TP Statistique

Cédric Milinaire, Corentin Laharotte

4 avril 2020

Voici le plan de ce qui sera fait dans le TP.

0. Visualisation de chemins

Lecture du fichier des villes :

```
villes <- read.csv('DonneesGPSvilles.csv',header=TRUE,dec='.',sep=';',quote="\"")
str(villes)
```

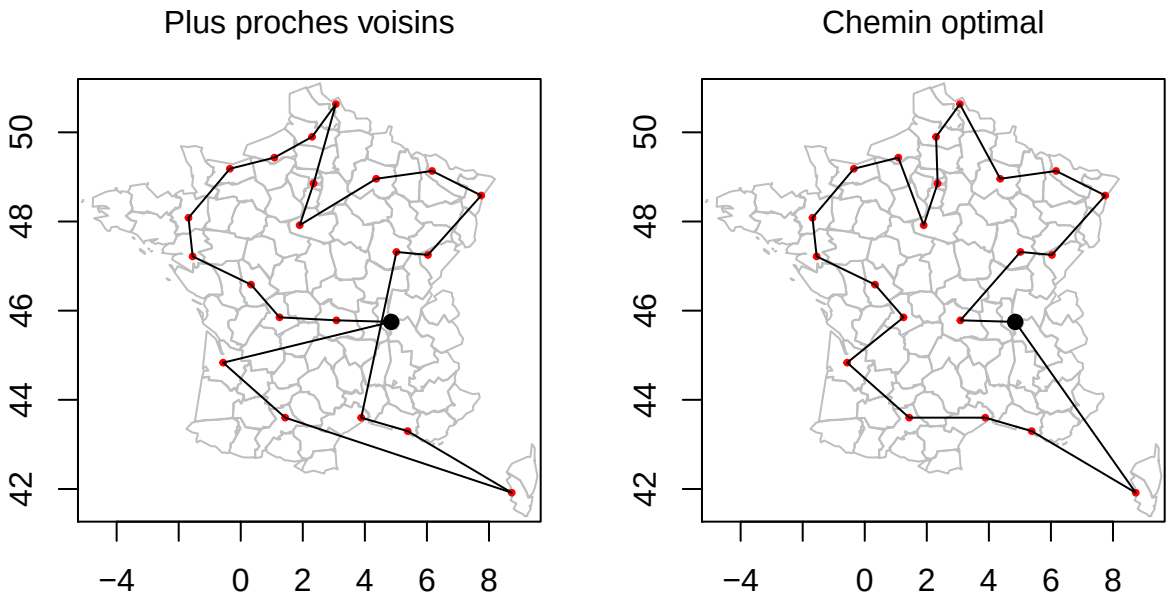
```
## 'data.frame':    22 obs. of  5 variables:
##  $ EU_circo : Factor w/ 7 levels "Centre","Est",..: 6 6 4 2 7 4 2 1 2 4 ...
##  $ region   : Factor w/ 22 levels "Alsace","Aquitaine",..: 22 9 19 10 2 4 8 3 5 17 ...
##  $ ville    : Factor w/ 22 levels "Ajaccio","Amiens",..: 11 1 2 3 4 5 6 7 8 9 ...
##  $ latitude : num  45.7 41.9 49.9 47.2 44.8 ...
##  $ longitude: num  4.847 8.733 2.3 6.033 -0.567 ...
```

Représentation des chemins par plus proches voisins et du chemin optimal :

```
coord <- cbind(villes$longitude,villes$latitude)
dist <- distanceGPS(coord)
voisins <- TSPnearest(dist)

pathOpt <- c(1,8,9,4,21,13,7,10,3,17,16,20,6,19,15,18,11,5,22,14,12,2)

par(mfrow=c(1,2),mar=c(1,1,2,1))
plotTrace(coord[voisins$chemin,], title='Plus proches voisins')
plotTrace(coord[pathOpt,], title='Chemin optimal')
```



Les longueurs des trajets (à vol d'oiseau) valent respectivement, pour la méthode des plus proches voisins :

```
## [1] 4303.568
```

et pour la méthode optimale :

```
## [1] 3793.06
```

Ceci illustre bien l'intérêt d'un algorithme de voyageur de commerce. Nous allons dans la suite étudier les performances de cet algorithme.

