Summer 2021

shutterplot: An R Package to Display All Summary Statistics of a Simple Linear Regression Model

Siddhanta Phuyal
DePauw University

Mamunur Rashid
DePauw University, mrashid@depauw.edu

Jyotirmoy Sarkar
Indiana University Purdue University Indianapolis

Recommended Citation

Phuyal, Siddhanta; Rashid, Mamunur; and Sarkar, Jyotirmoy, "shutterplot: An R Package to Display All Summary Statistics of a Simple Linear Regression Model" (2021). *Student Research*. 40.
https://scholarship.depauw.edu/studentresearchother/40

shutterplot: An R Package to Display All Summary Statistics of a Simple Linear Regression Model

Authors

Siddhanta Phuyal
DePauw University, Department of Mathematical Sciences
2 E. Hanna Street, Greencastle, IN 46135 USA
ORCiD: 0000-0002-0675-3033
siddhantaphuyal_2023@depauw.edu

Mamunur Rashid
DePauw University, Department of Mathematical Sciences
2 E. Hanna Street, Greencastle, IN 46135 USA
ORCiD: 0000-0001-8759-3803
mrashid@depauw.edu

Jyotirmoy Sarkar
Indiana University-Purdue University Indianapolis
Department of Mathematical Sciences
402 N. Blackford Street, Indianapolis, IN 46202 USA
ORCiD: 0000-0001-5002-5845
jsarkar@iupui.edu

Abstract

A shutterplot depicts all summary statistics in a simple linear regression model. Instead of reporting their numerical values, a shutterplot gives their visual representations. In this paper, we provide a step-by-step description of how the shutterplot package constructs a shutterplot and the options the user has in customizing it.

Keywords: Scatter plot, Regression lines, Correlation coefficient, Coefficient of determination, R, Shutterplot package.

AMC: 62J05,62-09

1. Introduction

Sarkar and Rashid (2020) proposed a shutterplot to depict all summary statistics in a simple linear regression model. Recall that for a single quantitative variable, the entire data are shown without distortion in a dot plot (see, for example, Wilkinson (1999)) and all summary statistics are presented in either a box plot (see, for example, Spear (1952) and Spear (1969)) or a summary seven plot (five-number-summary, mean, and standard deviation) Sarkar and Rashid (2020). On the other hand, for two quantitative variables, a scatter plot only shows their inter-relation, but not the other summary statistics. A shutterplot remedies this shortcoming: A shutter plot superimposes on a scatter plot several rectangles which together depict the means and the SDs of both variables, the two regression lines and the coefficient of correlation (r) and determination (r^2). Futhermore, the function shutter plot identifies potential x-outliers, y-outliers, and residual-outliers.

This article describes the following tasks as we proceed to construct a shutter plot:

(a) Visualizing Univariate data and their Summary Statistics.
(b) Visualizing Bivariate data and their Summary Statistics.

2. Visualizing univariate data and their summary statistics

When data are gathered on a quantitative variable measured on n observational units, we obtain n numbers $x_1, x_2, \ldots, x_n$. To visualize the entire dataset without distortion, one choice is to use the dotchart or stripchart function of the "graphics" package. The following R code can be used to produce a dot plot.

```
dev.new(width=12,height=10)
par(mai=c(4,2,.2,2)) # bottom, left, top, right
x= c(68.4,69.0,65.5,69.6,67.5,71.0,67.3,67.0,63.1,78.1,
     62.5,71.5,74.8,73.0,66.5,69.9,68.8,67.5,63.4,68.3)
x1=min(x)
x2=max(x)
stripchart(x,method="stack", pch = 20, cex=2.2, xlim=c(x1,x2),
           las=1,xaxs="i",yaxs="i",frame.plot = FALSE,
           ylab="", xlab="", xaxt="n",xpd=TRUE, col="black")
axis(1, pos=.96,at=c(seq(62,80,2)), padj=0,cex=1,xpd=T) # CHANGE VALUE
arrows(x1-1.5,.96,x2+3,.96, code = 2, xpd = TRUE, length=.10)
text(x2+4,.935, expression(italic(x)),xpd=T,cex=1.1)
```

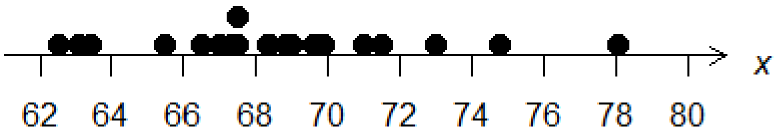


Figure 1: The unabridged data on one variable x are shown as a dot plot.

The display of entire data is reasonable when the sample size is small (say, $n < 50$). For larger values of n, oftentimes for ease of comprehension, instead of (or in addition to) displaying the entire data, it is preferable to display some summary statistics. A boxplot, also known as a box-and-whisker plot, depicts the five-number summary or the summary-5 statistics (the minimum, the three quartiles Q_1, Q_2, Q_3 which are also known as the 25th, the 50th, and the 75th percentiles, and the maximum).

The following R codes produce a box-and-whisker plot.

```
dev.new(width=10,height=10)
par(mar=c(1, 3, .6, 7))
height= c(68.4,69.0,65.5,69.6,67.5,71.0,67.3,67.0,63.1,78.1,
          62.5,71.5,74.8,73.0,66.5,69.9,68.8,67.5,63.4,68.3,
          68.8,65.6,64.1,63.6,72.5,61.7,64.8,70.1,65.1,67.1,
          64.4,64.1,64.7,63.3,68.1,65.6,74.2,72.8,63.4,67.5,
          67.9,63.8,68.5,63.5,68.5,68.8,67.0,74.8,70.7,82.0)
x1 <- min(height)
x2 <- max(height)
boxplot(height, horizontal = TRUE, frame.plot=FALSE, axes = FALSE, boxwex=0.2)
axis(1,pos=.85,at=c(seq(60,82,2)),xpd=T)
arrows(x1-3,.85,x2+5,.85, code = 2, xpd = TRUE, length=.10)
text(x2+6.5,.85, expression("height"),xpd=T,cex=1.1)
```

The parameter `horizontal = TRUE` is used to display the box plot in a horizontal orientation. The default value is FALSE. If `horizontal = TRUE` were omitted, then the plot would be shown vertically.

When the above codes are executed, the following plot is displayed.

