TD 1 : Régression Linéaire avec R

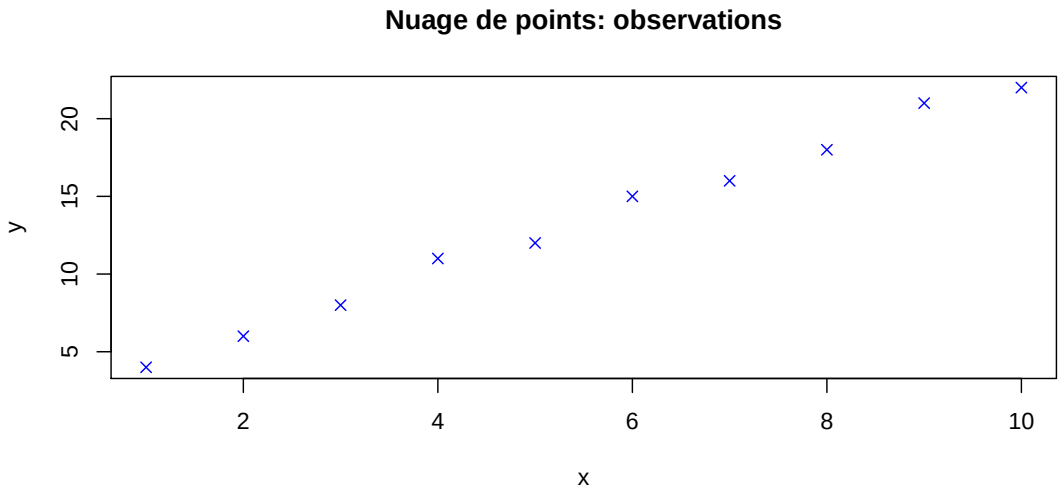
1 Calcul des coefficients de la droite de régression

On suppose qu'on dispose de n données (x_i, y_i) contenues dans un fichier .csv. On commence par lire ce fichier en affectant les valeurs dans une data frame nommées ici "donnees" puis on trace le nuage de points pour vérifier visuellement la potentielle linéarité.

```
> # Lecture des données dans un fichier csv (séparateur point virgule)
> donnees <- read.csv2("Droite.csv")
> print(t(donnees))# x=variable explicative,y=variable expliquée
```

```
  [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
x    1    2    3    4    5    6    7    8    9    10
y    4    6    8   11   12   15   16   18   21    22
```

```
> # t pour transposer le tableau
> # pour accèder aux variables de donnees par leur nom (seul) (simplifier leur nom)
> attach(donnees)# par ex donnees$x devient x
> # tracé: nuage de points
> eq = paste0("Nuage de points: observations")
> plot(x, y,type="p",pch=4,main=eq,col="blue")
```



Les points semblent alignés, on espère donc que le modèle suivi par les données est une droite d'équation

$$y = a_1 + a_2 x$$

dont il faut déterminer les coefficients.

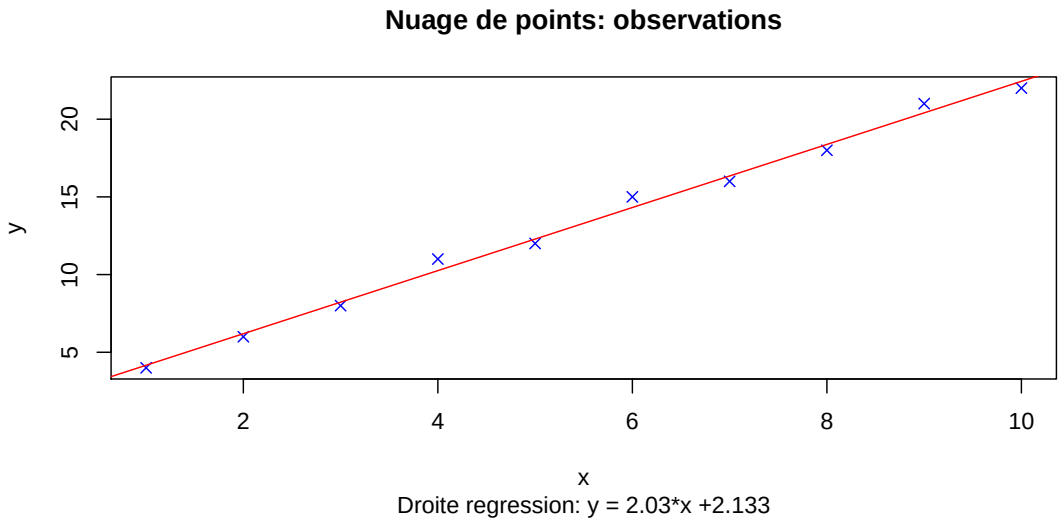
1.1 Calcul matriciel

La méthode des moindres carrés consiste à déterminer les valeurs a_1 et a_2 qui minimisent les écarts au carré entre les valeurs expérimentales et les valeurs prédites par le modèle (écarts aussi appelés résidus). C'est un problème de recherche du minimum d'une fonction de deux variables a_1 et a_2 (cf cours):

$$S(a_1, a_2) = \sum_{i=1}^{n} (a_1 + a_2 x_i - y_i)^2$$

Cela conduit à un système linéaire qu'il suffit d'inverser

```
> # y=a1+a2*x détermination de a1 et a2 par la méthode des moindres carrés
> # Construction de X et Y (cf cours)
> c1=rep(1,10)
> X<-matrix(c(c1,donnees$x),ncol = 2)
> Y<-matrix(donnees$y,ncol=1)
> M=t(X)%*%X # transposée(X) *X
> b<-t(X)%*%Y # transposée(X) *Y
> a=solve(M,b) # inversion du système donne le vecteur a=les coefficients a1 et a2
> plot(x, y,type="p",pch=4,main=eq,col="blue")# tracé du nuage de points (x,y)
> abline(a, col="red",main=eq)# tracé droite de regression (courbe de tendance linéaire)
> # equation de la droite sous forme de chaine de caractères
> eq = paste0("Droite regression: y = ", round(a[2],3), "*x +",round(a[1],3))
> title(sub=eq)# ajout d'un sous titre contenant l'équation de la droite
```

