression about their zero mean value. The better the linear model fits the data, the stronger the symmetry of the summary points. A lack of symmetry means that the fitted value $\hat{y}_k$ of the dependent variable fall far away from the observed value y_k, for some $k = 1, \dots, n$. More specifically, the better the model fits the data, the closer the summary points are to the corresponding summary points of the zero centered Gaussian distribution with standard deviation given by the *Residual Standard Error* (*RSE*), because *RSE* constitutes an estimate of the (unknown) standard deviation of the error U in the linear regression. We recall that

$$RSE = \sqrt{\frac{1}{df}\sum_{k=1}^{n}\hat{u}_k^2}, \tag{187}$$

where *df* stands for the *degrees of freedom* of the the residuals of the linear model, that is the *number of observations* of the linear model minus the *number of coefficients*. In symbols,

$$df = n - c, \tag{188}$$

where n is the size of the data set $\mathbf{x}$ and c is the number of the coefficients of the linear regression (we clearly have $c = 2$ for a simpe linear regression). Moreover, RSE can be computed by the function *sigma()* (see https://stat.ethz.ch/R-manual/R-devel/library/stats/html/sigma.html). For example,

```
RSE <- sigma(Y_X_lm)
u_hat <- as.vector(Y_X_lm$residuals)
df <- Y_X_lm$df.residual
round(RSE,digits=14)==round(sqrt((1/df)*sum(u_hat^2)),digits=14)
```

```
## [1] TRUE
```

```
show(RSE)
```

```
## [1] 1.482415
```

As a consequence, we have the following table which compares the summary points of the zero centered Gaussian distribution with standard deviation equal to *RSE* with the corresponding summary points of the residuals of the linear model

$$\begin{array}{ccccc} Min\,(99.73\%) & 1stQ & Median & 3rdQ & Max\,(99.73\%) \\ -4.4472 & -0.9999 & 0.0000 & 0.9999 & 4.4472 \\ -3.0029 & -1.1879 & -0.0344 & 0.9608 & 3.9863, \end{array} \tag{189}$$

since

$$\begin{array}{c} Min\,(99.73\%) = Mean - 3 * RSE, \qquad Max\,(99.73\%) = Mean + 3 * RSE, \\ 1stQ = qnorm(0.25,\ mean = 0,\ sd = RSE,\ lower.tail = TRUE), \\ 33dQ = qnorm(0.75,\ mean = 0,\ sd = RSE,\ lower.tail = TRUE), \end{array} \tag{190}$$

Note that the five summary points of the residuals of the linear model, except perhaps the minimum, fit somehow the corresponding summary points of the zero centered Gaussian distribution with *RSE* standard deviation, presented in the first row of the above table. In fact, for such a distribution we have

```r
RSE <- sigma(Y_X_lm)
Min_99.73 <- -round(3*RSE, digits=4)
Ist_Q <- round(qnorm(0.25, mean=0, sd=RSE, lower.tail=TRUE),4)
IIIrd_Q <-  round(qnorm(0.75, mean=0, sd=RSE, lower.tail=TRUE),4)
Max_99.73 <- round(3*RSE, digits=4)
show(c(Min_99.73,Ist_Q, IIIrd_Q, Max_99.73))
```

```
## [1] -4.4472 -0.9999  0.9999  4.4472
```

In addition, computing the skewness and kurtosis of the residuals we obtain

```r
# library(moments)
Skew <- moments::skewness(Y_X_lm$residuals) # theoretical value 0.
Kurt <- moments::kurtosis(Y_X_lm$residuals) # theoretical value 3.
show(c(Skew, Kurt))
```

```
## [1] 0.2395991 2.5987031
```

These result confirm an overall reasonable fit between the residual generating distribution and the zero centered Gaussian distribution with RSE standard deviation.

The Coefficients section of the model output presents two rows of 4 summary points. These points are related to the estimates of the linear regression coefficients.

The first [resp. second] Coefficients Estimate is the OLS (Ordinary Least Square) estimated value of the intercept [resp. slope] of the regression line that we have already obtained as the output of the *structure* command.

The first [resp. second] Coefficients Standard Error measures the average amount by which the estimated intercept [resp. slope] regression parameter may vary from the true value. The smaller is the standard error (relative to its coefficient estimate), the better is the fit of the model. Eventually, the first [resp. second] Standard Error is the standard deviation of the unbiased intercept [resp. slope] estimator under the assumption that the noise variable U is uncorrelated with the independent variable X and is Gaussian distributed.

The first [resp. second] Coefficients t-value is the value taken by the t-statistic derived from the Gaussian distributed intercept [resp. slope] estimator under the null hypothesis that the true value of the intercept [resp. slope] is zero. The higher is the t-value, the more reasonable is the rejection of the null hypothesis.

The first [resp. second] Coefficients $Pr(>|t|)$ is the p-value correspponding to the value of the t-statistic derived from the Gaussian distributed intercept [resp. slope] estimator. The smaller is the p-value, the more reasonable is the rejection of the null hypothesis.

We stress once more that the validity of both t-values and p-values is subject to the assumption that the error variable U in the linear regression is uncorrelated with the dependent variable X and Gaussian distributed. Under this assumption the estimators for the intercept and slope of the linear regression turn out to be the best linear unbiased estimators $BLUE$. In addition, the residuals of the linear regression turn out to be uncorrrelated and Gaussian distributed. Therefore, as we will see, a necessary step to assess the adequacy of the linear model is the residual analysis.

Proceeding with the summary examination, the Residual or Regression Standard Error (RSE) measures the overall quality of the linear regression fit. RSE is an estimate of the standard deviation of the residuals in the linear regression model. In a linear regression model, the best single error statistic to consider is RSE. The lower is RSE the better is the linear regression fit. Actually, the estimated coefficients of the linear regression are obtained minimizing $SSE \equiv RSS$ and $\min SSE \equiv RSS$.

We set

```
Data_lm <- Y_X_lm
```

and we store the *Data_lm* summary in a list

```
Data_lm_summary <- summary(Data_lm)
```

We recall that TSS = Total Sum of Squares represents the variability of the dependent varieble Y (the unbiased variance of Y is given by *TSS/(n-1)*)

```
TSS <- sum((Data_lm$model$Y - mean(Data_lm$model$Y))^2)
show(TSS)
```

```
## [1] 337.959
```

ESS = Explained Sum of Squares represents the part of TSS which can be explained, via the linear model, by the variability of the independent variable X.

```
ESS <- sum((Data_lm$fitted.values - mean(Data_lm$model$Y))^2)
show(ESS)
```

```
## [1] 12.72116
```

also

```
ESS <- sum((Data_lm$fitted.values - mean(Data_lm$fitted))^2)
show(ESS)
```

```
## [1] 12.72116
```

RSS = Residual Sum of Squares Regression represents the part of TSS which cannot be explained, via the linear model, by the variability of the independent variable X.

```
RSS <- sum((Data_lm$fitted.values - Data_lm$model$Y)^2)
show(RSS)
```

```
## [1] 325.2379
```

also

```
RSS <- sum(Data_lm$residual^2)
show(RSS)
```

```
## [1] 325.2379
```

Note that

```
round(TSS,digits=12)==round(ESS+RSS,digits=12)
```

```
## [1] TRUE
```

Note also that $RSE = \sqrt{RSS/df}$ where $df* stands for the degrees of freedom of the model, that is the size of the sample minus the number of the regression coefficients.

```
RSE <- sqrt(RSS/Data_lm$df.residual)
show(RSE)
```

```
## [1] 1.482415
```

To be compared with

```
Data_lm_summary$sigma
```

```
## [1] 1.482415
```

The coefficient of determination or Multiple R-squared R^2 is the portion/percentage of the variability of the dependent variable Y which can be explained, via the linear model, in terms of the independent variable X. Formally, $R^2 \equiv ESS/TSS$. In contrast, $1 - R^2 = RSS/TSS$ is the portion/percentage of the variability of the explained variable Y which can be explained via the linear model in terms of the explanatory variable X. We have

```
R_square <- ESS/TSS
show(R_square)
```

```
## [1] 0.03764113
```

To be compared with

```
Data_lm_summary$r.squared
```

```
## [1] 0.03764113
```

The R-squared statistic provides a measure of how well the model is fitting the actual data. More specifically, R^2 is a measure of the linear relationship between the independent and the dependent variable. R^2 always lies between 0 and 1. The closer R^2 to 1, the better the linear regression explains the variability of the dependent variable in terms of the dependent variable. Note that in multiple regression models, the R^2 always increase as more variables are considered. Therefore, the Adjusted R-squared, is the preferred measure of the variability of the dependent variable in terms of the dependent variables, as $\tilde{R}^2$ accounts for the number of variables considered. We have

$$\tilde{R}^2 \stackrel{\text{def}}{=} 1 - \frac{\frac{RSS}{n-c}}{\frac{TSS}{n-1}}. \tag{191}$$

