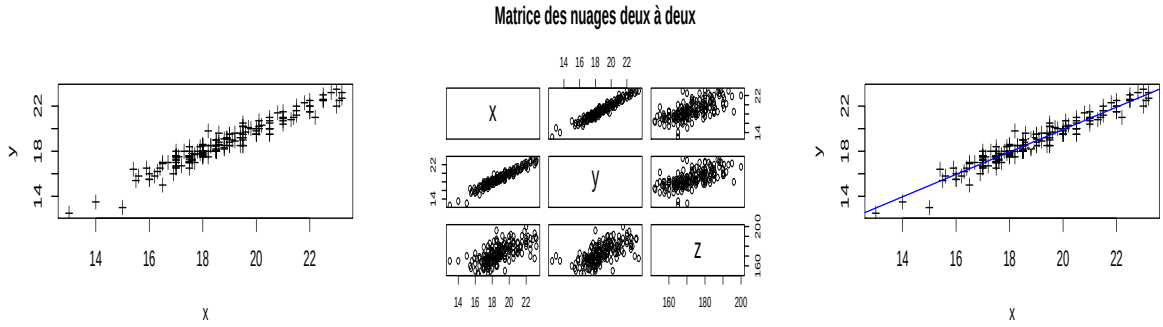
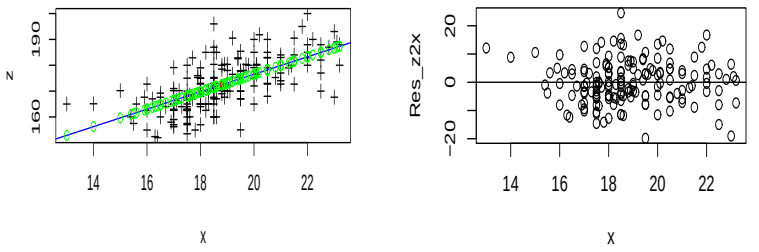
**Cours 05**
**Régression linéaire**

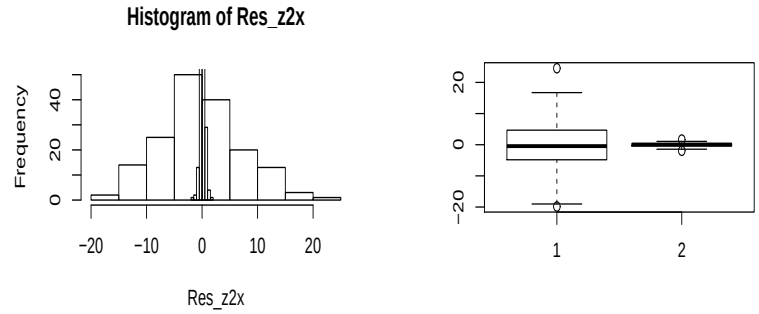
# 1 Nuage et droite




# 2 Résidus

On a choisi de représenter les résidus de la régression des `z` sur les `x` car ils sont plus grands que ceux de la régression des `y` sur les `x` : cette différence se voit aussi sur leurs histogramme et leurs boite à moustaches.




Les *valeurs prédites* par le modèle linéaire : `y2x[i]`$= \hat{y}_i = ax_i + b$. Elles sont données par la commande `fitted(`*`DteReg_y_sur_x`*`)` dans le premier exemple du cours.

Les *résidus* : `Res[i]`$= \epsilon_i = y_i - \hat{y}_i$

Les $a$ et $b$ sont choisis de manière à minimiser la somme des carrés des résidus $\sum_{i=1}^{n} \epsilon_i^2$.

# 3 Résumé

Le code distribué comporte à la fois des commandes `R` déjà connue, des commandes nouvelles, qui elles-même se divisent en commande de type gestion graphique, et des commandes spécifiques à la notion de statistique qui sont l'objet de ce cours : la régression linéaire ; il y a aussi quelques commandes de manipulation de données qui seront approfondies plus tard : nous pensons que vous pouvez commencer à la utiliser en mimant les exemples. Il y a enfin des nom de variables, choisi de façon à être facilement mémorisées, mais dont le choix est totalement arbitraire. C'est un bon exercice de distinguer ces trois derniers types, en soulignant/stabilotant de façon distincte ces trois derniers types (réservez le stabilo à la régression linéaire).

### Regression linéaire

`lm` Exemple : `DteReg_y_sur_x=lm(y x,data=MonTableau)`. Effectue la régression.
`coef` Exemple : `coef(DteReg_y_sur_x)`. Retournes les coefficients $b =$`Intercept` et $a$.
`abline` Exemple : `abline(coef(DteReg_y_sur_x),col="blue")`. Ajoute la droite de régression.
`fitted` Exemple : `y2x=fitted(DteReg_y_sur_x)`. Calcule les valeurs du caractère $y$ que prédirait le modèle pour l'échantillon `x` ; les différences avec les valeurs mesurées `y` s'appellent les résidus du modèle.