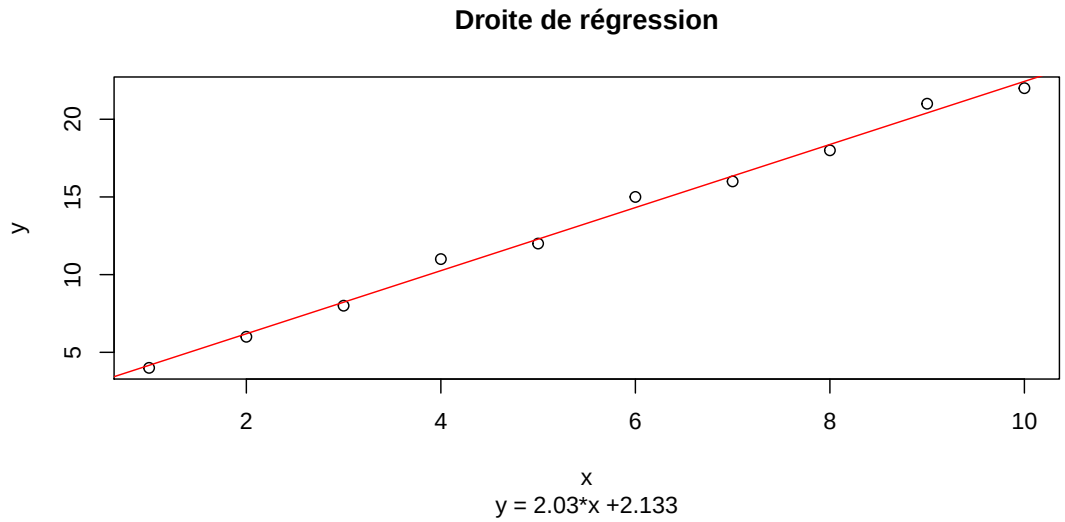


Nous avons ainsi déterminé la droite de régression par la méthode des moindres carrés c'est-à-dire en minimisant les écarts au carré entre les points expérimentaux et la droite.

1.2 Coefficients de la droite de régression avec lm

Il existe sous R une fonction qui permet le calcul des coefficients de la droite de régression sans avoir besoin de programmer les matrices X et Y (la fonction le fait elle même). C'est la fonction lm (pour Linear Models).

```
> droite <- lm(y~x, data=donnees)
> # y~x signifie chercher une relation entre y et x du type y=a1 + a2*x sur les données de la data frame donn
> print(droite) # affichage du résultat nommé ici droite qui est une liste

Call:
lm(formula = y ~ x, data = donnees)

Coefficients:
(Intercept)            x
      2.133        2.030

> a1 <- coef(droite)[1] # ordonnée à l'origine appelé intercept
> a2 <- coef(droite)[2] # la pente
> # tracé: nuage de points avec un titre principal et en sous titre l'équation
> eq = paste0("y = ", round(a2,3), "*x +",round(a1,3))
> plot(x, y,main="Droite de régression", sub=eq)
> # tracé: droite de regression (courbe de tendance linéaire)
> abline(droite, col="red")
```



On retrouve bien les mêmes coefficients qu'avec le calcul matriciel (évidemment, c'est le même calcul !). Mais on obtient en plus d'autres résultats.

2 Approche probabiliste: qualité de la prédiction

Si on veut estimer la qualité de la prédiction, on peut avoir une approche probabiliste: elle consiste à faire l'hypothèse que dans les observations notées (x_i, y_i), les x_i sont des valeurs exactes et certaines d'une variable non aléatoire x et les y_i sont des réalisations d'une variable aléatoire notée Y.
On fait la supposition que cette V.A. vérifie

$$Y = \alpha_1 + \alpha_2 x + \epsilon$$

où l'erreur ϵ est une variable aléatoire vérifiant $\epsilon \sim \mathcal{N}(0, \sigma^2)$. On peut montrer que les paramètres, a_1 et a_2, déterminés par les moindres carrés, sont alors les meilleurs estimateurs sans biais de α_1 et α_2 [1] de la méthode de maximum de vraisemblance. On peut utiliser cette régression pour répondre à certaines questions statistiques (signification, comparaison, prédiction, ...) que nous allons détailler. Mais on doit vérifier que $\epsilon \sim \mathcal{N}(0, \sigma)$ en étudiant la répartition des réalisations de la variable aléatoire, les résidus r_i (où $r_i = a_1 + a_2 x_i - y_i$).

2.1 Qualité de la prédiction: R^2, résidus, écart-type, tests...

2.1.1 Analyse de la variance

Lors de l'appel de lm, l'analyse de la variance a été faite

```
>   anova(droite)# table d'analyse de la variance

Analysis of Variance Table

Response: y
          Df Sum Sq Mean Sq F value   Pr(>F)
x          1 340.08  340.08    1344 3.36e-10 ***
Residuals  8   2.02    0.25
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> # Response: y
> #           Df   Sum Sq  Mean Sq       F value   Pr(>F)
> # x         Df1  SSR     MS1=SSR/Df1   MS1/MS2    pvalue prob(F(alpha,Df1,Df2)>F)
> # Residuals Df2   SSE    MS2=SSE/Df2
```

[1] Plus loin, on notera α le risque (à ne pas confondre avec α_1 et α_2)

```
> # H0 "pente nulle" est rejetée si pvalue<risque_alpha
>
> # SSR=sum square regression (variance expliquée par le modèle)
> # SSE= sum square error (variance non expliquée par le modèle)
> # SST=SSE+SSR (variance totale)
```

Dans la table, on obtient en colonnes

- les dégrés de liberté "Df" (pour les lois des tests sur les paramètres)
- les sommes des carrés (Sum Sq): Sum Square Regression: variance expliquée par le modèle et Sum Square Erreur: variance résiduelle non expliquée par le modèle.
- les moyennes de sommes des carrés (Mean Sq): SSR/Df1 et SSE/Df2 (estimation de la variance σ^2 de l'erreur ϵ)
- la statistique (F value) (quotient de deux valeurs précédentes-suit une distribution de Fisher $\mathcal{F}_{1,n-2}$).
- la p-value (Pr(>F)) associée au test d'hypothèse H_0 : "la pente a_2 est nulle" contre H_1 : "la pente n'est pas nulle". On rejette H_0 au risque α, si la p-value est inférieure à α. Dans notre cas, par exemple, on rejette l'hypothèse "la pente est nulle".