1. Comparaison d'algorithmes

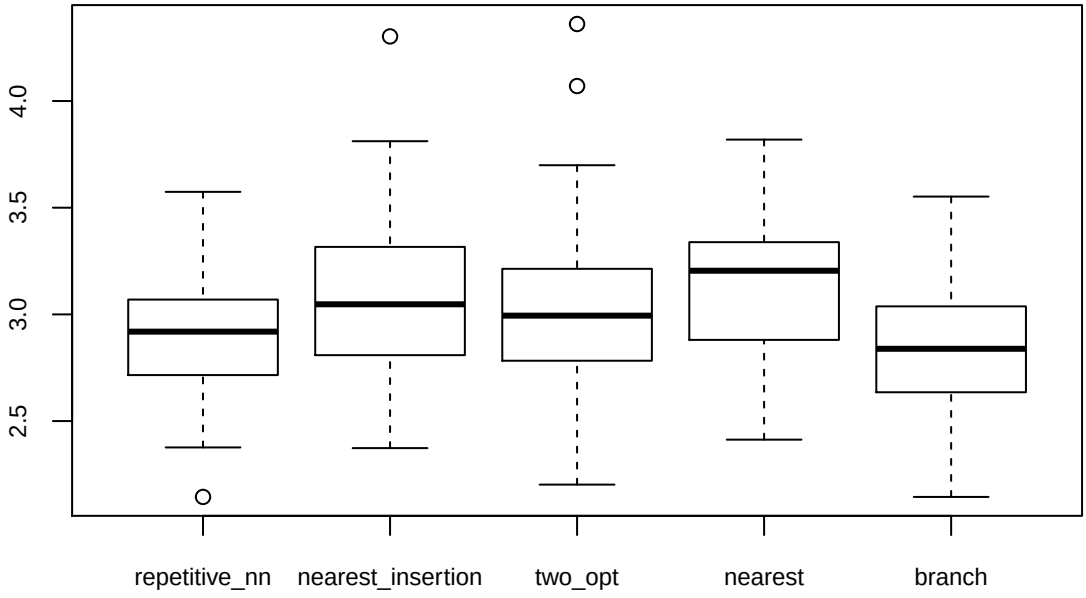
Dans cette partie, nous souhaitons comparer les méthodes repetitive_nn, nearest_insertion, two_opt, nearest, et branch. Pour cela, nous allons générer des graphes aléatoires de 10 sommets, et tester les longueurs des chemins calculés et le temps de calcul des différentes méthodes.

```
      n <- 10
sommets <- data.frame(x = runif(n), y = runif(n))
  couts <- distance(sommets)
```

1.1. Longueur des chemins

Dans un premier temps, nous allons comparer les longueurs des chemins hamiltoniens calculés par les 5 méthodes sur 50 réalisations de graphes aléatoires.

- Représentation de la longueur des chemins hamiltoniens obtenus par différentes méthodes :



L'affichage sous forme de boxplot nous permet de remarquer que : * la méthode branch renvoie le plus souvent un chemin plus court que les autres méthodes * la méthode nearest renvoie le plus souvent un chemin plus long que les autres méthodes * la boîte de la méthode repetitive_nn est moins étendue que les boîtes obtenues par les autres méthodes, ce qui nous permet de constater que 50% des valeurs sont très proches de la valeur médiane * la boîte de la méthode nearest_insertion est plus étendue que les boîtes obtenues par les autres méthodes, ce qui nous permet de constater que 50% des valeurs sont assez étendues autour de la valeur médiane

L'affichage obtenu est assez cohérent puisqu'aucune méthode n'a de valeur moyenne complètement absurde par rapport aux autre méthodes.

- test entre 'nearest' et 'branch'

On souhaite maintenant comparer les méthodes des plus proches voisins et Branch&Bound. On réalise donc un test sur l'espérance de chaque méthode. Notre hypothèse nulle (H0) est que la moyenne des chemins hamiltoniens obtenus avec la méthode des plus proches voisins est inférieure ou égale à la moyenne des chemins hamiltoniens obtenus avec la méthode Branch&Bound. Notre hypothèse alternative (H1) est que la moyenne des chemins hamiltoniens obtenus avec la méthode des plus proches voisins est supérieure à la moyenne des chemins hamiltoniens obtenus avec la méthode Branch&Bound. (H0) $m_{nn} - m_b <= 0 <=> m_{nn} <= m_b$ (H1) $m_{nn} - m_b > 0 <=> m_{nn} > m_b$

Nous allons ensuite tester si au seuil de 5% la moyenne des chemins hamiltoniens obtenus avec la méthode des plus proches voisins est inférieure ou égale à la moyenne des chemins hamiltoniens obtenus avec la méthode Branch&Bound. Pour cela, nous allons faire une comparaison d'échantillons gaussiens appariés.En effet, les deux méthodes étant basées sur les mêmes graphes, les résultats obtenus ne peuvent pas être considérés comme indépendant.