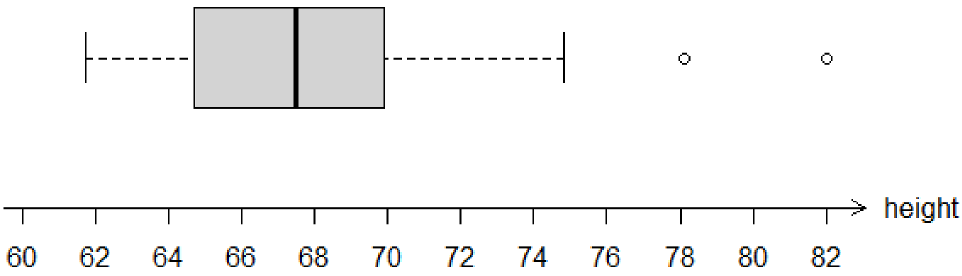


Figure 2: A horizontal box plot of height (inch) of 50 male students in a college freshman class. Unfortunately, there is no record of the sample size in the plot or the frequencies of the outliers.

In preparation for efficiently depicting the summary statistics of univariate data, we modify the standard boxplot of Figure 2 in three ways to construct Figure 3:

I. The length of the box (in horizontal direction) denotes the inter-quartile range, but the height of the box is non-informative. So, we replace the inter-quartile-box by a solid line segment enclosed within open and closed parentheses which denote the first and the third quartiles, respectively. We call this simplified diagram the five-number line.

II. On top of the five-number line, we draw a thick right-arrow whose tail is at the mean and arrowhead at the mean plus one SD. We call this augmented diagram the seven-number line, since it depicts the summary-7 statistics (that is, the summary-5 statistics, the mean, and the SD).

III. We print the sample size as a subtile, or to the right of the seven-number lines when multiple groups are compared in the same graph.

Remark 1: The sample size is often omitted from a summary. This may cause readers to give an undue importance to a small dataset or falsely conclude groups are comparable when, in fact, the sample sizes of the groups are drastically different. We strongly recommend declaring the sample size of each group, especially when multiple seven-number lines are drawn on the same graph.

The function summary7plot of package "shutterplot" depicts the seven-number line.

The R code for the function summary7plot is:

```
summary7plot <- function(x){
  h<- 1.95
  x<-na.omit(x)
  y<-rep(h,length(x))
  dev.new(width =12, height=10)
  plot(x,y, main=paste("Seven Number Plot","\n","(non-missing n=",length(x),")"),
       yaxt="none", xlab="", ylab="", type="n",
       ylim=c(0,2),xlim=c(min(x),(max(x)*5-min(x))/4),
       frame.plot="F",axes=FALSE)
  x1 <- as.integer(min(x))
  x2 <- as.integer(max(x))
  axis(1, pos=1.7,at=c(seq(x1-2,x2+2,2)), padj=0,cex=1,xpd=TRUE) # CHANGE VALUE
  arrows(x1-2,1.7,x2+2,1.7, code = 2, xpd = TRUE, length=.10)
  segments(x0=min(x),y0=h, x1= max(x), y1 =h, lty="dashed")
  segments(x0=quantile(x,0.25), y0 =h, x1= quantile(x,0.75), y1=h, lty="solid")
  text(quantile(x,0.75),h,')')
  text(quantile(x,0.25),h,'(')
  text(min(x),h,'|')
  text(max(x),h,'|')
  text(median(x),h,'|')
  sdeviation = sqrt(var(x))
  finalposition <- mean(x)+ sdeviation
  arrows(x0=mean(x),y0=h,x1=finalposition,y1=h, length="0.15", lty="solid", lwd=2.8)
}
```

The function takes one parameter “x” which is the data for the quantitative variable. The function `summary7plot` depicts the `min(x)`, the first quartile, the second quartile or the `median(x)`, the third quartile, the `mean(x)`, the `sd(x)`, and the `max(x)`. The starting tail end of the arrow denotes the `mean(x)` and the length of the arrow denotes the `sd(x)`. The function also prints the non-missing number of observations as a subtitle at the top of the plot.

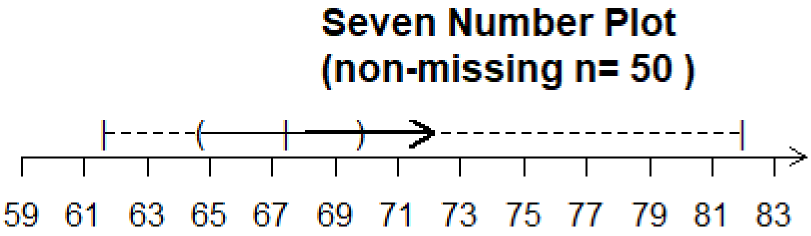


Figure 3: A horizontal summary-7 plot of height (inch) of 50 male students in a college freshman class. The sample size is recorded in the subtitle.

3. Visualizing bivariate data and their summary statistics

Suppose that we have n observational units on each of which two quantitative variables are measured. We can tabulate the n pairs of numbers $\{(x_1, y_1),\ (x_2, y_2), \ldots,\ (x_n, y_n)\}$ as raw data. Again, if the order in which the measurements are collected is irrelevant, we may sort the data in ascending (or descending) order with respect to either variable x or variable y. (There being no natural bivariate ordering, the pairs cannot be sorted simultaneously.)

If one wishes to visualize the entire dataset without distortion, one can use the function `plot` of the package `graphics`. Usually, the independent variable is plotted in the horizontal direction and the dependent variable in the vertical direction.