2.1.2 Commande Summary

On peut obtenir d'autres informations statistiques, en tapant la commande summary().

```
> summary(droite)# résumé des calculs effectués par l'instruction lm

Call:
lm(formula = y ~ x, data = donnees)

Residuals:
    Min      1Q  Median      3Q     Max
-0.4364 -0.3303 -0.2091  0.4046  0.7454

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.13333    0.34363   6.208 0.000257 ***
x            2.03030    0.05538  36.661 3.36e-10 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.503 on 8 degrees of freedom
Multiple R-squared:  0.9941,        Adjusted R-squared:  0.9933
F-statistic:  1344 on 1 and 8 DF,  p-value: 3.36e-10

> # Coefficients:
> # Estimation a1, sa1=ecart-type de a1, t=a1/sa1, probabilité critique PCa1
> # H0: la droite passe par l'origine"vs H1 la droite ne passe pas pas l'origine
> # H0 est rejetée si t<student(n-2,1-risque/2) ie si la pvalue PCa1<risque.
> # Estimation a2, sa2=ecart-type de a, t=a2/sa2, probabilité critique PCa2
> # H0: la pente de la droite est nulle" rejetée si t<student(n-2,1-risque/2)
> # H0: la pente de la droite est nulle" rejetée si la pvalue PCa2<risque.
> # Residual standard error: SQRT(MS2)
> # doit être faible pour que le modèle soit considéré comme bon (=prédictif)
> # Multiple R-squared:  SSR/SST,
> # Plus cette valeure sera proche de 1 meilleur sera l'ajustement.
> # Adjusted R-squared:  ajustement du R^2 au nombre p de variables explicatives
> #F-statistic:  MS1/MS2 on Df1 and Df2,  p-value
```

Dans la 1ère table "Residuals", on trouve les minimum et maximum ainsi que les quartiles des résidus. On peut aussi afficher les valeurs des résidus avec la commande

```
> resid(droite)# les résidus
```

```
         1          2          3          4          5
-0.1636364 -0.1939394 -0.2242424  0.7454545 -0.2848485
         6          7          8          9         10
 0.6848485 -0.3454545 -0.3757576  0.5939394 -0.4363636
```

Dans la seconde table "Coefficients"

- la colonne "estimate" contient les estimations de l'ordonnée à l'origine, a_1 appelé (intercept) puis de la pente, a_2,
- leurs écarts-types (Std. Error) respectifs,
- pour chaque paramètre, la statistique observée (t value) ainsi que la p-value ($\Pr(>|t|)$) associée au test d'hypothèse H_0 : "le paramètre est nul" contre H_1 : "le paramètre n'est pas nul". On rejette H_0 au risque α, si la p-value est inférieure à α. Dans notre cas, par exemple, on rejette les hypothèses "la pente est nulle" et "la droite passe par l'origine".

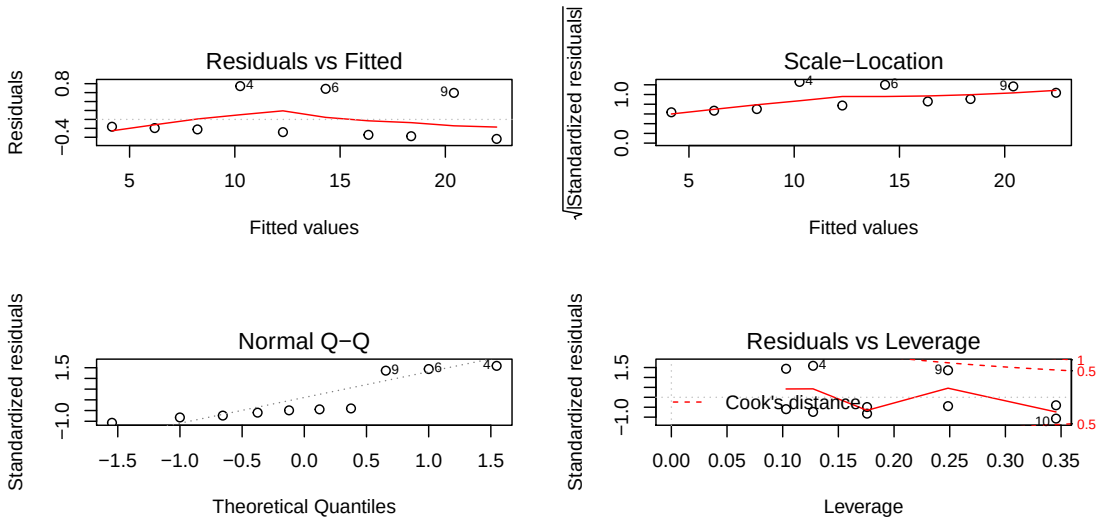
Ensuite, on trouve

- "Residual standard error" l'estimation s_r de l'écart-type σ de la VA ϵ où $s_r = \sqrt{\frac{SSE}{Df2}}$.
- "Multiple R-squared" le coefficient de détermination $R^2 = \frac{SSR}{SST}$ qui exprime le rapport entre la variance expliquée par le modèle et la variance totale. Ce coefficient varie entre 0 et 1. S'il est égal à 1, cela signifie que le modèle explique parfaitement les données.
- "Adjusted R-squared" le coefficient de détermination ajusté (prend en compte le nombre de variables)
- la statistique F déjà définie dans l'anova.

2.1.3 Analyse des résidus

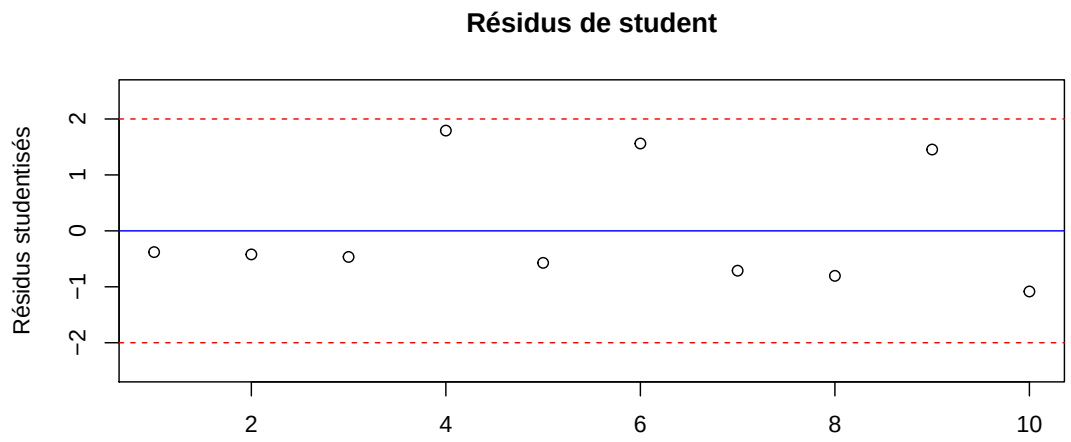
Une étape fondamentale dans la démarche de la régression est l'étude des résidus. En effet, les résultats statistiques donnés précédemment sont basés sur l'hypothèse que les erreurs (les résidus) sont indépendantes, normalement distribuées, de moyenne nulle et de variance constante. La fonction lm fournit quatre graphiques qui permettent d'en juger.

```
> layout(matrix(1:4,2,2))# fenetre graphique coupée en 4
> plot(droite) # 4 graphiques
```