On pose $a = 0.05$.

On obtient une p_{valeur} de :

```
## [1] 7.011422e-12
```

```
## [1] "p_valeur < a"
## [1] "On peut rejeter H0"
```

On observe que la p_{valeur} obtenue est strictement inférieure à a. On peut rejeter H0, et affirmer avec un risque de 5% que les chemins hamiltoniens obtenus avec la méthode des plus proches voisins sont en moyenne

plus longs que ceux obtenus avec la méthode Branch&Bound.

- tests 2 à 2

Ici (H0) mi=mj (H1) mi!=mj

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  results and methods
##
##                   branch  nearest nearest_insertion repetitive_nn
## nearest           0.00078 -       -                 -
## nearest_insertion 0.02272 0.94921 -                 -
## repetitive_nn     0.94921 0.01702 0.20157           -
## two_opt           0.09341 0.53849 0.94921           0.53849
##
## P value adjustment method: holm
```

A COMMENTER

Si on accepte de se tromper de alpha=5%, on rejette H0 si la pvaleur de (i,j) est inférieure à alpha.

On rejette H0: - nearest branch - insertion branch - reptitve nearest

Avec un risque de 5% nous pouvons affirmer que ces méthodes ne sont pas similaires sur la longueur moyenne des chemins calculés

1.2. Temps de calcul

Comparaison des temps à l'aide du package microbenchmark.

Application de microbenchmark :

```
## Unit: microseconds
##                                  expr      min        lq      mean    median
##      TSPsolve(couts, "repetitive_nn") 4803.654 4907.4305 5444.7276 5259.9675
##  TSPsolve(couts, "nearest_insertion")  741.516  783.4450  915.6111  822.5985
##            TSPsolve(couts, "two_opt")  422.178  486.8580  715.4303  537.6500
##            TSPsolve(couts, "nearest")   11.016   14.2065   16.6584   15.7890
##             TSPsolve(couts, "branch") 1172.664 2289.5095 3889.0759 3225.1335
##         uq       max neval cld
##  5788.4315  6926.277    20   c
##  1012.2025  1490.706    20 a
##   710.3825  1887.008    20 a
##    17.8720    27.099    20 a
##  4365.4130 11378.412    20  b
```

-en moyenne ils sont tous a peu pres equivalent en parecequ'on ne peut pas rejeter h0 pour tous sauf repitive nn -confirmer en regardant les chiffres repetitve nn prend en moyenne au moins 2x plus de temps que les autres

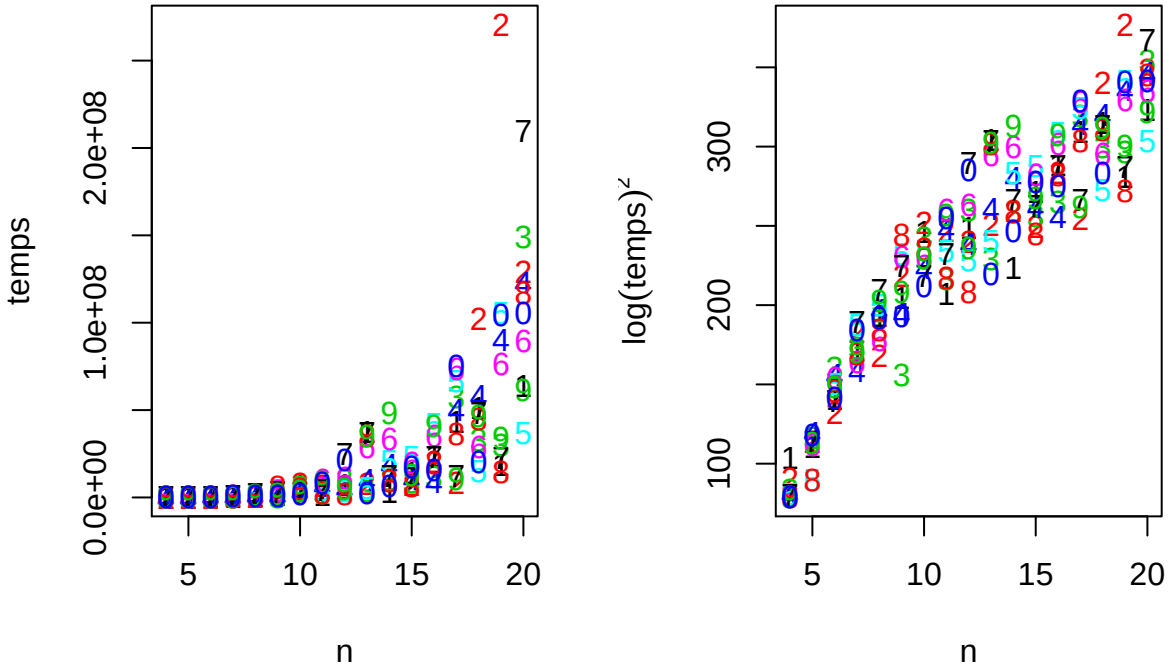
commentaire prof

Tout les membres d'un meme groupe n'ont pas de différence significative pour leurs moyenne et les groupes {a, b, c, d, . . . } sont rangés de manière croisante. Exemple - si variables X et variable Y sont dans le groupe a alors m X ' m Y où plutot qu'il n'a pas pu être mis en évidence une différence significative entre les deux. - si variables X et variable Y sont dans le groupe a et b alors m X 6 = m Y significativement. Et comme {a, b, c, d, . . . } sont rangés de manière croisante alors m a < m b donc m X < m Y

2. Etude de la complexité de l'algorithme Branch and Bound

2.1. Comportement par rapport au nombre de sommets : premier modèle

Récupération du temps sur 10 graphes pour différentes valeurs de n.



Les nombres représentés sont les numéros de colonnes de la valeur à la nième ligne !

Ajustement du modèle linéaire de $\log(temps)^2$ en fonction de n.