In the case considered,

```
n <- 150
c <- 2
Adj_R_square <- 1-(RSS/(n-c))/(TSS/(n-1))
show(Adj_R_square)
```

```
## [1] 0.03113871
```

To be compared with

```
Data_lm_summary$adj.r.squared
```

```
## [1] 0.03113871
```

In the end, the F-statistic is a good indicator of whether there is a linear relationship between the dependent and the independent variable. The further the F-statistic is from 1 the better it is. However, how much larger the F-statistic needs to be depends on both the number of data points and the number of independent variables. Generally, when the number of data points is large, an F-statistic that is only a little bit larger than 1 is already sufficient to reject the null hypothesis of no linear relationship between the dependent and the independent variable. The reverse is true as the number of data points is small: a large F-statistic is required to affirm that there might be a linear relationship between independent and dependent variables. We have

We have

$$F \overset{\text{def}}{=} \frac{n-c}{c-1}\frac{R^2}{1-R^2}. \tag{192}$$

In the case considered,

```
n <- 150
c <- 2
F_stat <- ((n-c)/(c-1))*((R_square)/(1-R_square))
show(F_stat)
```

```
## [1] 5.788784
```

To be compared with

```
Data_lm_summary$fstatistic
```

```
##      value      numdf      dendf
##   5.788784   1.000000 148.000000
```

As mentioned above, to assess the adequacy of the linear model, we need to show that the residuals are likely generated by independent sampling from a gaussian distribution.

We build an appropriate data frame *Y_X_df* for the linear model *Y_X_lm*

```
Y_X_df <- data.frame(t=1:size, X=Gauss_df$X, Y=Gauss_df$Y, Fit=Y_X_lm$fitted.values, Res=Y_X_lm$residual
head(Y_X_df)
```

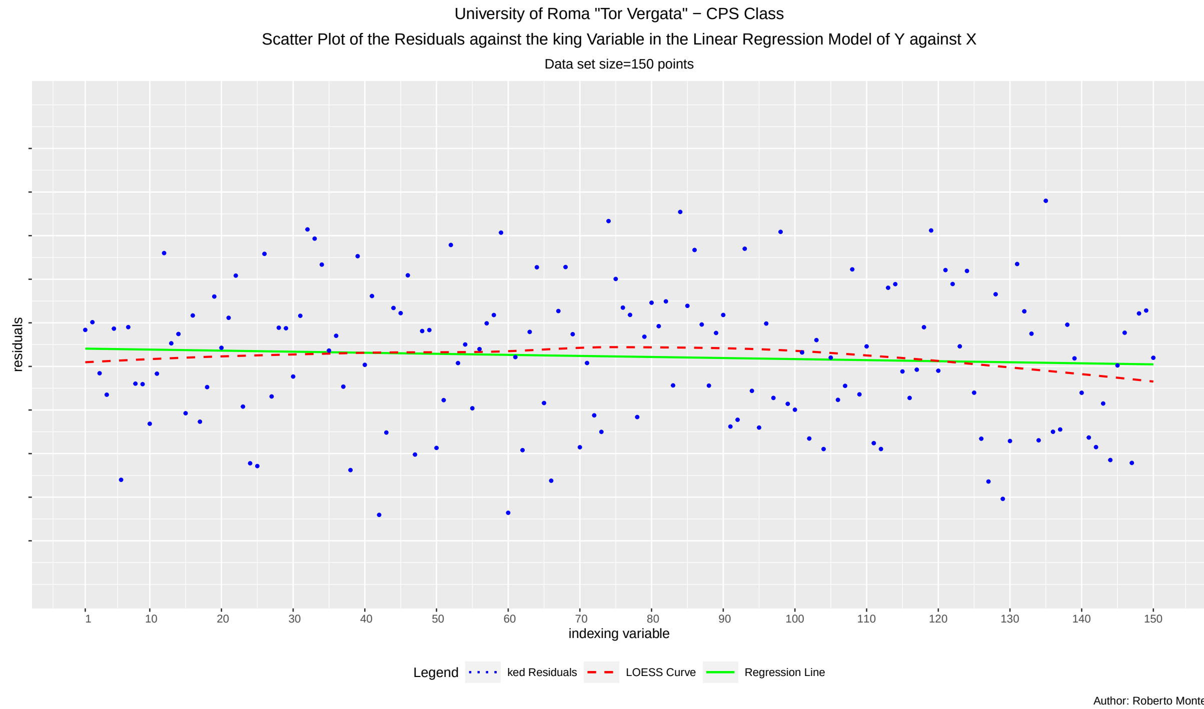
```
##   t        X         Y      Fit        Res
## 1 1 8.026032 3.6347229 3.260169  0.3745542
## 2 2 6.732687 2.8021444 3.087663 -0.2855183
## 3 3 5.477368 3.4026390 2.920229  0.4824102
## 4 4 5.368712 3.8646020 2.905736  0.9588656
## 5 5 1.220284 0.2013109 2.352421 -2.1511096
## 6 6 8.043723 7.2487956 3.262528  3.9862673
```

```
tail(Y_X_df)
```

```
##       t        X         Y      Fit        Res
## 145 145 6.214936 3.5836846 3.018605  0.5650793
## 146 146 2.351966 0.8657286 2.503364 -1.6376352
## 147 147 2.049241 3.8455583 2.462986  1.3825719
## 148 148 6.334291 2.3926662 3.034525 -0.6418586
## 149 149 7.341035 2.8264979 3.168804 -0.3423061
## 150 150 4.517738 2.5508327 2.792234 -0.2414010
```

We consider the residuals scatter plot.

```
Data_df <- Y_X_df
n <- nrow(Data_df)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - CPS Class",
                             "Scatter Plot of the Residuals against the king Variable in the Linear Reg
subtitle_content <- bquote(paste("Data set size=", .(n), " points"))
caption_content <- "Author: Roberto Monte"
x_name <- bquote("indexing variable")
y_name <- bquote("residuals")
x_breaks <- c(1,seq(from=10, to=n, by=10))
x_labs <- format(x_breaks, scientific=FALSE)
x_lims <- c(1,n)
y_breaks_num <- 10
y_binwidth <- round((max(Data_df$Res)-min(Data_df$Res))/y_breaks_num, digits=3)
y_breaks_low <- ceiling((min(Data_df$Res)/y_binwidth))*y_binwidth
y_breaks_up <- floor((max(Data_df$Res)/y_binwidth))*y_binwidth
y_breaks <- c(round(seq(from=y_breaks_low, to=y_breaks_up, by=y_binwidth),3))
y_labs <- format(y_breaks, scientific=FALSE)
K <- 1
y_lims <- c((y_breaks_low-K*y_binwidth), (y_breaks_up+K*y_binwidth))
col_1 <- bquote("ked Residuals")
col_2 <- bquote("LOESS Curve")
col_3 <- bquote("Regression Line")
leg_labs <- c(col_1, col_2, col_3)
leg_cols <- c("col_1"="blue", "col_2"="red", "col_3"="green")
leg_ord <- c("col_1", "col_2", "col_3")
Data_df_Res_sp <- ggplot(Data_df) +
  geom_smooth(alpha=1, size=0.8, linetype="solid", aes(x=t, y=Z_1, color="col_3"),
              method="lm" , formula=y ~ x, se=FALSE, fullrange=TRUE) +
  geom_smooth(alpha=1, size=0.8, linetype="dashed", aes(x=t, y=Z_1, color="col_2"),
              method="loess", formula=y ~ x, se=FALSE) +
  geom_point(alpha=1, size=1.0, shape=19, aes(x=t, y=Z_1, color="col_1")) +
  scale_x_continuous(name=x_name, breaks=x_breaks, label=x_labs, limits=x_lims) +
  scale_y_continuous(name=y_name, breaks=y_breaks, labels=NULL, limits=y_lims,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  scale_colour_manual(name="Legend", labels=leg_labs, values=leg_cols, breaks=leg_ord,
                      guide=guide_legend(override.aes=list(shape=c(NA,NA,NA), linetype=c("dotted", "dash
  theme(plot.title=element_text(hjust=0.5), plot.subtitle=element_text(hjust=0.5),
        axis.text.x=element_text(angle=0, vjust=1),
        legend.key.width=unit(1.0,"cm"), legend.position="bottom")
plot(Data_df_Res_sp)
```



From the inspection of the scatter plot, we have visual evidence for homogeneously spread sample points around an almost horizontal regression line. The almost flat LOESS line swinging slightly around the regression line strengthen this evidence,

Acording to the visual evidence we are lead to think that the data sets have been generated by indepemdent sampling from the same distribution, at least by independent sampling from a family of distributions with constant mean and variance.

We tackle the issue of constant mean by applying the Kwiatkowski-Phillips-Schmidt-Shin ($KPSS$) and test Dickey-Fueller (DF) test in the simplest form.