Le premier graphe "Residual vs fitted" donne les résidus en fonction des valeurs prédites. Les points doivent être répartis aléatoirement autour de l'axe horizontal $y = 0$ et ne pas montrer de tendance. Le graphe "Scale-Location" donne la racine des résidus standardisés en fonction des valeurs prédites et ne doit pas non plus montrer de tendance. Le graphe "Normal Q-Q" permet de vérifier la normalité des résidus en comparant les quantiles de la population avec ceux de la loi normale (remarque: on peut aussi utiliser un test de Shapiro Wilk pour la normalité lorsque la validation visuelle ne suffit pas. On peut aussi tester que les résidus ne sont pas corréles avec un test de Durbin-Watson). Le dernier graphique "Residuals vs Leverage" met en valeur l'importance de chaque point dans la regression. On se questionnera particulièrement sur les points ayant une distance de Cook supérieure à 1 (rend la donnée suspecte=>point aberrant ?). On peut aussi afficher les résidus studentisés qui en pratique devront être compris entre les bornes -2 et 2.

```
> # résidus studentisés
> res <- rstudent( droite)
> # tracé des résidus studentisés (ylim échelle pour y)
> plot(res,ylab="Résidus studentisés",xlab="",main="Résidus de student",ylim=c(-2.5,2.5))
> # tracés des droites y=0 (trait plein bleu), y=+/- 2 (trait espacé rouge)
> # h= contient équation y=(-2,0,2) et lty le type de lignes
> abline(h=c(-2,0,2),lty=c(2,1,2),col=c("red","blue","red"))
```



En conclusion, les points suspects sont les points dont le résidu studentisé est supérieur à 2 en valeur absolue et/ou la distance de Cook est supérieure à 1. Dans ce dernier cas, le point contribue très/trop fortement à la détermination des coefficients du modèle comparativement aux autres. Il n'y a pas de méthode universelle pour traiter ce type de points.

2.1.4 Les prédictions: intervalle de confiance et de prédiction

On peut obtenir les valeurs prédites par la commande fitted

```
> fitted(droite)# les valeurs prédites
```

```
        1         2         3         4         5         6
 4.163636  6.193939  8.224242 10.254545 12.284848 14.315152
        7         8         9        10
16.345455 18.375758 20.406061 22.436364
```

On peut obtenir de nouvelles prédictions avec intervalle de confiance.

```
> newx=seq(1.5,10,1);
> pc=predict(droite,data.frame(x= newx), level = 0.95, interval = "confidence")
> print(pc)
```

```
        fit       lwr       upr
1  5.178788  4.549897  5.807679
```

```
2  7.209091  6.678678  7.739504
3  9.239394  8.792415  9.686373
4 11.269697 10.881287 11.658107
5 13.300000 12.933186 13.666814
6 15.330303 14.941893 15.718713
7 17.360606 16.913627 17.807585
8 19.390909 18.860496 19.921322
9 21.421212 20.792321 22.050103
```

Attention, un intervalle de confiance n'est pas un intervalle dans lequel la valeur a une probabilité de 95 % de se trouver. L'IC a 95 % de chance de contenir la vraie valeur si on répète les estimations un grand nombre de fois. Autrement dit, un intervalle de confiance à 95 % donnera un encadrement correct 95 fois sur 100.

On peut aussi obtenir un intervalle de prédiction

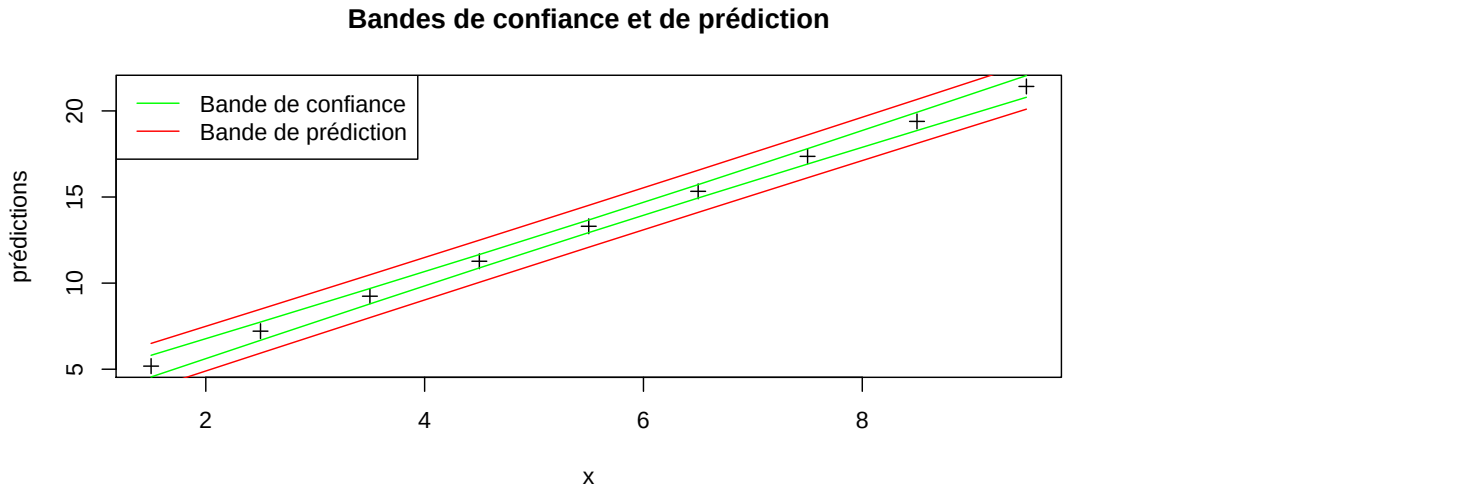
```
> newx=seq(1.5,10,1);
> pp=predict(droite,data.frame(x= newx), level = 0.95, interval = "prediction")
> print(pp)
```

```
        fit       lwr       upr
1  5.178788  3.859306  6.498269
2  7.209091  5.933605  8.484577
3  9.239394  7.996286 10.482502
4 11.269697 10.046427 12.492967
5 13.300000 12.083414 14.516586
6 15.330303 14.107033 16.553573
7 17.360606 16.117498 18.603714
8 19.390909 18.115423 20.666395
9 21.421212 20.101731 22.740694
```

Un intervalle de prédiction est un intervalle qui est susceptible de contenir une observation individuelle future. L'intervalle de prédiction est toujours plus large que l'intervalle de confiance.