### Commandes graphiques

`plot` Exemple : `plot(x,z,type="p",pch=8)` : dessine le nuage de points.
`points` Exemple : `points(mean(x),mean(y),col="white",pch=19)` : ajoute un point.
`windows` Exemple : `windows()` : ouvre une nouvelle fenêtre.
`dev.cur` Exemple : `dev.cur()` : récupère le numéro de la fenêtre active.
`dev.set`

### Commandes qui seront revues plus tard

`data.frame` Exemple : *`MonTableau`*`=data.frame(x,y,z)`
`pairs` Exemple : `pairs( x+y+z,data=MonTableau, main="Matrice des nuages deux à deux")`

### Les noms de variables utilisés

Ce sont *`x`*, *`y`*, *`z`* qui sont les noms des échantillons utilisés, *`MonTableau`* qui est le tableau (data.frame) constitué de ces trois échantillons, *`DteReg_y_sur_x`* qui est le résultat de la régression linéaire des `y` sur les `x`, *`DteReg_z_sur_x`*, idem pour les `z`, *`y2x`* et `z2x` sont, pour l'échantillon `x`, les valeurs qui seraient prédites par le modèle linéaire $y = a_1x + b_1$ et $z = a_2x + b_2$. *`Res_z2x`*, et *`Res_y2x`* sont les résidus des deux régressions.

```
#        Cours de Stat Mass 05: regression linéaire / résidus
# initialisation des variables "Tailles" et "x", etc.
library(MASS)  # MASS = Modern Applied Statistics with S
data(survey)   #lecture des données de  survey
survey.cc<-survey[complete.cases(survey),] #nettoyage de  survey
x<-survey.cc$Wr.Hnd;x  #x  prends la valeur de la première colonne de  Tailles.
y<-survey.cc$NW.Hnd  #y  prends la valeur de la deuxième colonne de  Tailles.cc
z<-survey.cc$Height  #z  prends la valeur de la troisième colonne de  Tailles.cc

# Nuages
plot(x,y,pch=3) # pch=3 -> "+"  ; pch=10 ajouterait un rond au tour
points(mean(x),mean(y),col="red",pch=19) # (mean(x),mean(y)) est le Centre de Gravité

# Basic Scatterplot Matrix
pairs(~Wr.Hnd+NW.Hnd+Height,data=survey,
   main="Simple Scatterplot Matrix")

# On recommence avec des noms plus simples: x, y, z
MonTableau=data.frame(x,y,z)
head(MonTableau)
# Scatter-plot
pairs(~x+y+z,data=MonTableau,
   main="Matrice des nuages deux à deux")

# un nuage peu rectiligne
plot(x,z,type="p",pch=8)

# un nuage assez rectiligne
plot(x,y,type="p",pch=3)   # pch=1->ronds; 2->triangles; 3->"+";
# et sa regression de y sur x
DteReg_y_sur_x=lm(y~x,data=MonTableau)
abline(coef(DteReg_y_sur_x),col="blue")#ordonnée en 0 (b=Intercept) et pente (a) de la droite y=a
coef(DteReg_y_sur_x)
points(mean(x),mean(y),col="white",pch=19)
# summary(DteReg_y_sur_x) étudierons plus tard ce que fait cette commande
# les "prédites" y de x
y2x=fitted(DteReg_y_sur_x)
points(x,y2x,col="green",pch=1)
points(mean(x),mean(y),col="red",pch=19)

#retour au nuage peu rectiligne des (x,z)
plot(x,z,,pch=3) # pch=3 donnes des "+"
DteReg_z_sur_x=lm(z~x,data=MonTableau)
abline(coef(DteReg_z_sur_x),col="blue")
points(mean(x),mean(z),col="red",pch=19)
coef(DteReg_z_sur_x)#ordonnée à l'origine (b) et pente (a) de la droite y=ax+b (sic!)
# les "prédites" z de x
z2x=fitted(DteReg_z_sur_x)
points(x,z2x,col="green",pch=1)

#étude des "résidus"
windows();dev.cur()   # on ouvre une fenêtre et affiche le nom
Res_z2x=z - z2x
plot(x,Res_z2x)
abline(coef(lm(Res_z2x~x,data=MonTableau)))
hist(Res_z2x)
#hist(Res_z2x,add=T)

# on revient aux "bien alignés" (x,y)
dev.set(2);plot(x,y)
abline(coef(DteReg_y_sur_x),col="blue") # ordonnée à l'origine (a) et pente (b) de la droite y=bx
Res_y2x=y - y2x
dev.set(3)
plot(x,Res_y2x)
abline(coef(lm(Res_z2x~x,data=MonTableau)))
# hist(Res_y2x,add=TRUE)
hist(Res_y2x)
boxplot(Res_z2x,Res_y2x)
hist(Res_z2x)
hist(Res_y2x,add=TRUE)
```