We apply the $KPSS$ test.

```
# library(urca) # The library for this vesion of the test
z <- Y_X_df$Res   # Choosing the data set to be tested

Res_KPSS_mu <- ur.kpss(z, type="mu", lags="nil", use.lag=NULL)
# Applying the simplest form of the KPSS test which considers the hull hypothesis that
# the data set is generated by an autoregressive process with constant mean,
# while the alternative hypothesis is that the data set is generated a process with a random walk compo

summary(Res_KPSS_mu)# Showing the result of the test
```

```
##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: mu with 0 lags.
##
## Value of test-statistic is: 0.0544
##
```

```
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

The test statistics of the $KPSS$ test takes value outside the rejection region at the significance level $\alpha = 0.1$ or $\alpha = 10\%$. Therefore we cannot reject the null hypothesis in favor of the alternative.

We apply the DF test.

```
# library(urca) # The library for this vesion of the test.
z <- Y_X_df$Res   # Choosing the data set to be tested.
no_lags <- 0            # Setting the lag parameter for the test.

Res_DF_none <- ur.df(z, type="none", lags=no_lags, selectlags="Fixed")
# Applying the form of the DF test which considers the null hypothesis that the data set is generated b
# a process with a random walk component, while the alternative hypothesis is that the data set is gene
# by an autoregressive process with no drift and trend.
summary(Res_DF_none) # Showing the result of the test
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression none
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 - 1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.9812 -1.1902 -0.0749  0.9750  3.9012
##
## Coefficients:
##         Estimate Std. Error t value Pr(>|t|)
## z.lag.1 -1.03956    0.08212  -12.66   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.481 on 148 degrees of freedom
## Multiple R-squared:  0.5198, Adjusted R-squared:  0.5166
## F-statistic: 160.2 on 1 and 148 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic is: -12.6582
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau1 -2.58 -1.95 -1.62
```

The test statistics of the DF test takes value inside the rejection region at the significance level $\alpha = 0.01$ or $\alpha = 1\%$. Therefore we can reject the null hypothesis in favor of the alternative.

Note again that, with the goal of not rejecting the null hypotesis, the higher is the value of the parameter β_0 (among the standard ones) the strongest is the not rejection. On the contrary, with the goal of rejecting the null hypotesis the lowest is the value of the parameter α the strongest is the rejection.

The lack of rejection of the null hypothesis of the *KPSS* test and the rejection of the null hypothesis of the *DF* test in favor of the alternative hypothesis constitute together significant evidence that the residuals of the linear regression have been generated by a process with constant mean.

To deal with the issue of constant variance, besides the visual inspection of the scatterplots of the residuals, we apply also on the *Breusch-Pagan* (*BP*) and *White* (*W*) test presented above.

The (Studentized) *BP* test.

```
# library(lmtest) # The library for this vesion of the test.
# Studentized Breusch-Pagan test
x <- Y_X_df$t      # The independent variable in the test
y <- Y_X_df$Res    # The dependent variable in the test
Res_BP <- bptest(formula=y~x, varformula=NULL, studentize=TRUE)
# The command which launchs the BP test and stores the results in the Res_BP list.
show(Res_BP) # The summary of the Gauss_1_BP list.
```

```
##
##  studentized Breusch-Pagan test
##
## data:  y ~ x
## BP = 0.030347, df = 1, p-value = 0.8617
```

The *W* test.

```
# library(lmtest)
# White test
x <- Y_X_df$t      # The independent variable in the test
y <- Y_X_df$Res    # The dependent variable in the test
var.formula <- ~ x+I(x^2) # The formula which allows to switch from *BP* to *W* test.
Res_W <- bptest(formula=y~x, varformula=var.formula, studentize=TRUE)
# The command which launchs the W test and stores the results in the Res_W list.
show(Res_W) # The summary of the Gauss_1_W list.
```

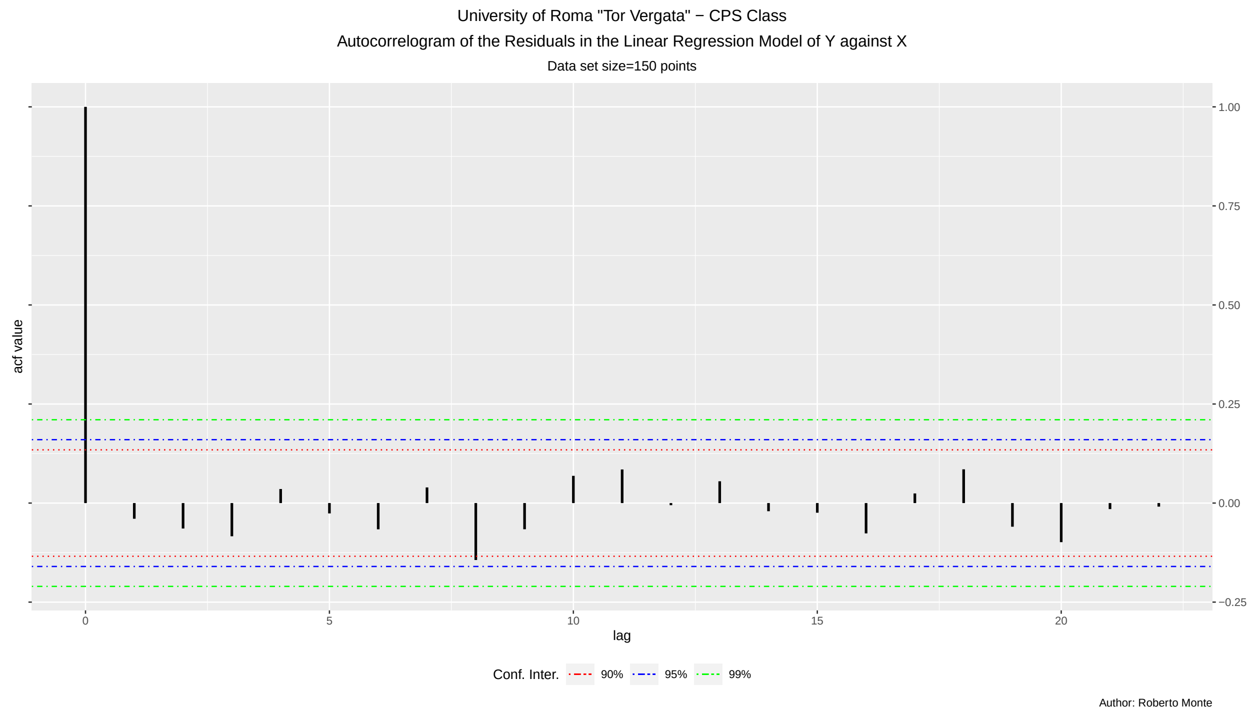
```
##
##  studentized Breusch-Pagan test
##
## data:  y ~ x
## BP = 0.072375, df = 2, p-value = 0.9645
```

Both the *BP* and *W* test cannot reject the null hypothesis of homosckedasticity at the lowest standard 10% significance level. Hence, in light of the visual inspections, and the results of the *BP* and *W* test, we have significant evidences to not reject the null hypothesis that residuals of the linear regression have been generated by a process with constant variance.

The third important step to assess the adequacy of the linear model is that the residuals have been generated by independent random sampling from the same distribution (not necessarily Gaussian). As above, we can make a visual check of this by plotting the autocorrelogram and the partial autocorrelogram of the residuals.

The autocorrelogram of the residuals.

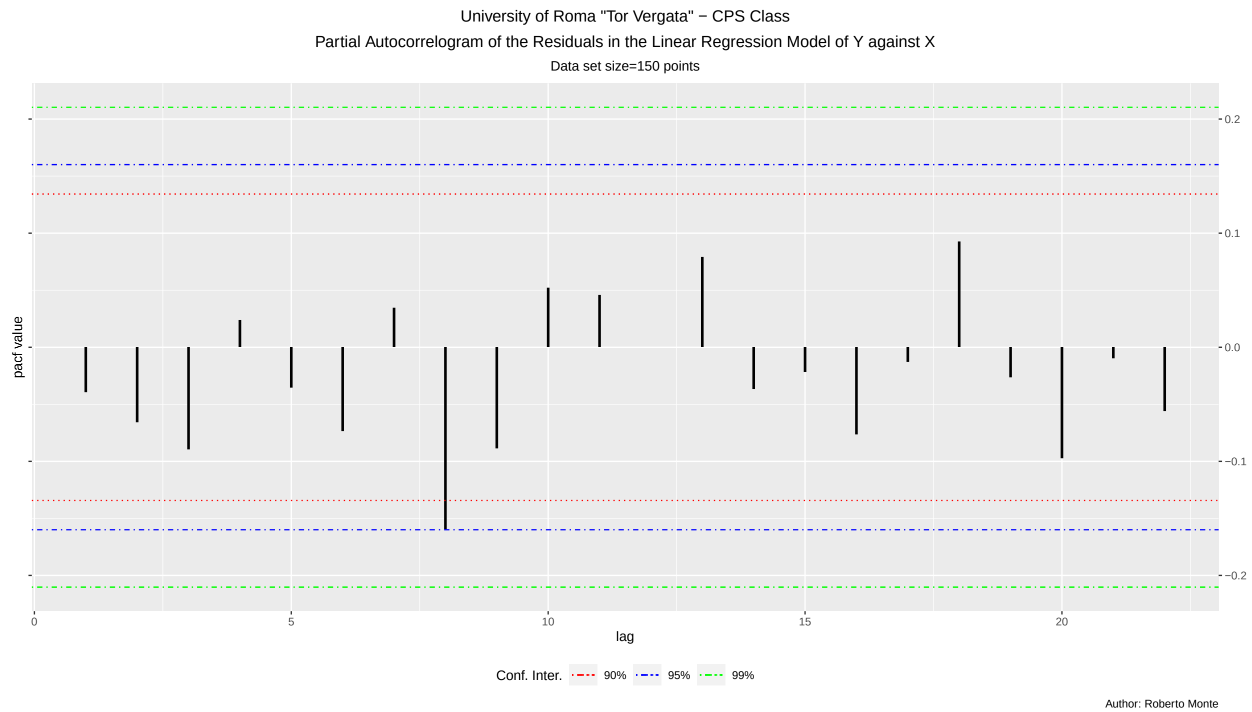
```
z <- Y_X_df$Res
n <- length(z)
maxlag <- ceiling(10*log10(n))
Aut_Fun_z <- acf(z, lag.max=maxlag, type="correlation", plot=FALSE)
ci_90 <- qnorm((1+0.90)/2)/sqrt(n)
ci_95 <- qnorm((1+0.95)/2)/sqrt(n)
ci_99 <- qnorm((1+0.99)/2)/sqrt(n)
Plot_Aut_Fun_z <- data.frame(lag=Aut_Fun_z$lag, acf=Aut_Fun_z$acf)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - CPS Class",
                             "Autocorrelogram of the Residuals in the Linear Regression Model of Y agai
subtitle_content <- bquote(paste("Data set size=", .(n), " points"))
caption_content <- "Author: Roberto Monte"
ggplot(Plot_Aut_Fun_z, aes(x=lag, y=acf)) +
  geom_segment(aes(x=lag, y=rep(0,length(lag)), xend=lag, yend=acf), size=1, col="black") +
  geom_hline(aes(yintercept=-ci_90, color="CI_90"), show.legend=TRUE, lty=3) +
  geom_hline(aes(yintercept=ci_90, color="CI_90"), lty=3) +
  geom_hline(aes(yintercept=ci_95, color="CI_95"), show.legend=TRUE, lty=4) +
  geom_hline(aes(yintercept=-ci_95, color="CI_95"), lty=4) +
  geom_hline(aes(yintercept=-ci_99, color="CI_99"), show.legend=TRUE, lty=4) +
  geom_hline(aes(yintercept=ci_99, color="CI_99"), lty=4) +
  scale_x_continuous(name="lag", breaks=waiver(), label=waiver()) +
  scale_y_continuous(name="acf value", breaks=waiver(), labels=NULL,
                     sec.axis=sec_axis(~., breaks=waiver(), labels=waiver())) +
  scale_color_manual(name="Conf. Inter.", labels=c("90%","95%","99%"),
                     values=c(CI_90="red", CI_95="blue", CI_99="green")) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  theme(plot.title=element_text(hjust=0.5),
        plot.subtitle=element_text(hjust= 0.5),
        plot.caption=element_text(hjust=1.0),
        legend.key.width=unit(0.8,"cm"), legend.position="bottom")
```



The partial autocorrelogram of the residuals*.