The following R codes produce the scatter plot:

```
data1<- c(68.4,69.0,65.5,69.6,67.5,71.0,67.3,67.0,63.1,78.1,
          62.5,71.5,74.8,73.0,66.5,69.9,68.8,67.5,63.4,68.3,
          68.8,65.6,64.1,63.6,72.5,61.7,64.8,70.1,65.1,67.1,
          64.4,64.1,64.7,63.3,68.1,65.6,74.2,72.8,63.4,67.5,
          67.9,63.8,68.5,63.5,68.5,68.8,67.0,74.8,70.7,82.0)

data2<- c(183.9,185.8,171.3,183.9,196.9,181.8,167.3,177.4,
          173.5,192.0,160.2,179.7,194.0,185.6,169.9,180.2,
          181.0,175.8,172.6,173.9,189.1,182.6,185.4,172.4,
          182.0,161.7,140.8,183.6,173.1,180.5,174.4,158.9,
          171.2,169.2,177.2,174.2,163.4,180.2,173.4,179.1,
          172.9,173.9,171.0,171.4,169.6,182.3,172.8,187.0,
          186.9,206.0)
dev.new(width =12, height=10)
plot(data1,data2,frame.plot = FALSE)
```

While a scatter plot depicts the bivariate data simultaneously, it does not depict any summary statistics. When the scatter plot indicates that the points are hovering around a line, perhaps after suitable transformations, a least-squares regression model is suitable to study the linear dependence between the two variables. Under this model, one variable is a linear function of the other variable, plus a random error. In such a case, sometimes the scatter plot is superimposed with the least-squares regression line (of y on x) given by

$$\hat{y} = a + bx = \left(\bar{y} - r\frac{s_y}{s_x}\bar{x}\right) + r\frac{s_y}{s_x}x = \bar{y} + r\frac{s_y}{s_x}(x - \bar{x})$$

which is known to pass through the mean vector $I = (\bar{x},\ \bar{y})$, though this point may or may not be marked on the diagram. Very rarely, also shown is the other least-squares regression line (of x on y) given by

$$\hat{x} = c + dy = \left(\bar{x} - r\frac{s_x}{s_y}\bar{y}\right) + r\frac{s_x}{s_y}y = \bar{x} + r\frac{s_x}{s_y}(y - \bar{y})$$

in which case the point of intersection of these regression lines is the mean vector.

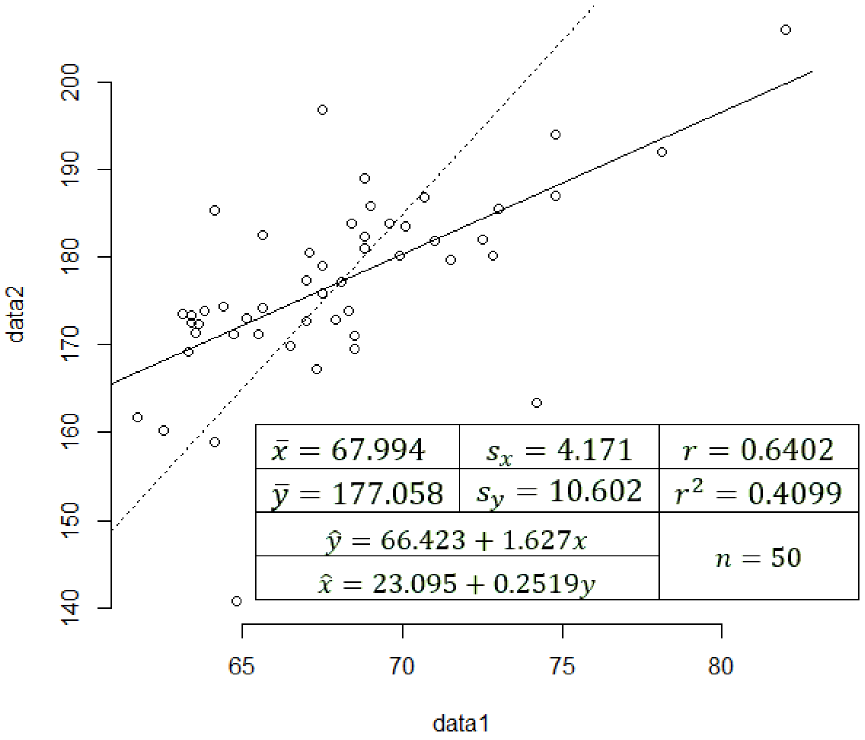
There are many ways to draw the regression lines in R. One easy way is to use the `abline` function along with the `lm` function, which belong to packages `graphics` and `stats`, respectively. The following R code can be used to produce a regression line of y on x in the plot.

```
abline(lm(data2~data1))
```

For producing the regression line of x on y, the following R codes can be used.

```
fit <- lm(data1~data2)
xonygradient <- 1/fit$coefficient[[2]]
xonyintercept <- (0-fit$coefficient[[1]])*xonygradient
abline(xonyintercept,xonygradient,lty=3)
```

If all the above R codes, along with `plot()`, are run together, we obtain the following scatter plot superimposed by the regression lines of y on x and x on y.



$\bar{x} = 67.994$	$s_x = 4.171$	$r = 0.6402$
$\bar{y} = 177.058$	$s_y = 10.602$	$r^2 = 0.4099$
$\hat{y} = 66.423 + 1.627x$		$n = 50$
$\hat{x} = 23.095 + 0.2519y$		

Figure 4: The two regression lines are superimposed on a scatter plot of height (inch) and weight (pound) of 50 male students in a college freshman class and these lines intersect at the mean vector. The inset prints the other summary statistics.

As shown in Figure 4, the other bivariate summary statistics are usually not depicted in a scatter plot. Instead, their numerical values can be printed either in an inset (as done in Figure 4) or in the caption or in the accompanying text. How can we visualize most of these univariate and bivariate summary statistics printed in the inset of Figure 4? The answer to this question is the main contribution of the pakage `shutterplot`.

4. Shutterplot

The shutter plot is constructed by the function shutterplot in the package "shutterplot". The package already invokes other existing packages such as `graphics`, `grDevices`, and `stats`.

1. The first step in depicting a shutter plot is to omit the NA values from the data. The package constructs a data frame, and drops all (x, y) pairs for which either x or y or both values are missing. The new dataset with non-missing values of both variables is stored back into x and y vectors. The R code for omitting NA values is:

```
creatingDataFrame<- data.frame(x=x,y=y)
cleanDataFrame<- na.omit(creatingDataFrame)
x<- cleanDataFrame$x
y<- cleanDataFrame$y
```

After omitting the NA values, the package checks whether (the absolute value of) the correlation between the two variables is less than 0.1. If so, the shutter plot is not going to be beneficial. As such, a message is printed on the console to alert the user. On the other hand, if correlation exceeds 0.1 (in absolute value), then a scatter plot is drawn using the plot function together with its optional arguments:

```
plot(x,y,xlab=xlab, ylab=ylab,las=1,main=main,
    ylim=c(min(y)-wspace*(max(y)-min(y)),max(y)+wspace*(max(y)-min(y))),
    xlim=c(min(x)-wspace*(max(x)-min(x)),max(x)+wspace*(max(x)-min(x))),frame.plot="F",
    pch=pch, cex=cex, col = colOfPoints,xaxs= "i",yaxs="i")
```

The package gives the user options to choose a suitable title for the plot, labels for the independent variable and the dependent variable. The default names for the plot, the independent variable and the dependent variable are “Shutter Plot”, “x”, and “y” respectively. However, the user can change these arguments. For example, if they want to change the name of the independent variable to “Weight”, they should pass `xlab = "Weight"` in the shutterplot function. The package gives the user a choice to change the `las` value as needed.