```
##
## Call:
## lm(formula = vect_temps ~ vect_dim)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -60.782 -18.731   0.566  18.227  55.455
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  66.4787     5.3757   12.37   <2e-16 ***
## vect_dim     14.0181     0.4147   33.80   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 26.49 on 168 degrees of freedom
## Multiple R-squared:  0.8718, Adjusted R-squared:  0.871
## F-statistic:  1142 on 1 and 168 DF,  p-value: < 2.2e-16
```

COMMENTER

- on peut voire que log(tmps^2) en fonction de n suit une courbe lineaire (R squareed = 0.8705) du coups le temps est une fonction exponentielle de n i.e la complexite de temps est exponentielle.

Analyse de la validité du modèle :

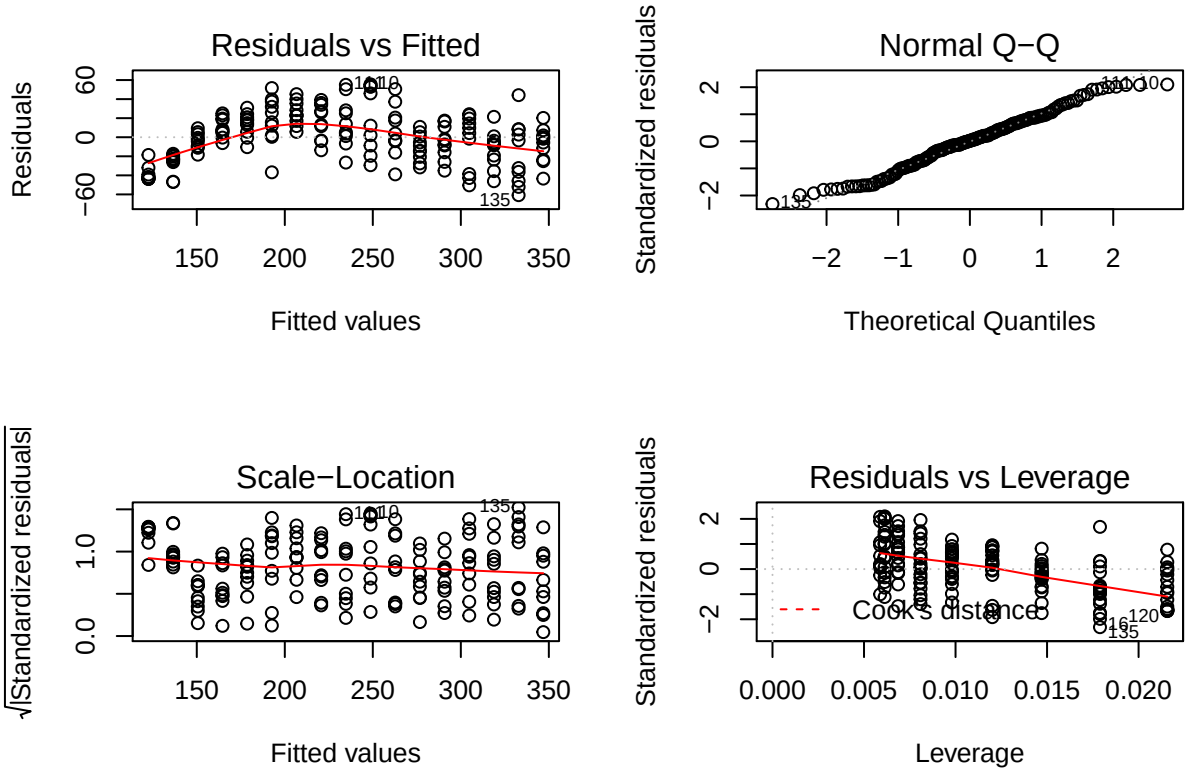
Le modèle nous renvoie une fonction de type: $Y = aX + b + \epsilon$. En effet nous avons les paramètres suivants:

$a = 14.7$, $b = 68.6$. Il reste donc a savoir les coéfficients et donc le modèle sont pertinents. Nous allons tous d'abord analyser la pertinence des coefficients puis celle du modèle en géneral.

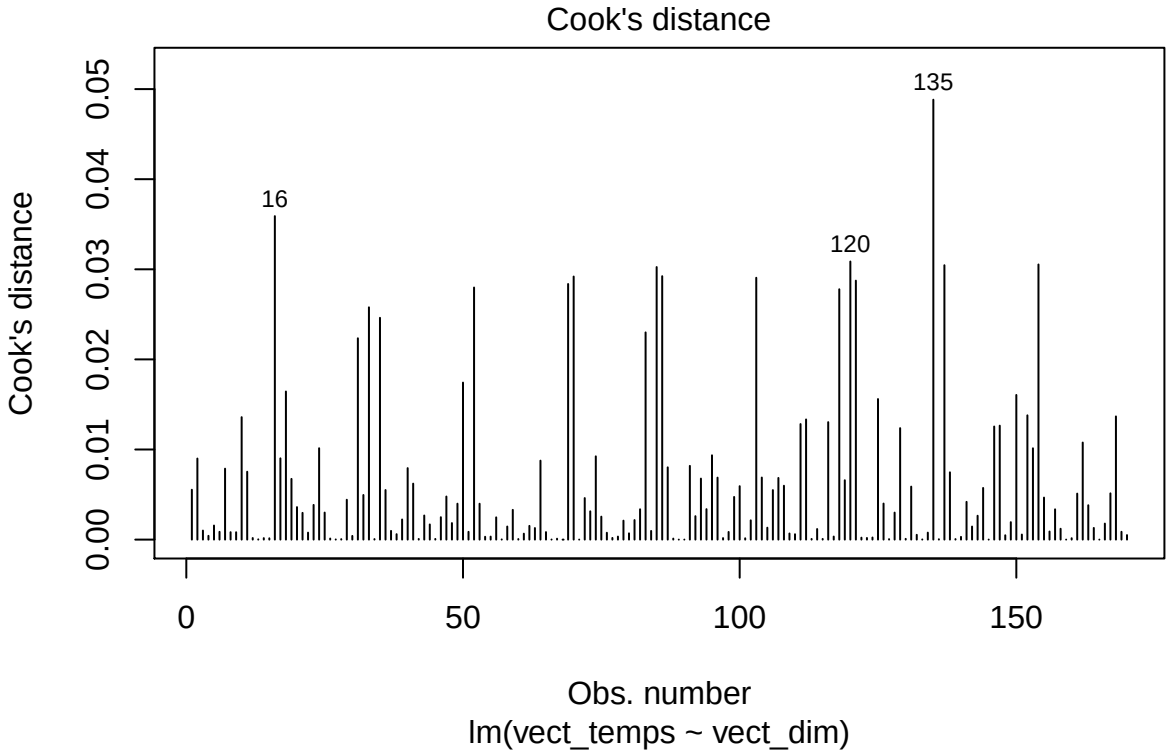