```
z <- Y_X_df$Res
n <- length(z)
maxlag <- ceiling(10*log10(n))
P_Aut_Fun_z <- pacf(z, lag.max=maxlag, type="correlation", plot=FALSE)
ci_90 <- qnorm((1+0.90)/2)/sqrt(n)
ci_95 <- qnorm((1+0.95)/2)/sqrt(n)
ci_99 <- qnorm((1+0.99)/2)/sqrt(n)
Plot_P_Aut_Fun_z <- data.frame(lag=P_Aut_Fun_z$lag, pacf=P_Aut_Fun_z$acf)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - CPS Class",
                             "Partial Autocorrelogram of the Residuals in the Linear Regression Model o
subtitle_content <- bquote(paste("Data set size=", .(n), " points"))
caption_content <- "Author: Roberto Monte"
ggplot(Plot_P_Aut_Fun_z, aes(x=lag, y=pacf)) +
  geom_segment(aes(x=lag, y=rep(0,length(lag)), xend=lag, yend=pacf), size=1, col="black") +
  geom_hline(aes(yintercept=-ci_90, color="CI_90"), show.legend=TRUE, lty=3) +
  geom_hline(aes(yintercept=ci_90, color="CI_90"), lty=3) +
  geom_hline(aes(yintercept=ci_95, color="CI_95"), show.legend=TRUE, lty=4) +
  geom_hline(aes(yintercept=-ci_95, color="CI_95"), lty=4) +
  geom_hline(aes(yintercept=-ci_99, color="CI_99"), show.legend=TRUE, lty=4) +
  geom_hline(aes(yintercept=ci_99, color="CI_99"), lty=4) +
  scale_x_continuous(name="lag", breaks=waiver(), label=waiver()) +
  scale_y_continuous(name="pacf value", breaks=waiver(), labels=NULL,
                     sec.axis=sec_axis(~., breaks=waiver(), labels=waiver())) +
  scale_color_manual(name="Conf. Inter.", labels=c("90%","95%","99%"),
                     values=c(CI_90="red", CI_95="blue", CI_99="green")) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  theme(plot.title=element_text(hjust=0.5),
        plot.subtitle=element_text(hjust= 0.5),
```

```
        plot.caption=element_text(hjust=1.0),
        legend.key.width=unit(0.8,"cm"), legend.position="bottom")
```



The number of peaks corresponding to positive lags crossing the confidence lines of the autocorrelogram is within the statistical tolerance. In fact, we have no peaks crossing the 95% confidence lines (the tolerance is *floor(maxlag* $*0.05$*)=floor(*$22*0.05$*)*=1 and only one peak crosses the 90% confidence lines (the tolerance is *floor(maxlag* $*0.10$*)=floor(*$22*0.10$*)*=2. Also the number of peaks crossing the confidence lines of the partial autocorrelogram is within the statistical tolerance. In fact, we still have no peaks crossing the 95% confidence line (the tolerance is still *floor(maxlag* $*0.05$*)=1)* and only one peak crosses the 90% confidence line (the tolerance is still *floor(maxlag* $*0.10$*)=2)*. Therefore, we have visual evidence that the residuals have been generated by independent random sampling from the same distribution at the 90% confidence level.

We also consider the Ljung-Box (LB) test, which assumes the null hypothesis that the data set is generated by independent random sampling from the same distribution. We have

```
z <- Y_X_df$Res
n <- length(z)
maxlag <- ceiling(10*log10(n))
X_LB <- Box.test(z, lag=maxlag, type="Ljung-Box")
show(X_LB)
```

```
##
##  Box-Ljung test
##
## data:  z
## X-squared = 14.592, df = 22, p-value = 0.8792
```

The fourth and last step is to check whether the residuals are Gaussian distributed. In the positive case, the statistics of the summary will be solidly based. We start with building the Q-Q plots of the residuals. First, we add the sorted residuals to the *Y_X_df* data frame (recall that the residuals are necessarily sorted).