The package also lets the user decide the amount of white space they want in their plot. The argument `wspace` in the function shutterplot determines the amount of white space to the left, right, top, and bottom side of the plot. It is used while calculating the `xlim` and `ylim` of the plot. The default value is 0.1, which means 10 percent of the range of each variable is shown as blank space. For example, if the `wspace = 0.1`, `min(x) = 50`, and `max(x) = 200`, then `xlim` ranges from 35 to 215. The `ylim` is also calculated in a similar way.

The function `shutterplot` gives the user options to change the size, color, and shape of the scatter points. The user can change the color of the points using the argument `colOfPoints`. The default color of the points is “grey68” which the user can change by passing a different color; For example, `colOfPoints = "red"`. The user can change the shape of the points using the argument `pch`. The default shape is 20, which is a diamond; however, the user can change it by passing a different value. The size of the points can also be changed via the argument `cex`. The default size is 0.7, but the user can change it as they desire. All these arguments are made available to the user through the function `shutterplot`.

2. The second step is to superimpose the regression lines on the scatter plot. For the regression line of y on x, the package uses the functions `abline` and `lm`; for the regression line of x on y, the package calculates the y-intercept and gradient and then, uses the function `abline`. The R codes to produce the regression lines are:

```
# regression line of y on x
    fit <- lm(y~x)
    yonxintercept <- fit$coefficient[[1]]
    yonxgradient <- fit$coefficient[[2]]
    abline(yonxintercept,yonxgradient)

#regression line of x on y
    fit <- lm(x~y)
    xonygradient <- 1/fit$coefficient[[2]]
    xonyintercept <- (0-fit$coefficient[[1]])*xonygradient
    abline(xonyintercept,xonygradient, lty=3)
```

3. After superimposing the regression lines, the package draws on the plot an outlier box which contains the central data and declares points outside the box as x-outliers or y-outliers or both.

```
#lines for the outliers box
    vec1<- c(mean(x)-mult*sd(x),mean(x)+mult*sd(x),
    mean(x)+mult*sd(x),mean(x)-mult*sd(x),mean(x)-mult*sd(x))
    vec2<- c(mean(y)-mult*sd(y),mean(y)-mult*sd(y),
    mean(y)+mult*sd(y),mean(y)+mult*sd(y),mean(y)-mult*sd(y))
    lines(vec1,vec2,lty=2, col="#7a7a7a")
```

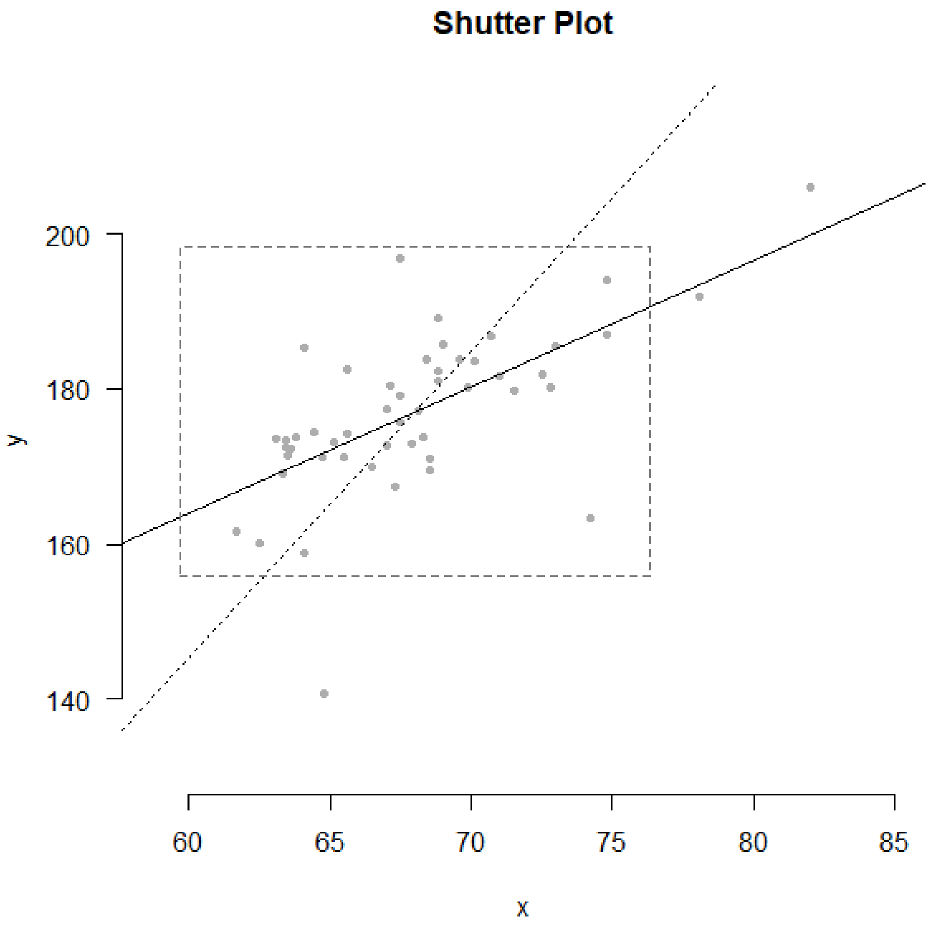


Figure 5: Shutter Plot diagram with the regression lines and outlier box. Any point outside the box is either an x outlier, or a y outlier or both.

4. Continuing on the above figure, we draw the univariate seven-number summary lines—not in the margins, but parallel to the respective axes—so that they intersect at $I = (\bar{x},\ \bar{y})$, the bivariate mean.