On peut afficher les prédictions, les bandes de confiance et de prédiction

```
> plot( pp[,1] ~ newx, type='p',pch=3,ylab="prédictions",xlab="x" )
> points( pc[,2] ~ newx, type='l', col="green" )
> points( pc[,3] ~ newx, type='l', col="green" )
> points( pp[,2] ~ newx, type='l', col="red" )
> points( pp[,3] ~ newx, type='l', col="red" )
> title(main="Bandes de confiance et de prédiction")
> legend("topleft", c("Bande de confiance", "Bande de prédiction"),lwd=1, lty=1, col=c("green", "red") )
```



3 Généralisation

On peut faire de la régression linéaire autre que le cas de la droite avec R. La syntaxe est lm(y∼formule)
où y est le nom de la variable à prédire et formule est l'équation du modèle. Pour accèder aux différents calculs statistiques, il faut définir une liste contenant le résultat res=lm(y∼formule)

Formule	Modèle	Commentaires
y~x	$y(x) = a_1 + a_2 x$	droite
y~-1+ x	$y(x) = a_2 x$	droite passant par l'origine
y~x+I(x^2)	$y(x) = a_1 + a_2 x + a_3 x^2$	polynôme d'ordre deux. La fonction I() permet de programmer des fonctions mathématiques
y~poly(x,deg)	$y(x) = \sum_{i=0}^{deg} P_i(x)$	polynômes (orthogonaux) de Legendre d'ordre deg.
y~x+z	$y(x) = a_1 + a_2 x + a_3 z$	termes du 1er ordre en x et z sans terme en xz
y~x:z	$y(x) = a_1 + a_2 xz$	1er ordre seulement le terme en xz
y~x*z	$y(x) = a_1 + a_2 x + a_3 z + a_4 xz$	1er ordre complet
y~(x+z+t)^2	$y(x) = a_1 + a_2 x + a_3 z + a_4 t + a_5 xz + a_6 xt + a_7 zt$	équivalent à x*z*t-x:z:t

4 Exercice

Exercice 4.1 *On dispose des données expérimentales dans le fichier poly.csv (à télécharger sour Moodle).*

1. *La commande* **nom=file.choose()** *permet de choisir un fichier dans le répertoire. En utilisant cette commande, sélectionner le fichier poly.csv*

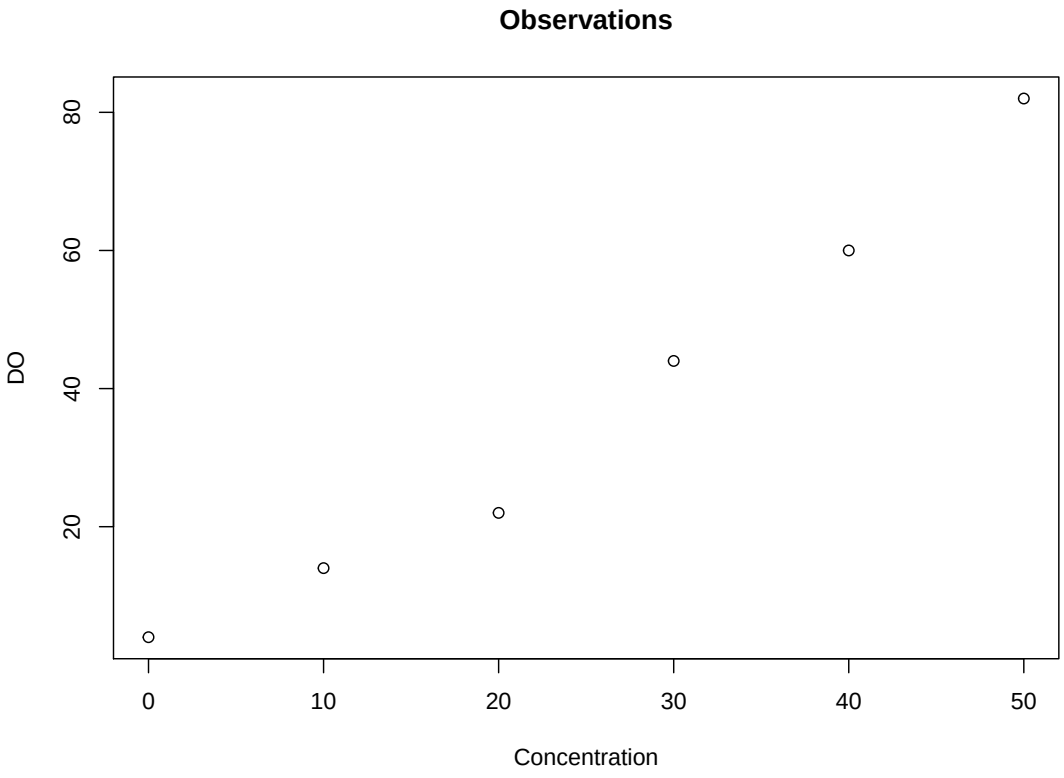
   ```
   > nom=file.choose()
   ```

2. *Lire les données contenues dans nom (poly.csv) en les affectant à une data frame nommée obs et on fera un attach(obs) pour simplifier le nom des variables.*

   ```
   > obs <- read.csv2(nom)
   > attach(obs)# pour simplifier les noms des variables
   ```

3. *Afficher le nuage de points ($conc_i$,DO_i).*

   ```
   > titre="Observations"
   > plot(conc,DO,main=titre,xlab="Concentration",ylab="DO")
   ```



4. *Droite de régression*

   (a) *Détermier les paramètres* (α_1, α_2) *du modèle*

$$DO = \alpha_1 + \alpha_2 conc + e$$

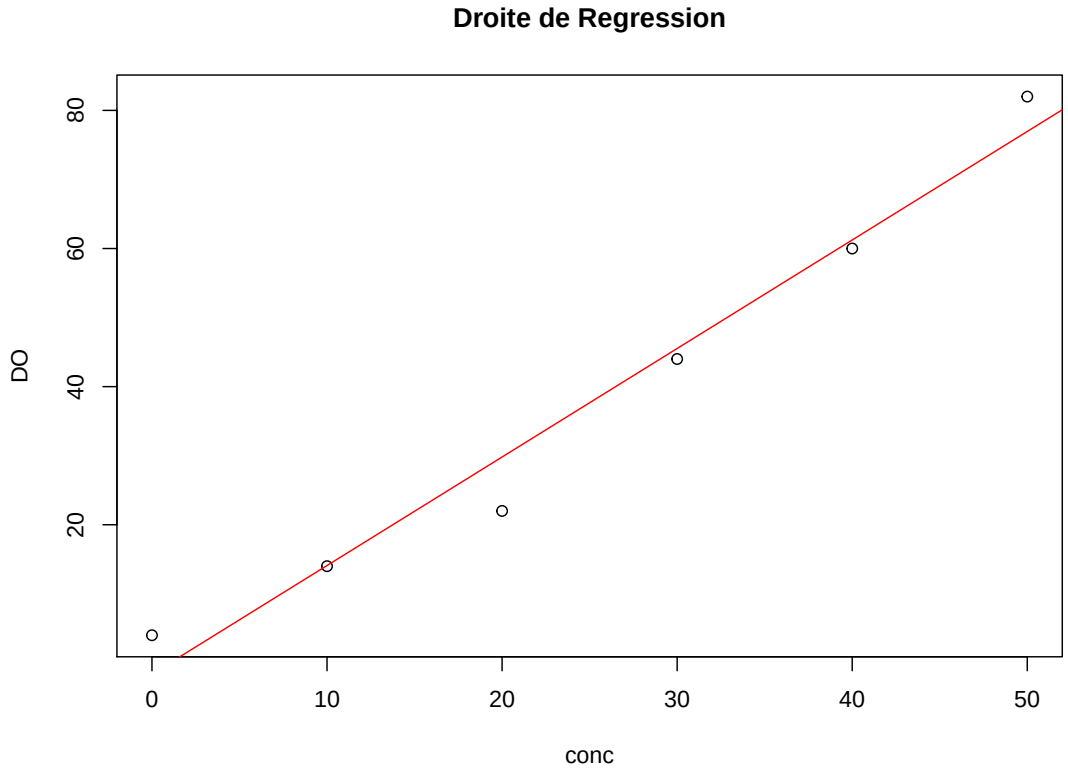
```
> modele=lm(DO~conc,data=obs)
> summary(modele)

Call:
lm(formula = DO ~ conc, data = obs)

Residuals:
       1        2        3        4        5        6
 5.61905 -0.09524 -7.80952 -1.52381 -1.23810  5.04762

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   -1.619      3.995  -0.405 0.706051
conc           1.571      0.132  11.908 0.000285 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 5.521 on 4 degrees of freedom
Multiple R-squared:  0.9726,        Adjusted R-squared:  0.9657
F-statistic: 141.8 on 1 and 4 DF,  p-value: 0.0002849
```