```
Y_X_df <- add_column(Y_X_df, Y_5=sort(Y_X_df$Res), .after="Res")
head(Y_X_df)
```

```
##   t        X         Y      Fit        Res       Y_5
## 1 1 8.026032 3.6347229 3.260169  0.3745542 -3.002905
## 2 2 6.732687 2.8021444 3.087663 -0.2855183 -2.916359
## 3 3 5.477368 3.4026390 2.920229  0.4824102 -2.768774
## 4 4 5.368712 3.8646020 2.905736  0.9588656 -2.569248
## 5 5 1.220284 0.2013109 2.352421 -2.1511096 -2.446065
## 6 6 8.043723 7.2487956 3.262528  3.9862673 -2.426864
```

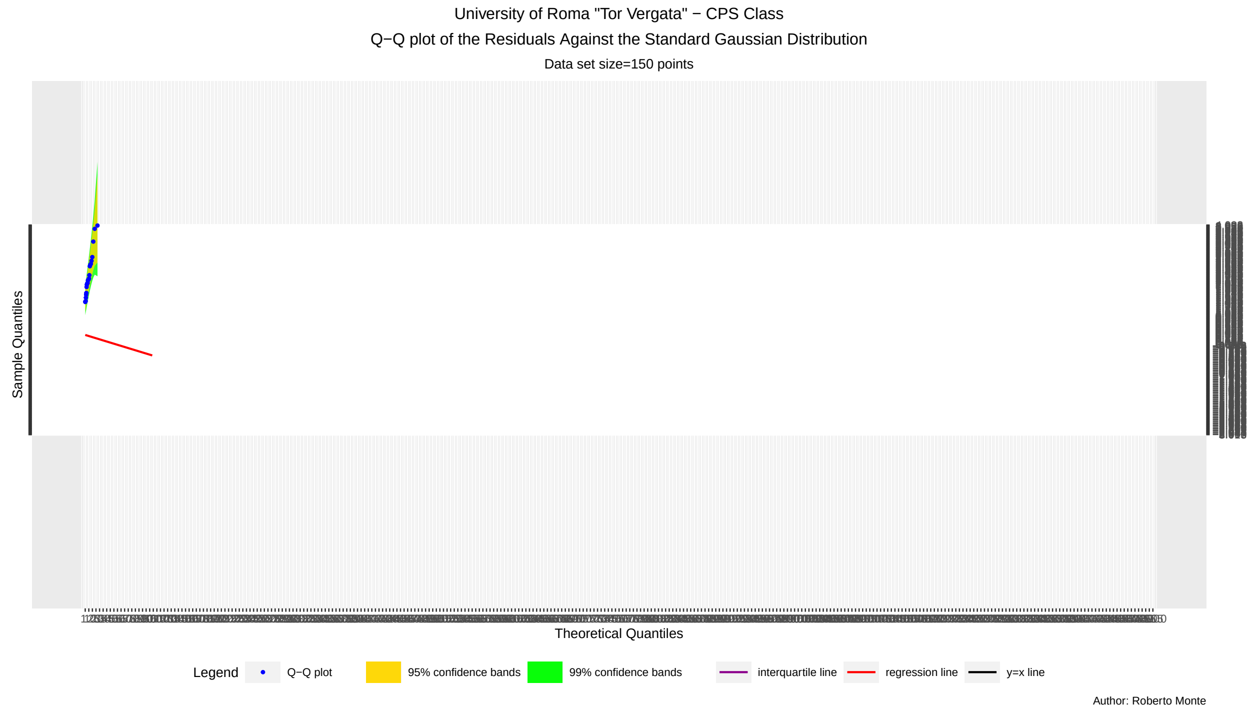
Second we draw the Q-Q plots.

The Q-Q plot of the residuals.

```
Data_df <- Y_X_df
n <- nrow(Data_df)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - CPS Class",
                             "Q-Q plot of the Residuals Against the Standard Gaussian Distribution"))
subtitle_content <- bquote(paste("Data set size=", .(n), " points"))
caption_content <- "Author: Roberto Monte"
distr <- "norm"
distr_pars <- list(mean=0, sd=1)
x_name <- bquote("Theoretical Quantiles")
y_name <- bquote("Sample Quantiles")
x_breaks_min <- floor(Data_df$t[1])
x_breaks_max <- ceiling(Data_df$t[n])
x_breaks <- seq(from=x_breaks_min, to=x_breaks_max, by=0.5)
x_labs <- format(x_breaks, scientific=FALSE)
x_lims <- c(x_breaks_min,x_breaks_max)
y_breaks_num <- length(x_breaks)
y_binwidth <- round((max(Data_df$Y_5)-min(Data_df$Y_5))/y_breaks_num, digits=3)
y_breaks_low <- floor((min(Data_df$Y_5)/y_binwidth))*y_binwidth
y_breaks_up <- ceiling((max(Data_df$Y_5)/y_binwidth))*y_binwidth
y_breaks <- c(round(seq(from=y_breaks_low, to=y_breaks_up, by=y_binwidth),3))
y_labs <- format(y_breaks, scientific=FALSE)
y_lims <- c(y_breaks_low, y_breaks_up)
y1_shape <- bquote("Q-Q plot")
y1_fill <- bquote("95% confidence bands")
y2_fill <- bquote("99% confidence bands")
col_1 <- bquote("interquartile line")
col_2 <- bquote("regression line")
col_3 <- bquote("y=x line")
leg_shape_labs <- y1_shape
leg_fill_labs <- c(y1_fill, y2_fill)
leg_col_labs <- c(col_1, col_2, col_3)
leg_shape_cols <- c("y1_shape" = 19)
leg_fill_cols <- c("y1_fill"="gold", "y2_fill"="green")
leg_col_cols <- c("col_1"="darkmagenta", "col_2"="red", "col_3"="black")
leg_shape_ord <- "y1_shape"
leg_fill_ord <- c("y1_fill", "y2_fill")
leg_col_ord <- c("col_1", "col_2", "col_3")
Res_QQ_plot <- ggplot(Data_df, aes(sample=Y_5)) +
  stat_qq_band(aes(fill="y2_fill"), distribution=distr, dparams=distr_pars, conf = 0.99) +
```

```
  stat_qq_band(aes(fill="y1_fill"), distribution=distr, dparams=distr_pars, conf = 0.95) +
  stat_qq_line(aes(colour="col_1"), distribution=distr, dparams=distr_pars)+
  stat_smooth(alpha=1, size=0.8, linetype="solid", aes(x=X, y=Y_5, colour="col_2"),
              method="lm" , formula=y~x, se=FALSE, fullrange=FALSE) +
  geom_segment(aes(x=X[1], xend=-X[1], y=X[1], yend=-X[1], colour="col_3"),
               size=0.8, linetype="solid", show.legend=FALSE) +
  stat_qq_point(aes(shape="y1_shape"), colour="blue", alpha=1, size=1.0,
                distribution=distr, dparams=distr_pars) +
  scale_x_continuous(name=x_name, breaks=x_breaks, label=x_labs, limits=x_lims) +
  scale_y_continuous(name=y_name, breaks=y_breaks, labels=NULL, limits=NULL,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  scale_shape_manual(name="Legend", labels=leg_shape_labs, values=leg_shape_cols, breaks=leg_shape_ord)
  scale_fill_manual(name="", labels=leg_fill_labs, values=leg_fill_cols, breaks=leg_fill_ord) +
  scale_colour_manual(name="", labels=leg_col_labs, values=leg_col_cols, breaks=leg_col_ord) +
  guides(shape=guide_legend(order=1), fill=guide_legend(order=2), colour=guide_legend(order=3)) +
  theme(plot.title=element_text(hjust=0.5), plot.subtitle=element_text(hjust=0.5),
        axis.text.x=element_text(angle=0, vjust=1),
        legend.key.width=unit(1.0,"cm"), legend.position="bottom")
plot(Res_QQ_plot)
```

```
## Warning: Removed 5 rows containing non-finite values (`stat_smooth()`).

## Warning: The following aesthetics were dropped during statistical transformation: sample
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a `group` aesthetic or to convert a numerical
##   variable into a factor?

## `geom_path()`: Each group consists of only one observation.
## i Do you need to adjust the group aesthetic?

## Warning: Removed 150 rows containing missing values (`geom_segment()`).
```



The inspection of the Q-Q plots shows that the interquartile line and the regression line are rather close to each other and the spread of the points of the Q-Q plots from the interquartile line seems to be even inside the 95% confidence band. Therefore we have visual evidence for Gaussianity of the residuals. Furthermore, since the pattern of the Q-Q plots are steeper than the straigth line $y = x$, the residuals are likely drawn from a Gaussian distribution which is more dispersed than the standard Gaussian distribution.

We consider also the normality tests for the residuals. The SW test

```
# Shapiro-Wilks (*SW*) test.
# library(stats)
z <- Y_X_df$Res
Res_SW <- shapiro.test(z)
show(Res_SW)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  z
## W = 0.98706, p-value = 0.1768
```

By applying the SW test we cannot reject the null hypothesis of Gaussianity for the residuals.

The DP test

```
# D'Agostino-Pearson (*DP*) test.
# library(fBasics)
z <- Y_X_df$Res
Res_DP <- dagoTest(z)
show(Res_DP)
```

```
##
```

```
## Title:
##  D'Agostino Normality Test
##
## Test Results:
##   STATISTIC:
##     Chi2 | Omnibus: 2.5768
##     Z3  | Skewness: 1.2367
##     Z4  | Kurtosis: -1.0233
##   P VALUE:
##     Omnibus  Test: 0.2757
##     Skewness Test: 0.2162
##     Kurtosis Test: 0.3061
```

By applying the *DP* test we cannot reject the null hypothesis of Gaussianity for the residuals.

The *AD* test

```r
# Anderson-Darling (*AD*) test.
# library(nortest)
z <- Y_X_df$Res
Res_AD <- ad.test(z)
show(Res_AD)
```

```
##
##  Anderson-Darling normality test
##
## data:  z
## A = 0.45097, p-value = 0.2711
```

By applying the *AD* test we cannot reject the null hypothesis of Gaussianity for the residuals.

The *JB* test