5. If the correlation is positive, then in the north-east quarter with respect to the relocated seven-number lines, we draw a rectangle that measures s_x in the horizontal direction and s_y in the vertical direction; and we call it the **bivariate-SD-rectangle**, and its diagonal through the bivariate mean I, which when extended becomes a diagonal of the outlier box, we call the **bivariate-SD-line** (dotted). (If the correlation coefficient is negative, we draw the bivariate-SD-rectangle in the south-east quarter. See an example in the last section.) Once the bivariate-SD-rectangle is drawn, there is no need for the univariate SD-arrows; and as such we eliminate them. The R codes for superimposing the bivariate-SD-rectangle and bivariate-SD-line are:

```
#lines for the sd box
```

```
  vec3<- c(mean(x),min(max(x),mean(x)+sd(x)),min(max(x),mean(x)+sd(x)),mean(x),mean(x))
  vec4<- c(mean(y),mean(y),min(max(y),mean(y)+sign(cor(x,y))* sd(y)),
        min(max(y),mean(y)+sign(cor(x,y))*sd(y)),mean(y))
  lines(vec3,vec4,lty=1,col="#7a7a7a")

#sd-line
    segments(x0 = mean(x)-mult*sign(cor(x,y))*sd(x),
    y0 = mean(y)-mult*sd(y),
    x1=mean(x)+mult*sign(cor(x,y))*sd(x),
    y1=mean(y)+mult*sd(y),lty= 2)
```

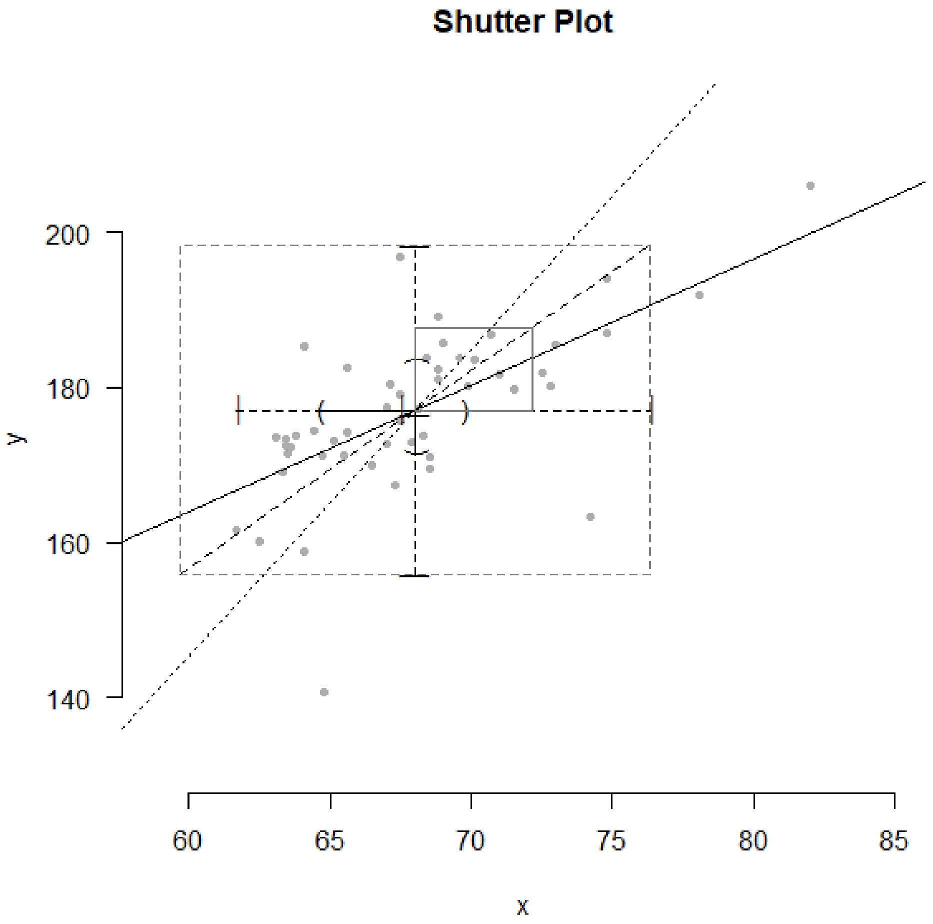


Figure 6: Shutter Plot diagram depicting outlier box, regression lines, horizontal and vertical lines passing through means, SD-rectangle, and SD-line

6. Next, we split the bivariate-SD-rectangle by a horizontal line at a vertical distance $r \cdot s_y$ from the bottom edge; and call the lower part the $\hat{y}$**-rectangle**; and its diagonal through the bivariate mean I is indeed the **least squares regression line of y on x** or simply the $\hat{y}$**-line**. Thus, r is the ratio of heights of $\hat{y}$-rectangle and bivariate-SD-rectangle. We make bold the $r^2 \cdot s_x$ portion of this horizontal line starting from the left edge and terminating at the $\hat{x}$**-line** to be explained in the next paragraph.

If through the point where the upper edge of the $\hat{y}$-rectangle intersects the bivariate-SD-line we draw a vertical line, then we split the bivariate-SD-rectangle a second time by a vertical line at a distance $r \cdot\ s_x$ from the left edge; and call the left part the $\hat{x}$**-rectangle**; and its diagonal through the bivariate mean is indeed the **least squares regression line of x on y** or simply the $\hat{x}$**-line**. Thus, r is also the ratio of widths of $\hat{x}$-rectangle and bivariate-SD-rectangle. We make bold the $r^2 \cdot\ s_y$ portion of the vertical line starting from the bottom edge and terminating at the $\hat{y}$**-line** already explained in the previous paragraph.

The R codes for splitting the bivariate-SD-box by superimposing a horizontal and a vertical line are:

```
#dividing SD Box using vertical line
    segments(x0 = (mean(x)+r*sign(cor(x,y))*sd(x)),y0 = mean(y),
             x1=(mean(x)+r*sign(cor(x,y))*sd(x)),
             y1=min(max(y),mean(y)+sign(cor(x,y))*sd(y)),lty=1,col="#7a7a7a")

#dividing SD Box using a horizontal line
    segments(x0 = mean(x),y0 = mean(y)+r*sd(y),
             x1=min(max(x),mean(x)+sd(x)),
             y1= mean(y)+r*sd(y), lty=1, col="#7a7a7a")

#bold lines in the inner rectangle
    segments(x0 = (mean(x)+r*sign(cor(x,y))*sd(x)),y0 = mean(y),
             x1=(mean(x)+r*sign(cor(x,y))*sd(x)),
             y1=mean(y)+r*sign(cor(x,y))*r*(min(max(y),mean(y)+sd(y))-mean(y)),lty=1,lwd=2)
    segments(x0 = mean(x),y0 = mean(y)+r*sd(y),
             x1=mean(x)+r*r*(min(max(x),mean(x)+sd(x))-mean(x)),
             y1= mean(y)+r*sd(y), lty=1,lwd=2)
```

The overlap between the $\hat{x}$-rectangle and the $\hat{y}$-rectangle (which are both bottom aligned and left aligned) is, of course, another rectangle, which we shade and call the **determination-rectangle**, since its area as a fraction of the area of the SD-rectangle is indeed the coefficient of determination r^2.