- L'analyse de a, permet d'établir un premier résultat quantifiant la significativité du modèle. En effet nous allons tester la significativité de a via le test statistique: $(H0) : a = 0$ contre $(H1) : a \neq 0$. La p-value de celui-ci ce retrouve dans le tableau summary(temps.lm) et est $2.2e - 16$. Nous sommes donc capable d'affiirmer avec un risque de de moins que 0.1% (chiffre arbitraire plus grand que $2.2e - 14$) que a est significatif.

- L'analyse de b est la moins importante. Il nous indique seulement l'importance de l'intercept. Le test statistique est analogique à a. Ca p-value est aussi $2.2e - 16$ Nous sommes donc capable d'affiirmer avec un risque de de moins que 0.1% (chiffre arbitraire plus grand que $2.2e - 14$) que l'intercept est utile.

- Nous pouvons maintenant passé a l'analyse des résidus:
  - Pour ceci nous allons tous d'abord nous intéréssé a plusieurs graphique:



-



- étude des hypothèses sur les résidus.

PAS SUR QUE LE PARAMETRE SOIT temps.lm

(H0) les résidus suivent une loi normale (H1) les résidus ne suivent pas une loi normale

On prend un risque alpha=5%

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(temps.lm)
## W = 0.98558, p-value = 0.07675
```

```
## [1] "p-valeur >= alpha"
## [1] "On ne peut pas rejeter H0"
```