Si on regarde le R^2 *(le rapport de la variance expliquée par la régression SSR sur la variance totale SST) la régression est bonne: le modèle reproduit bien les données, la proportion de variabilité expliquée par le modèle est importante.*
On voit que l'on peut accepter l'hypothèse d'une relation linéaire entre les variables (accepter l'hypothèse "la

pente de la droite n'est pas nulle"). Mais il n'y a pas de preuve que la droite ne passe pas par l'origine: lorsque l'on teste l'hypothèse H_0 "la droite passe par l'origine" contre l'hypothèse H_1 "la droite ne passe pas par l'origine", on peut accepter l'hypothèse H_0 au risque $\alpha = 0.05$ (on refute l'hypothèse H_1 avec un risque $\alpha = 0.05$ de se tromper). On essayera donc de modéliser les points par une droite passant par l'origine.

(b) Tracer sur un même graphe le nuage de points et la courbe théorique obtenue en rouge. Ajouter un titre.

```
> plot(conc,DO,main="Droite de Regression")
> abline(modele,col="red")
```

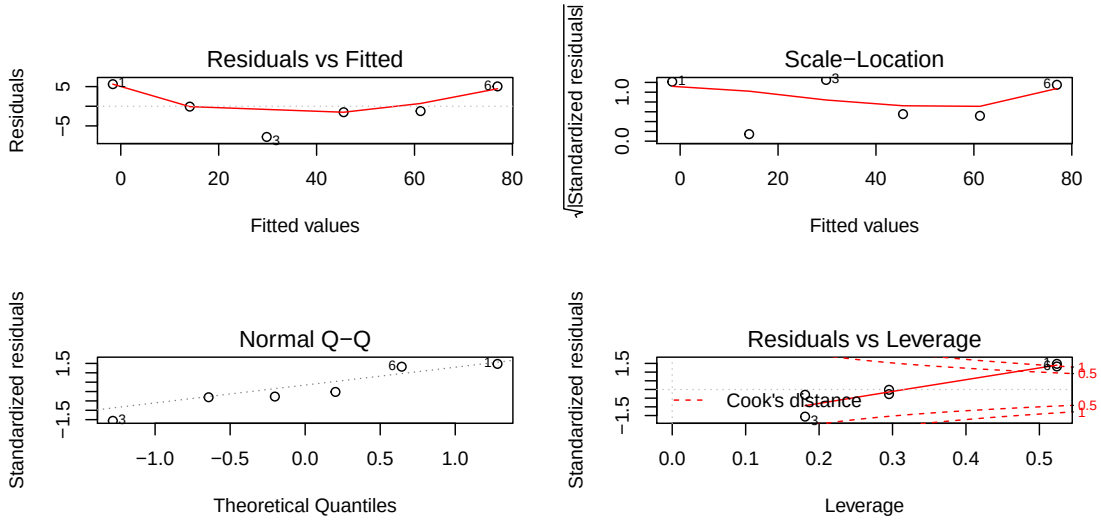


D'autres commandes parfois utiles

```
> coef(modele)#coefficients du modèle

(Intercept)        conc
  -1.619048    1.571429

> resid(modele)#residus du modele

        1          2          3          4          5
 5.6190476 -0.0952381 -7.8095238 -1.5238095 -1.2380952
        6
 5.0476190

> fitted(modele)#valeurs predites par le modele

       1         2         3         4         5         6
-1.619048 14.095238 29.809524 45.523810 61.238095 76.952381

> library(pwr)
```

(c) Etudier les résidus (residus en fonction des valeurs prédites, résidus standardisés, Distance de Cook, résidus studentisés, ...) et commenter.

```
> layout(matrix(1:4,2,2))# fenetre graphique coupée en 4
> plot(modele) # 4 graphiques
```



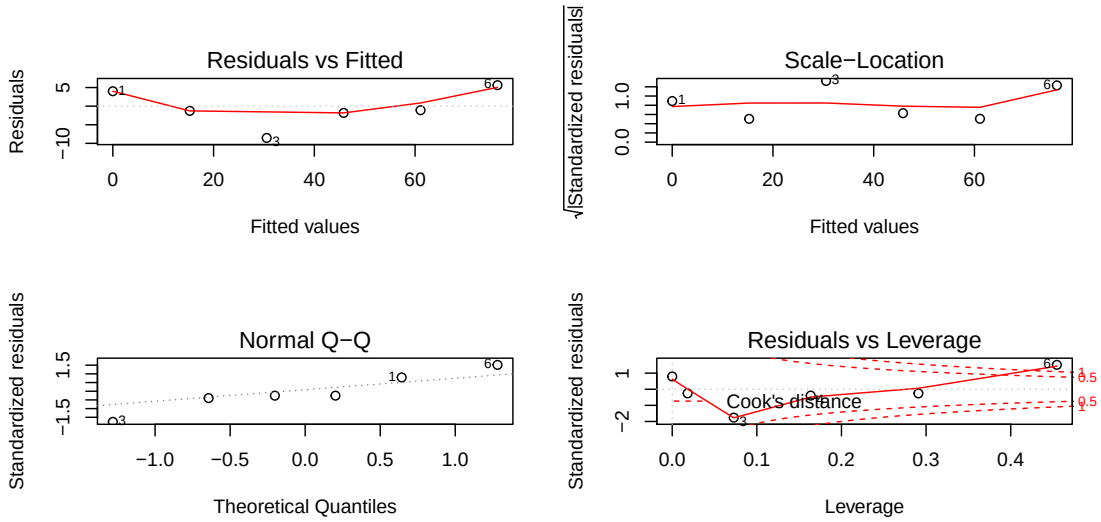
Dans le premier graphe, on observe une tendance qui laisse penser que le modèle n'est pas adéquate (peut-être ajouter un effet conc^2). Le troisième teste la normalité des résidus (on peut aussi la tester avec un test de Shapiro Wilk). Le dernier montre que certains points sont suspects (ils ont une importance exagérée dans la détermination des coefficients), il faut s'assurer qu'ils ne sont pas dus à des erreurs.