```r
# Jarque-Bera (*JB*) test.
# library(tseries)
z <- Y_X_df$Res
Res_JB <- jarque.bera.test(z)
show(Res_JB)
```

```
##
##  Jarque Bera Test
##
## data:  z
## X-squared = 2.4417, df = 2, p-value = 0.295
```

By applying the *JB* test we cannot reject the null hypothesis of Gaussianity for the residuals.

We have collected a highly significant evidence that the residuals have been generated by independent random sampling from a Gaussian distributions.

Therefore, we can assess the adequacy of the linear regression and the validity of the statistical results presented in the summary

Observe that according to our choice of the entries of the vector μ and the matrix A, given by

$$\mu \equiv \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} \equiv \begin{pmatrix} 5 \\ 3 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix} \equiv \begin{pmatrix} 1 & 2 \\ -1 & 1 \end{pmatrix}$$

we expect to obtain the linear regression function f, given by

$$f(x) = \mu_2 + \frac{a_{1,1}a_{2,1} + a_{1,2}a_{2,2}}{a_{1,1}^2 + a_{1,2}^2}(x - \mu_1) = 3 + \frac{-1+2}{1+4}(x-5) = 2 + \frac{1}{5}x.$$

Thus

$$\beta_0 = 2 \quad \text{and} \quad \beta_1 = \frac{1}{5} = 0.2.$$

In addition, since

$$Cov(X,Y) = \beta_1 \mathbf{D}^2[X],$$

we have

Note that

$$\mathbf{D}[U] = \sqrt{1.8} \approx 1.342$$

On the other hand, by regression analysis, we have obtained

$$\hat{\beta}_0 = 2.18966, \quad \hat{\beta}_1 = 0.13338$$

and

$$RSE = \hat{\sigma}_U = 1.482.$$

Can you build the confidence intervals of level $1-\beta_0 a$ for the values $\hat{\beta}_0$, $\hat{\beta}_1$, and $\hat{\sigma}_U$ and test the null hypoteses $H_0 : \beta_0 = 2$, $H_0 : \beta_1 = 0.2$, and $H_0 : \sigma_U = 1.342$ at the significance level α, for $\alpha = 0.1$, 0.05, and 0.01?

In the end, we draw the histogram of the residuals

Standard statistics on the residuals.

```
# Statistics of the data set Res
mode <- function(x) {
  d <- density(x)
  d$x[which.max(d$y)]
}
mu <- 0.00
sigma <- round(sqrt(1.8), digits=3)
Samp_Data <- Y_X_df$Res
Statistics=c("mean", "median", "mode", "min. (99.73%)", "max. (99.73%)", "1st quart.", "3rd quart.", "s

Teor_Stats <- rep(0,10)
Teor_Stats[1] <- as.character(formatC(mu, digits=3, format="f"))
Teor_Stats[2] <- as.character(formatC(qnorm(0.50, mean=mu, sd=sigma, lower.tail=TRUE, log.p=FALSE), dig
Teor_Stats[3] <- as.character(formatC(mu, digits=3, format="f"))
Teor_Stats[4] <- as.character(formatC(qnorm(0.00135, mean=mu, sd=sigma, lower.tail=TRUE, log.p=FALSE), 
Teor_Stats[5] <- as.character(formatC(qnorm(0.99865, mean=mu, sd=sigma, lower.tail=TRUE, log.p=FALSE), 
Teor_Stats[6] <- as.character(formatC(qnorm(0.25, mean=mu, sd=sigma, lower.tail=TRUE, log.p=FALSE), dig
Teor_Stats[7] <- as.character(formatC(qnorm(0.75, mean=mu, sd=sigma, lower.tail=TRUE, log.p=FALSE), dig
Teor_Stats[8] <- as.character(formatC(sigma, digits=3, format="f"))
Teor_Stats[9] <- as.character(formatC(0, digits=3, format="f"))
Teor_Stats[10] <- as.character(formatC(0, digits=3, format="f"))

Samp_Stats <- rep(0,10)
Samp_Stats[1] <- as.character(formatC(mean(Samp_Data), digits=3, format="f"))
```

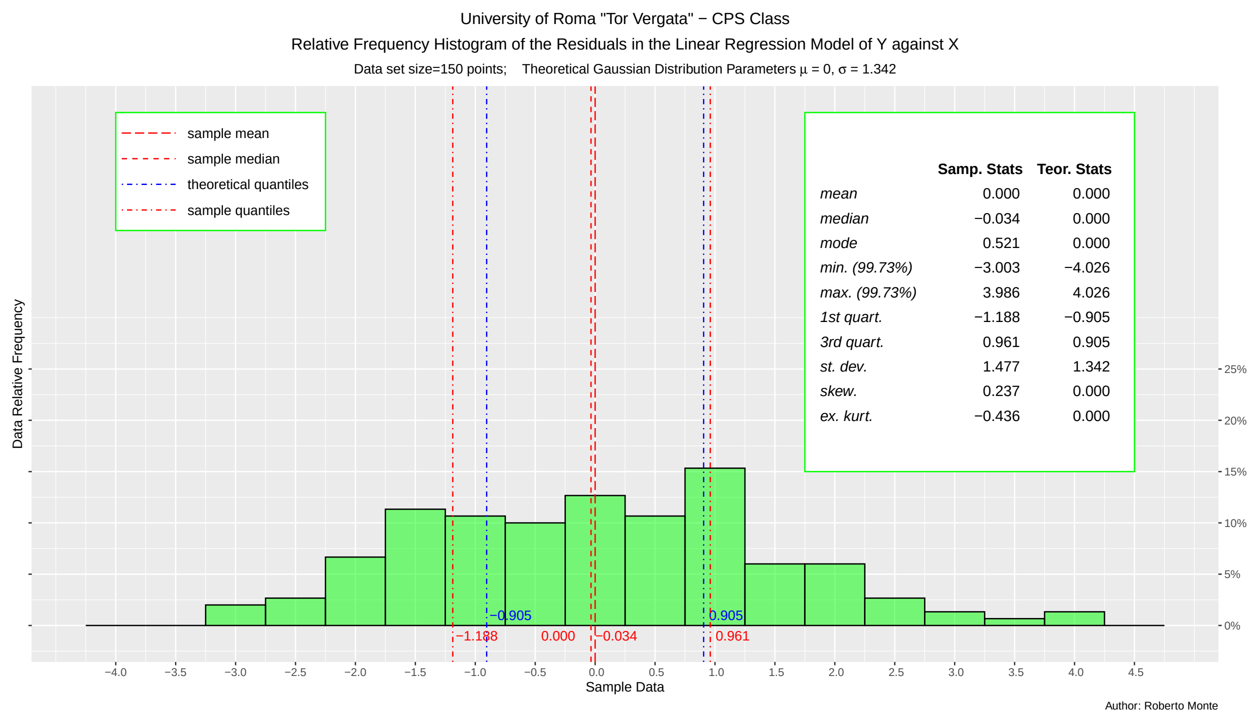
```
Samp_Stats[2] <- as.character(formatC(median(Samp_Data), digits=3, format="f"))
Samp_Stats[3] <- as.character(formatC(mode(Samp_Data), digits=3, format="f"))
Samp_Stats[4] <- as.character(formatC(min(Samp_Data), digits=3, format="f"))
Samp_Stats[5] <- as.character(formatC(max(Samp_Data), digits=3, format="f"))
Samp_Stats[6] <- as.character(formatC(quantile(Samp_Data,0.25), digits=3, format="f"))
Samp_Stats[7] <- as.character(formatC(quantile(Samp_Data,0.75), digits=3, format="f"))
Samp_Stats[8] <- as.character(formatC(sd(Samp_Data), digits=3, format="f"))
Samp_Stats[9] <- as.character(formatC(as.numeric(timeDate::skewness(Samp_Data, method="moment")), digit
Samp_Stats[10] <- as.character(formatC(as.numeric(timeDate::kurtosis(Samp_Data, method="excess")), digi

Table_Stats <- data.frame(Samp_Stats,Teor_Stats)
rownames(Table_Stats) <- Statistics
colnames(Table_Stats) <- c("Samp. Stats", "Teor. Stats")
```

Relative frequency and density histograms of the residuals.

The relative frequency histogram