The R code for shading the determination rectangle is:

```
#coloring the determination rectangle
rect(xleft=mean(x), xright=(mean(x)+r*sign(cor(x,y))*sd(x)),
  ybottom=mean(y),
  ytop=mean(y)+r*(min(max(y),mean(y)+sd(y))-mean(y)),
  border =NA,col= rgb(186, 188, 191, maxColorValue = 255,
  alpha=76.5))
```

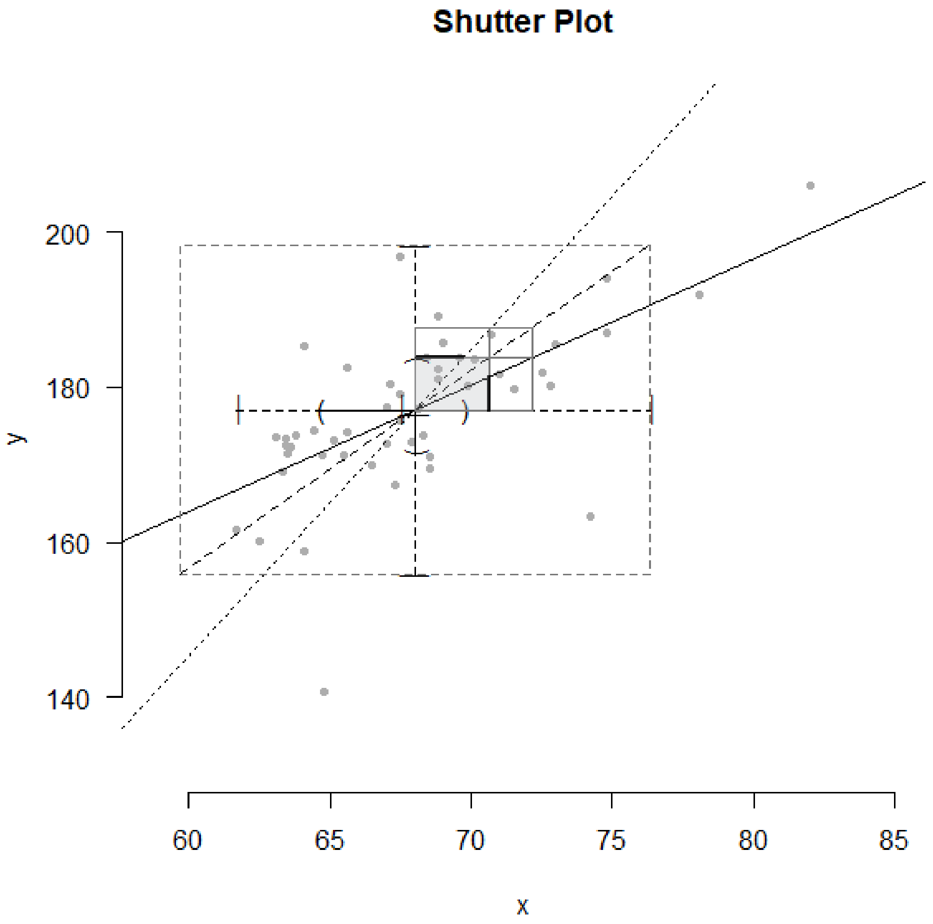


Figure 7: Shutter Plot diagram depicting the shaded determination rectangle.

7. Lastly, the prediction boundaries may be calculated and superimposed on the plot. However, the user can supress the boundaries by setting the argument `regbound=FALSE` available via the function `shutterplot`. The argument `regbound` is logical; the default value for `regbound` is TRUE, and the blue prediction boundaries are shown in the plot if TRUE value is passed.

The argument `alpha`, available via the function `shutterplot`, is the level of significance for the prediction boundaries. The default value is 0.05, which chooses the 97.5th percentile of a t-distribution with $(n-2)$ degrees of freedom for t* in the formula for calculating the prediction boundaries:

$$\bar{y} + r\frac{s_y}{s_x}(x-\bar{x}) \mp t^* \sqrt{1 + \frac{1}{n} + \frac{\left(\frac{x-\bar{x}}{s_x}\right)^2}{n-1}} \cdot \sqrt{1-r^2} \cdot s_y$$

The R codes used to calculate the prediction boundaries are:

```
#regboundaries
  sdofy<-sd(y)
  sdofx<-sd(x)
  lengthofx<-length(x)
```

```
regbound1 <-function(x) {
return(meanofy+(r*(sdofy/sdofx)*(x-meanofx))+t*
  ((1+(1/lengthofx)+
  ((((x-meanofx)/sdofx)^2)/(lengthofx-1)))^0.5)
  *sdofy*((1-(r^2))^0.5))
}

regbound2 <-function(x){
return(meanofy+(r*(sdofy/sdofx)*(x-meanofx))-t*
  ((1+(1/lengthofx)+
  ((((x-meanofx)/sdofx)^2)/(lengthofx-1)))^0.5)
  *sdofy*((1-(r^2))^0.5))
}
```

The function `regbound1` returns the values for the upper boundary, and the function `regbound2` returns the values for the lower boundary. The values returned by `regbound1` and `regbound2` are used with the `curve` function to superimpose the two prediction boundary curves on the plot.