5. *Droite de régression passant par l'origine*

   (a) *Détermier le paramètre (α) du modèle*

$$DO = \alpha_2 conc + e$$

```
> d_origine=lm(DO~-1+conc,data=obs)
> summary(d_origine)

Call:
lm(formula = DO ~ -1 + conc, data = obs)

Residuals:
     1      2      3      4      5      6
 4.000 -1.273 -8.545 -1.818 -1.091  5.636

Coefficients:
     Estimate Std. Error t value Pr(>|t|)
conc  1.52727    0.06793   22.48 3.24e-06 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 5.038 on 5 degrees of freedom
Multiple R-squared:  0.9902,        Adjusted R-squared:  0.9882
F-statistic: 505.4 on 1 and 5 DF,  p-value: 3.236e-06

> layout(matrix(1:4,2,2))# fenetre graphique coupée en 4
> plot(d_origine)
```



On peut faire les mêmes critiques que précédemment.

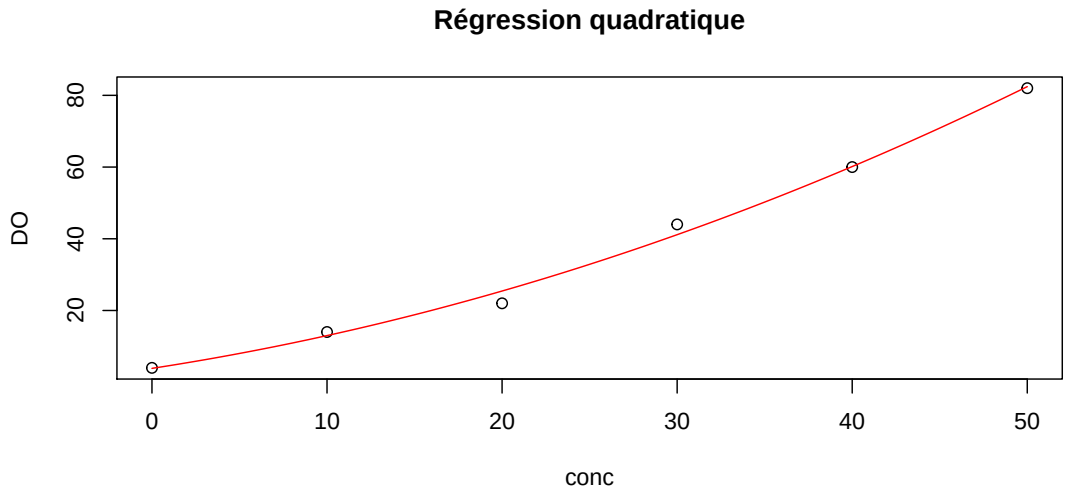
6. *On cherche maintenant à approcher le nuage de points selon le critère des moindres carrés par un polynôme de degré deux du type* $\alpha_1 + \alpha_2 conc + \alpha_3 conc^2$.

   (a) *Détermier les paramètres* $(\alpha_1, \alpha_2, \alpha_3)$ *en utilisant la fonction lm de Rstudio.*

```
> quadra=lm(DO~conc+I(conc^2),data=obs)
> summary(quadra)

Call:
lm(formula = DO ~ conc + I(conc^2), data = obs)

Residuals:
      1       2       3       4       5       6
 0.1429  1.0000 -3.4286  2.8571 -0.1429 -0.4286

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 3.857143   2.406058   1.603   0.2072
conc        0.750000   0.226321   3.314   0.0453 *
I(conc^2)   0.016429   0.004345   3.781   0.0324 *
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.655 on 3 degrees of freedom
Multiple R-squared:  0.9952,        Adjusted R-squared:  0.9921
F-statistic: 313.7 on 2 and 3 DF,  p-value: 0.0003282
```

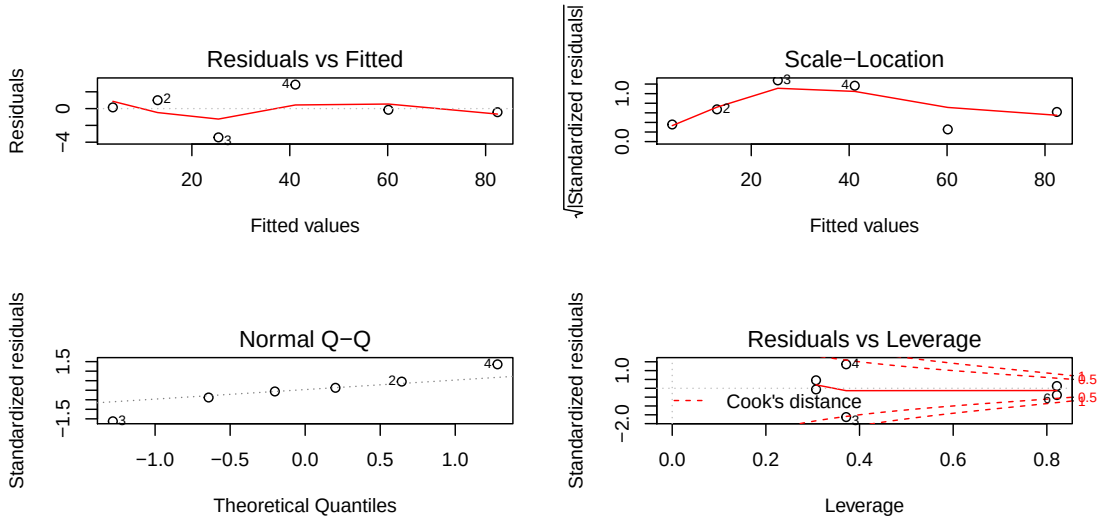
   (b) *Tracer sur un même graphe le nuage de points et la courbe théorique obtenue (cf help: curve) en rouge. Ajouter un titre.*

```
> plot(conc,DO,main="Régression quadratique") #
> coef(quadra)

(Intercept)        conc   I(conc^2)
 3.85714286  0.75000000  0.01642857

> curve(coef(quadra)[1]+coef(quadra)[2]*x+coef(quadra)[3]*x^2,0,max(conc),add=TRUE,col="red")
```