```
# library(gridExtra)
#### Relative Frequency Histogram + Sample Statistics
Data_df <- Y_X_df
n <- nrow(Data_df)
mu <- 0.00
sigma <- round(sqrt(1.8), digits=3)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - CPS Class",
                             "Relative Frequency Histogram of the Residuals in the Linear Regression Mo
subtitle_content <- bquote(paste("Data set size=", .(n), " points;    Theoretical Gaussian Distribution
caption_content <- "Author: Roberto Monte"
# x_breaks_num <- ceiling(n^(1/2)) # Tukey & Mosteller square-root rule
# x_breaks_num <- ceiling(1+log2(n)) # Sturges rule
# x_breaks_num <- ceiling((2*n)^(1/3)) # Teller & Scott rice rule
# x_breaks_num <- 10
# x_binwidth <- round((max(Y_X_df$Res)-min(Y_X_df$Res))/x_breaks_num, digits=1)
x_binwidth <- 0.5
x_breaks_low <- floor((min(Y_X_df$Res)/x_binwidth))*x_binwidth
x_breaks_up <- ceiling((max(Y_X_df$Res)/x_binwidth))*x_binwidth
J <- 1
x_breaks <- c(seq(from=(x_breaks_low-J*x_binwidth), to=(x_breaks_up+J*x_binwidth), by=x_binwidth))
x_labs <- format(x_breaks, scientific=FALSE)
x_lims <- c(x_breaks[1],x_breaks[length(x_breaks)])
y_breaks <- seq(from=0, to=0.25, by=0.05)
y_labs <- format(percent(y_breaks), scientific=FALSE)
y_lims <- c(-0.010, 0.50)
tt3 <- ttheme_minimal(core=list(fg_params=list(hjust=1, x=0.90)),
                      rowhead=list(fg_params=list(hjust=0, x=0)))
Table_Stats_Grob <- tableGrob(Table_Stats, theme=tt3)
Res_rel_freq_hist <- ggplot(Data_df, aes(x=Res)) +
  geom_histogram(binwidth=x_binwidth , aes(y=stat(count)/sum(count)), color="black", fill="green", alph
  scale_x_continuous(name="Sample Data", breaks=x_breaks, labels=x_labs) +
  scale_y_continuous(name="Data Relative Frequency", breaks=y_breaks, labels=NULL, limits=y_lims,
                     sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
  ggtitle(title_content) +
  labs(subtitle=subtitle_content, caption=caption_content) +
  theme(plot.title=element_text(hjust=0.5),
```

```
        plot.subtitle=element_text(hjust=0.5),
        plot.caption=element_text(hjust=1.0)) +
  geom_vline(aes(xintercept=as.numeric(qnorm(0.25, mean=mu, sd=sigma, lower.tail=TRUE, log.p=FALSE))),
             colour="blue", linetype="dotdash", size=0.5) +
  annotate("text", x=as.numeric(qnorm(0.25, mean=mu, sd=sigma, lower.tail=TRUE, log.p=FALSE))
           +0.025, y=0.01, colour="blue",
           label=Teor_Stats[6], hjust=0) +
  geom_vline(aes(xintercept=as.numeric(qnorm(0.75, mean=mu, sd=sigma, lower.tail=TRUE, log.p=FALSE))),
             colour="blue", linetype="dotdash", size=0.5) +
  annotate("text", x=as.numeric(qnorm(0.75, mean=mu, sd=sigma, lower.tail=TRUE, log.p=FALSE))
           +0.045, y=0.01, colour="blue", label=Teor_Stats[7], hjust=0) +
  geom_vline(aes(xintercept=as.numeric(quantile(Samp_Data,0.25))),
             colour="red", linetype="dotdash", size=0.5) +
  annotate("text", x=as.numeric(quantile(Samp_Data,0.25))+0.025, y=-0.01, colour="red",
           label=Samp_Stats[6], hjust=0) +
  geom_vline(aes(xintercept=as.numeric(quantile(Samp_Data,0.75))),
             colour="red", linetype="dotdash", size=0.5) +
  annotate("text", x=as.numeric(quantile(Samp_Data,0.75))+0.045, y=-0.01, colour="red",
           label=Samp_Stats[7], hjust=0) +
  geom_vline(aes(xintercept=mean(Samp_Data)), colour="red", linetype="longdash", size=0.5) +
  annotate("text", x=mean(Samp_Data)-0.450, y=-0.01, colour="red",
           label=Samp_Stats[1], hjust=0) +
  geom_vline(aes(xintercept=median(Samp_Data)), colour="red", linetype="dashed", size=0.5) +
  annotate("text", x=median(Samp_Data)+0.035, y=-0.01, colour="red",
           label=Samp_Stats[2], hjust=0) +
  annotate("rect", xmin=(x_lims[2]-2.750), xmax=x_lims[2], ymin=(y_lims[2]-0.35), ymax=y_lims[2],
            colour="green", fill="white") +
  annotation_custom(Table_Stats_Grob, xmin=(x_lims[2]-2.50), xmax=(x_lims[2]-0.25),
                    ymin=(y_lims[2]-0.20), ymax=(y_lims[2]-0.15)) +
  annotate("rect", xmin=x_lims[1], xmax=(x_lims[1]+1.75), ymin=(y_lims[2]-0.115), ymax=y_lims[2],
           colour="green", fill="white") +
  annotate("segment", x=(x_lims[1]+0.05), xend=(x_lims[1]+0.50), y=(y_lims[2]-0.020), yend=(y_lims[2]-0
           colour="red", lty="longdash") +
  annotate("text", x=(x_lims[1]+0.60), y=(y_lims[2]-0.020), colour="black", label="sample mean", hjust=(
  annotate("segment", x=(x_lims[1]+0.05), xend=(x_lims[1]+0.50), y=(y_lims[2]-0.045), yend=(y_lims[2]-0
           colour="red", lty="dashed") +
  annotate("text", x=(x_lims[1]+0.60), y=(y_lims[2]-0.045), colour="black", label="sample median", hjus
  annotate("segment", x=(x_lims[1]+0.05), xend=(x_lims[1]+0.50), y=(y_lims[2]-0.070), yend=(y_lims[2]-0
           colour="blue", lty="dotdash") +
  annotate("text", x=(x_lims[1]+0.60), y=(y_lims[2]-0.070), colour="black", label="theoretical quantiles
  annotate("segment", x=(x_lims[1]+0.05), xend=(x_lims[1]+0.50), y=(y_lims[2]-0.095), yend=(y_lims[2]-0
           colour="red", lty="dotdash") +
  annotate("text", x=(x_lims[1]+0.60), y=(y_lims[2]-0.095), colour="black", label="sample quantiles", h
plot(Res_rel_freq_hist)
```

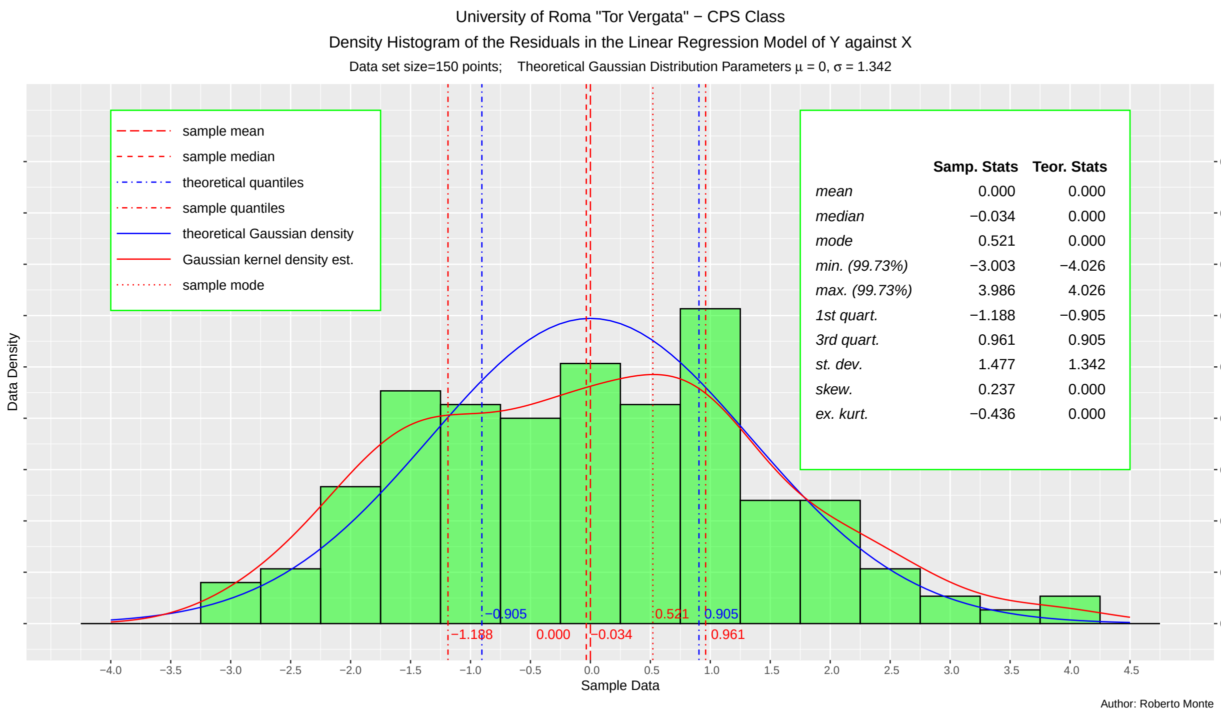


The density histogram of the residuals