The R code for drawing the prediction boundaries is:

```
if(regbound == TRUE){
curve(regbound1(x),add=TRUE,type="l",col= "blue",lty=2)
curve(regbound2(x),add=TRUE,type="l",col= "blue",lty=2)
}
```

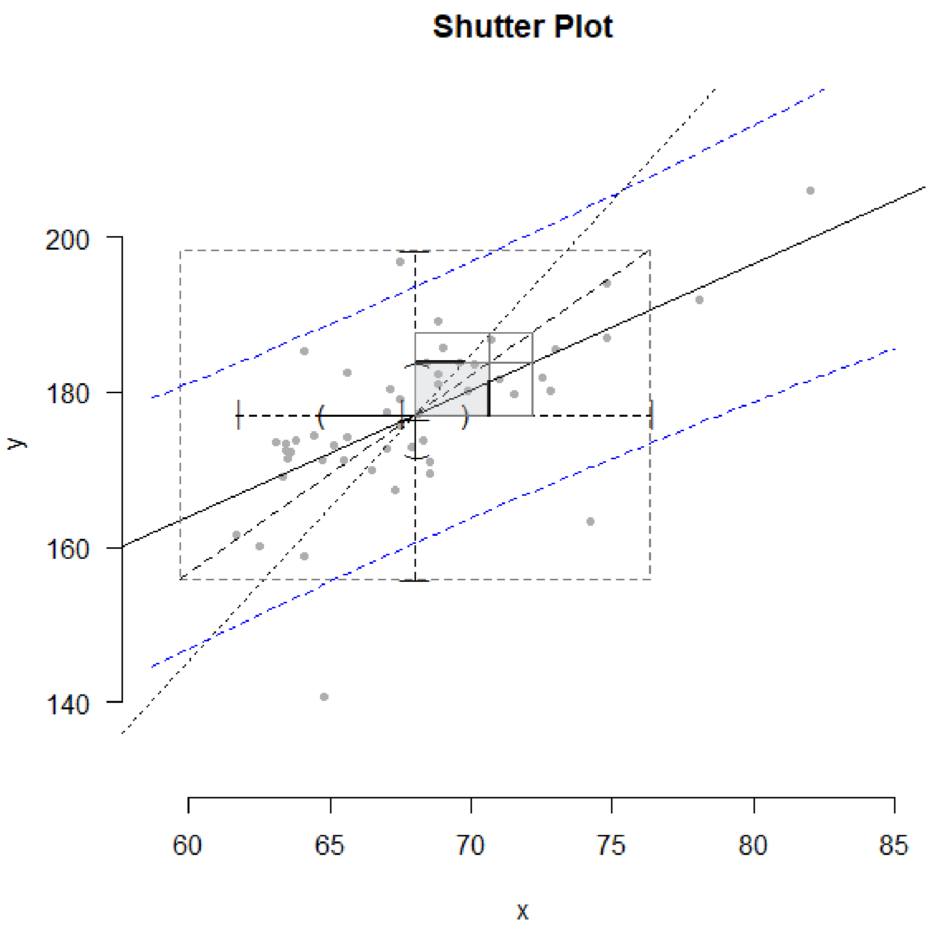


Figure 8: Shutter Plot diagram showing prediction boundary lines along with the previous information.

It is worth pointing out that although the blue prediction boundaries appear linear to the naked eye, they are indeed non-linear curves. Any scatter point that falls outside the prediction boundaries but inside the outlier box are called residual-outliers or regression outliers or simply R-outliers; the “shutterplot” package circles such points in red color. However, if the user does not want to circle the R-outliers, the package provides the user an option to reset the argument `regOutliers=FALSE` as was done in Figure 8. But in Figure 9, we use the default option. It is a logical argument, and its default value is TRUE.

The R codes for circling the R-outliers are:

```
# for r outliers:
upperlimit<-function(x,y){
  d<-mean(y)
  e<-(r*(sd(y)/sd(x))*(x-mean(x)))
  f<- t*((1+(1/length(x))+((((x-mean(x))/sd(x))^2)/(length(x)-1)))^0.5)
    *sd(y)*((1-(r*r))^0.5)
  return(d+e+f)
}
lowerlimit<-function(x,y){
  d<-mean(y)
  e<-(r*(sd(y)/sd(x))*(x-mean(x)))
```

```
    f<- t*((1+(1/length(x))+((((x-mean(x))/sd(x))^2)/(length(x)-1)))^0.5)
      *sd(y)*((1-(r*r))^0.5)
    return(d+e-f)
  }
  if((regbound == TRUE)&(regOutliers==TRUE)){
    points(x[(y>upperlimit(x,y))&(y<(mean(y)+mult*sd(y)))&
      (x>mean(x)-mult*sd(x))&(x<mean(x)+mult*sd(x))],
      y[(y>upperlimit(x,y))&(y<(mean(y)+mult*sd(y)))
      &(x>mean(x)-mult*sd(x))&(x<mean(x)+mult*sd(x))],pch = 1,col="red")
    points(x[(y<lowerlimit(x,y))&(y>(mean(y)-mult*sd(y)))&
      (x>mean(x)-mult*sd(x))&(x<mean(x)+mult*sd(x))],y[(y<lowerlimit(x,y))
      &(y>(mean(y)-mult*sd(y)))&(x>mean(x)-mult*sd(x))
      &(x<mean(x)+mult*sd(x))],pch = 1,col="red")
  }
```

8. Finally, the sample size is printed on the shutter plot unless the user suppresses it, and its location is determined according to the sign of the correlation. If the correlation is positive, the sample size is printed on the left side just above the top edge of the outlier box, and if the correlation is negative, the sample size is printed on the right side of the top edge of the outlier box. The user, through the argument `locationOfnStar`, can move the location of the sample size along a vertical line. The default value of the `locationOfnStar` is 1. The package gives the user an option to pass on any decimal value in the interval [-1, 1], with -1 denoting the bottom edge of the outlier box.