(c) *Etudier les résidus comme précédemment.*

```
> layout(matrix(1:4,2,2))# fenetre graphique coupée en 4
> plot(quadra) # 4 graphiques
```



7. *Quel est le meilleur modèle ? On peut regarder les paramètres et leurs écart-type, les résidus, le* R^2_{adj}*, utiliser le critère AIC (plus le critère est faible, meilleur est le fit).*

```
> AIC(modele,d_origine,quadra)
```

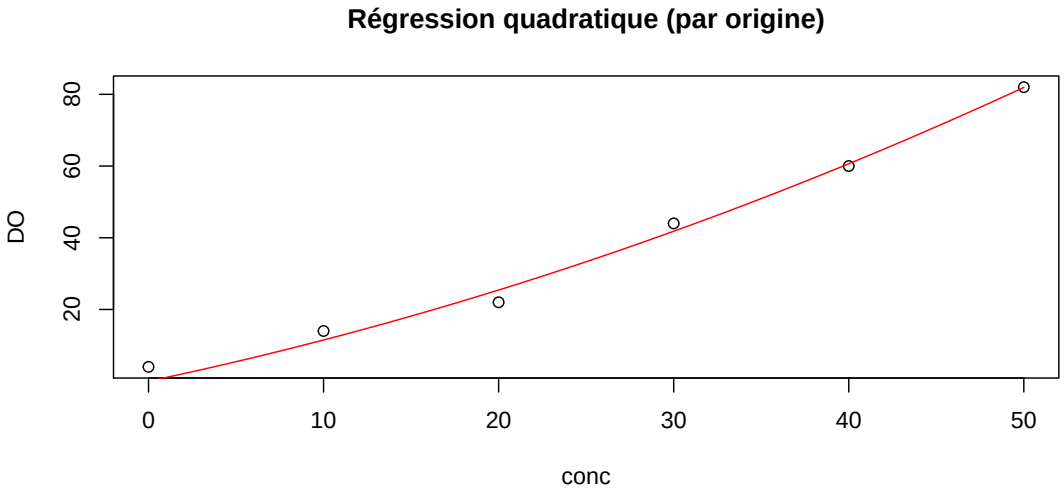
```
           df      AIC
modele      3 41.09615
d_origine   2 39.33753
quadra      4 32.58452
```

Le modèle quadratique est le meilleur selon le critère AIC (meilleur R^2_{adj} *aussi). Mais les écarts-types des paramètres sont importants et on ne peut pas rejeter l'hypothèse de nullité pour l'un. Le modèle droite passant par l'origine semble le mieux déterminé. On peut tester le modèle quadratique passant par l'origine.*

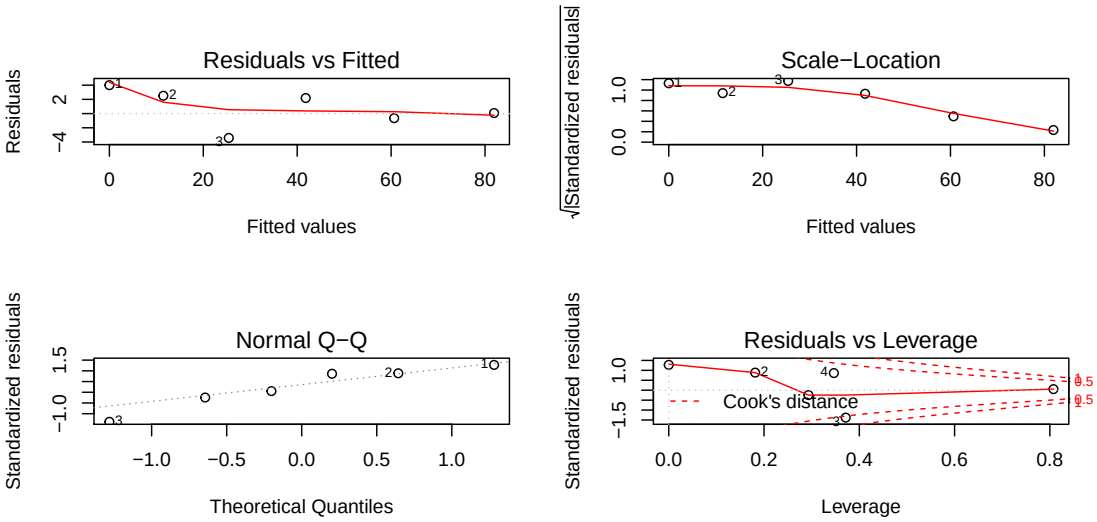
```
> quadra_origine=lm(DO~-1+conc+I(conc^2),data=obs)
> summary(quadra_origine)
```

```
Call:
lm(formula = DO ~ -1 + conc + I(conc^2), data = obs)

Residuals:
       1        2        3        4        5        6
 4.00000  2.50932 -3.42857  2.18634 -0.64596  0.07453

Coefficients:
          Estimate Std. Error t value Pr(>|t|)
conc      1.026708   0.172734   5.944  0.00402 **
I(conc^2) 0.012236   0.004094   2.989  0.04039 *
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.133 on 4 degrees of freedom
Multiple R-squared:  0.997,        Adjusted R-squared:  0.9955
F-statistic: 658.1 on 2 and 4 DF,  p-value: 9.18e-06
```

```
> plot(conc,DO,main="Régression quadratique (par origine)") #
> curve(coef(quadra_origine)[1]*x+coef(quadra_origine)[2]*x^2,0,max(conc),add=TRUE,col="red")
```




```
> layout(matrix(1:4,2,2))# fenetre graphique coupée en 4
> plot(quadra_origine) # 4 graphiques
```




```
> AIC(modele,d_origine,quadra)

           df      AIC
modele      3 41.09615
d_origine   2 39.33753
quadra      4 32.58452
```

```
> AIC(modele,d_origine,quadra_origine)

                df      AIC
modele           3 41.09615
d_origine        2 39.33753
quadra_origine   3 34.29713
```

Le modèle quadratique passant par l'origine semble adéquat mais le nombre de points est à peine suffisant. D'autre part, le coefficient devant conc^2 est petit relativement à celui de conc. De ce fait, seuls les points où conc est grand interviennent véritablement dans la détermination de ce coefficient (on parle de sensibilité du modèle par rapport aux paramètres). Plus le nombre de points est important, plus la détermination des paramètres s'améliore mais il faut veiller à ne pas surparamétrer un modèle: plus on a de paramètres, meilleur semble le résultat du fit. Mais si on reproduit les mesures, obtiendra-t-on les mêmes paramètres ? En cas de doute, on choisira donc le modèle le plus simple.