```
# library(gridExtra)
#### Density Histogram + Sample Statistics + Density Kernel Estimation
Data_df <- Y_X_df
n <- nrow(Data_df)
mu <- 0.00
sigma <- round(sqrt(1.8), digits=3)
title_content <- bquote(atop("University of Roma \"Tor Vergata\" - CPS Class",
                             "Density Histogram of the Residuals in the Linear Regression Model of Y ag
subtitle_content <- bquote(paste("Data set size=", .(n), " points;    Theoretical Gaussian Distribution
caption_content <- "Author: Roberto Monte"
# x_breaks_num <- ceiling(n^(1/2)) # Tukey & Mosteller square-root rule
# x_breaks_num <- ceiling(1+log2(n)) # Sturges rule
# x_breaks_num <- ceiling((2*n)^(1/3)) # Teller & Scott rice rule
# x_breaks_num <- 10
# x_binwidth <- round((max(Y_X_df$Res)-min(Y_X_df$Res))/x_breaks_num, digits=1)
x_binwidth <- 0.5
x_breaks_low <- floor((min(Y_X_df$Res)/x_binwidth))*x_binwidth
x_breaks_up <- ceiling((max(Y_X_df$Res)/x_binwidth))*x_binwidth
J <- 1
x_breaks <- c(seq(from=(x_breaks_low-J*x_binwidth), to=(x_breaks_up+J*x_binwidth), by=x_binwidth))
x_labs <- format(x_breaks, scientific=FALSE)
x_lims <- c(x_breaks[1],x_breaks[length(x_breaks)])
y_breaks <- seq(from=0, to=0.45, by=0.05)
y_labs <- format(y_breaks, scientific=FALSE)
y_lims <- c(-0.010,0.50)
tt3 <- ttheme_minimal(core=list(fg_params=list(hjust=1, x=0.90)),
                      rowhead=list(fg_params=list(hjust=0, x=0)))
Table_Stats_Grob <- tableGrob(Table_Stats, theme=tt3)
Res_dens_hist <- ggplot(Data_df, aes(x=Res)) +
```

```
geom_histogram(binwidth=x_binwidth, aes(y=..density..), # binwidth=0.5,  # Density Histogram
               color="black", fill="green", alpha=0.5) +
scale_x_continuous(name="Sample Data", breaks=x_breaks, labels=x_labs) +
scale_y_continuous(name="Data Density", breaks=y_breaks, labels=NULL, limits=y_lims,
                   sec.axis=sec_axis(~., breaks=y_breaks, labels=y_labs)) +
ggtitle(title_content) +
labs(subtitle=subtitle_content, caption=caption_content) +
theme(plot.title=element_text(lineheight=0.6, face="bold", hjust=0.5),
      plot.subtitle=element_text(hjust= 0.5),
      plot.caption=element_text(hjust=1.0)) +
stat_function(fun=dnorm, colour="blue", args=list(mean=mu, sd=sigma)) +
geom_vline(aes(xintercept=as.numeric(qnorm(0.25, mean=mu, sd=sigma, lower.tail=TRUE, log.p=FALSE))),
           colour="blue", linetype="dotdash", size=0.5) +
annotate("text", x=as.numeric(qnorm(0.25, mean=mu, sd=sigma, lower.tail=TRUE, log.p=FALSE))
         +0.025, y=0.01, colour="blue",
         label=Teor_Stats[6], hjust=0) +
geom_vline(aes(xintercept=as.numeric(qnorm(0.75, mean=mu, sd=sigma, lower.tail=TRUE, log.p=FALSE))),
           colour="blue", linetype="dotdash", size=0.5) +
annotate("text", x=as.numeric(qnorm(0.75, mean=mu, sd=sigma, lower.tail=TRUE, log.p=FALSE))
         +0.045, y=0.01, colour="blue", label=Teor_Stats[7], hjust=0) +
geom_vline(aes(xintercept=as.numeric(quantile(Samp_Data,0.25))),
           colour="red", linetype="dotdash", size=0.5) +
annotate("text", x=as.numeric(quantile(Samp_Data,0.25))+0.025, y=-0.01, colour="red",
         label=Samp_Stats[6], hjust=0) +
geom_vline(aes(xintercept=as.numeric(quantile(Samp_Data,0.75))),
           colour="red", linetype="dotdash", size=0.5) +
annotate("text", x=as.numeric(quantile(Samp_Data,0.75))+0.045, y=-0.01, colour="red",
         label=Samp_Stats[7], hjust=0) +
geom_vline(aes(xintercept=mean(Samp_Data)), colour="red", linetype="longdash", size=0.5) +
annotate("text", x=mean(Samp_Data)-0.450, y=-0.01, colour="red",
         label=Samp_Stats[1], hjust=0) +
geom_vline(aes(xintercept=median(Samp_Data)), colour="red", linetype="dashed", size=0.5) +
annotate("text", x=median(Samp_Data)+0.035, y=-0.01, colour="red",
         label=Samp_Stats[2], hjust=0) +
geom_vline(aes(xintercept=mode(Samp_Data)), colour="red", linetype="dotted", size=0.5) +
annotate("text", x=mode(Samp_Data)+0.020, y=+0.01, colour="red",
         label=Samp_Stats[3], hjust=0) +
annotate("segment", x=(x_lims[1]+0.05), xend=(x_lims[1]+0.50), y=(y_lims[2]-0.070), yend=(y_lims[2]-0
         colour="blue", lty="dotdash") +
annotate("text", x=(x_lims[1]+0.60), y=(y_lims[2]-0.070), colour="black", label="theoretical quantile
annotate("segment", x=(x_lims[1]+0.05), xend=(x_lims[1]+0.50), y=(y_lims[2]-0.095), yend=(y_lims[2]-0
         colour="red", lty="dotdash") +
annotate("text", x=(x_lims[1]+0.60), y=(y_lims[2]-0.095), colour="black", label="sample quantiles", h
geom_density(alpha=.2, colour="red") +
annotate("rect", xmin=(x_lims[2]-2.75), xmax=x_lims[2], ymin=(y_lims[2]-0.35), ymax=y_lims[2],
          colour="green", fill="white") +
annotation_custom(Table_Stats_Grob, xmin=(x_lims[2]-2.50), xmax=(x_lims[2]-0.25),
                  ymin=(y_lims[2]-0.20), ymax=(y_lims[2]-0.15)) +
annotate("rect", xmin=x_lims[1], xmax=(x_lims[1]+2.25), ymin=(y_lims[2]-0.195), ymax=y_lims[2],
           colour="green", fill="white") +
annotate("segment", x=(x_lims[1]+0.05), xend=(x_lims[1]+0.50), y=(y_lims[2]-0.020), yend=(y_lims[2]-0
         colour="red", lty="longdash") +
annotate("text", x=(x_lims[1]+0.60), y=(y_lims[2]-0.020), colour="black", label="sample mean", hjust=
```

```
  annotate("segment", x=(x_lims[1]+0.05), xend=(x_lims[1]+0.50), y=(y_lims[2]-0.045), yend=(y_lims[2]-0
           colour="red", lty="dashed") +
  annotate("text", x=(x_lims[1]+0.60), y=(y_lims[2]-0.045), colour="black", label="sample median", hjus
  annotate("segment", x=(x_lims[1]+0.05), xend=(x_lims[1]+0.50), y=(y_lims[2]-0.070), yend=(y_lims[2]-0
           colour="blue", lty="dotdash") +
  annotate("text", x=(x_lims[1]+0.60), y=(y_lims[2]-0.070), colour="black", label="theoretical quantiles
  annotate("segment", x=(x_lims[1]+0.05), xend=(x_lims[1]+0.50), y=(y_lims[2]-0.095), yend=(y_lims[2]-0
           colour="red", lty="dotdash") +
  annotate("text", x=(x_lims[1]+0.60), y=(y_lims[2]-0.095), colour="black", label="sample quantiles", h
  annotate("segment", x=(x_lims[1]+0.05), xend=(x_lims[1]+0.50), y=(y_lims[2]-0.120), yend=(y_lims[2]-0
  annotate("text", x=(x_lims[1]+0.60), y=(y_lims[2]-0.120), colour="black", label="theoretical Gaussian
  annotate("segment", x=(x_lims[1]+0.05), xend=(x_lims[1]+0.50),  y=(y_lims[2]-0.145), yend=(y_lims[2]-
  annotate("text", x=(x_lims[1]+0.60), y=(y_lims[2]-0.145), colour="black", label="Gaussian kernel densi
  annotate("segment", x=(x_lims[1]+0.05), xend=(x_lims[1]+0.50),  y=(y_lims[2]-0.170), yend=(y_lims[2]-
  annotate("text", x=(x_lims[1]+0.60), y=(y_lims[2]-0.170), colour="black", label="sample mode", hjust=
plot(Res_dens_hist)
```




```
knitr::knit_exit()
```