However, if the user does not want to superimpose the sample size, the package gives the user an option to remove the sample size through the argument `nprint=FALSE`; it is a logical argument; and its default value is TRUE.

```
# for printing the number of samples:
  if (nprint==TRUE){
  text((mean(x)-sign(cor(x,y))*1.5*sd(x)),
    mean(y)+locationOfnStar*(0.5+mult)*sd(y),
    paste("( n* =",length(x),")"),col="brown")
  }
```

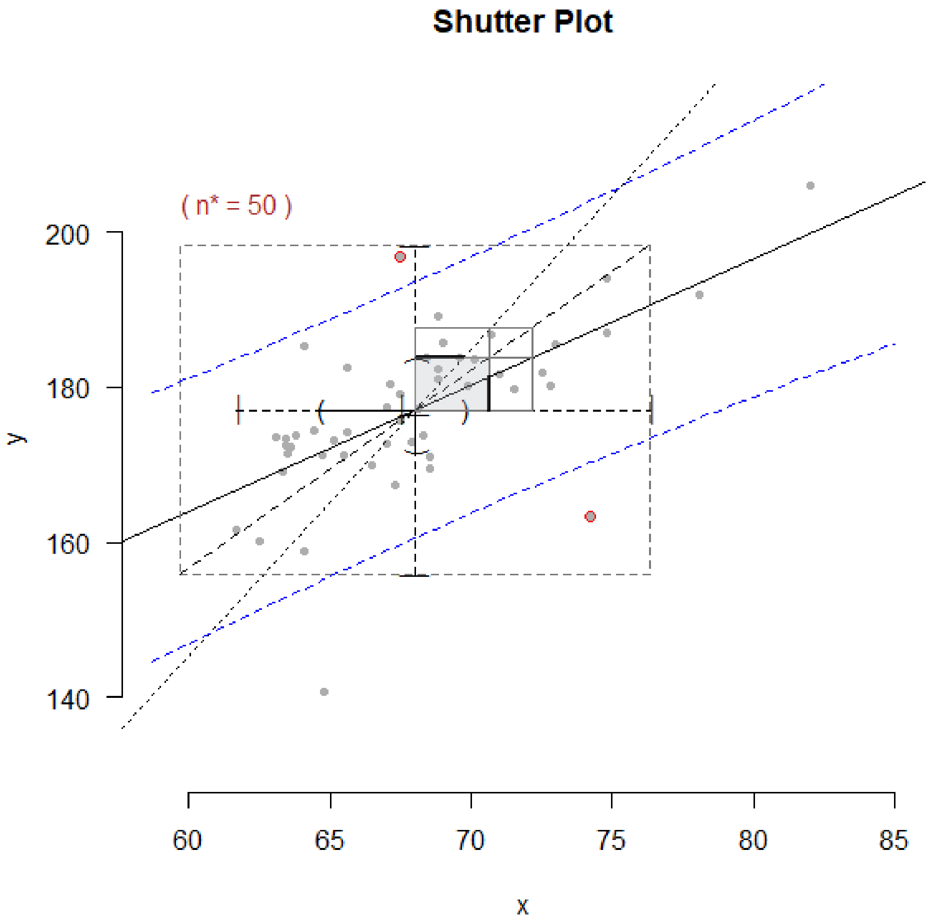


Figure 9: Final Shutter Plot diagram with the regression outliers circled and the number of non-missing samples printed.

5. Summary

A shutterplot shows in a single diagram all relevant statistics in a simple linear regression model. With some experience, the user can guess well the numerical values of these statistics from the diagram itself.

We show one more example when the correlation coefficient is negative. We leave it as an exercise to the reader to reproduce the Figure10 using the following data on body weight (pound) of and volume of air exhaled (ml) by 33 girls on the high school marching band:

```
weight<-c(
  86, 112, 147, 132, 104, 106, 124, 113, 110, 114, 122,
 168, 118, 130, 118, 104, 112, 129, 109, 127, 139, 127,
 150, 113, 113, 106, 120, 110,  92, 114, 129, 110, 128)

volume<-c(
 178, 185, 167, 172, 175, 174, 163, 172, 175, 172, 173,
 157, 173, 170, 173, 191, 171, 166, 177, 172, 172, 170,
 164, 174, 175, 177, 174, 174, 181, 176, 174, 178, 182)
```

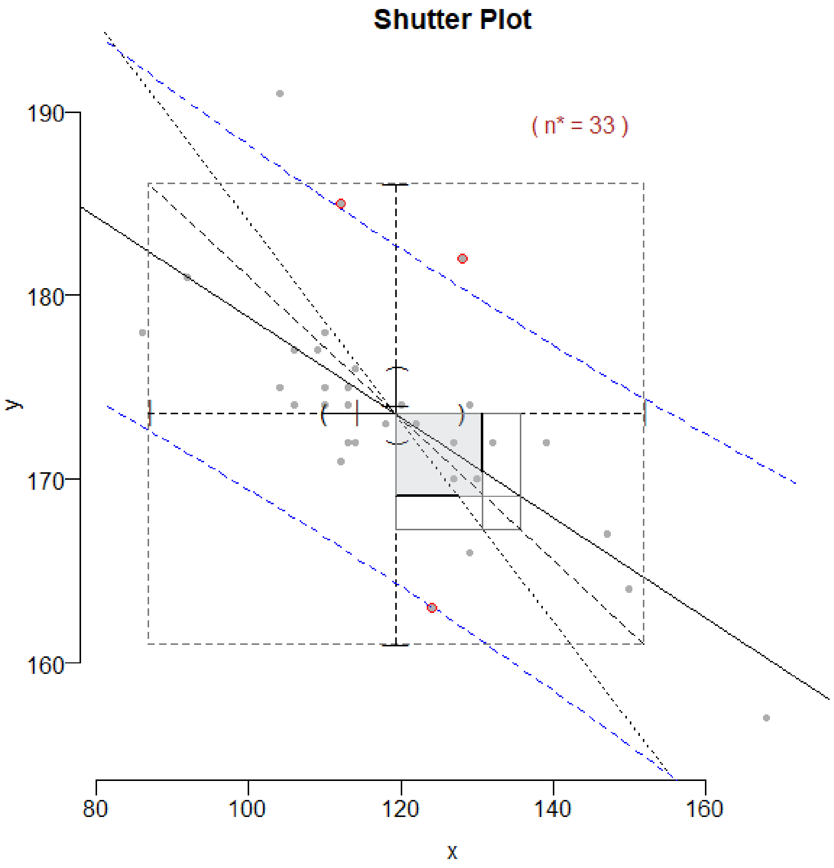


Figure 10: Shutter Plot for the body weight (pound) and volume of air exhaled (ml) by 33 girls on the high school marching band.

References

Sarkar, J., and M. Rashid. 2020. “Shutter Plot: A Visual Display of Summary Statistics over a Scatter Plot.” *International Journal of Statistical Sciences* 20 (2): 99–116.

Spear, M. E. 1952. “Charting Statistics.” *New York: McGraw-Hill.*

Spear, M. E. 1969. “Practical Charting Techniques.” *New York: McGraw-Hill.*

Wilkinson, L. 1999. “Dot Plot.” *The American Statistician* 53 (3): 276–81.