On rejette H0, donc nous pouvons affirmer que les résidus ne suivent pas une loi normale.

2.2. Comportement par rapport au nombre de sommets : étude du comportement moyen

Récupération du temps moyen.

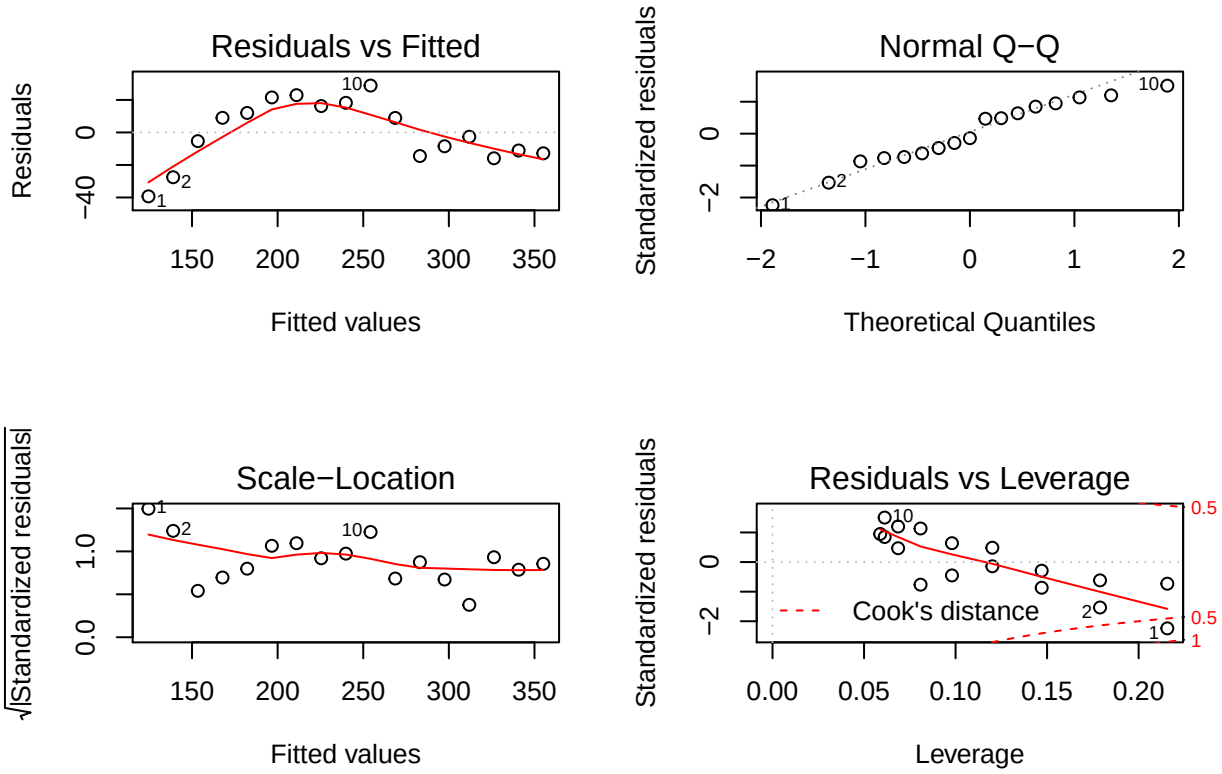
Ajustement du modèle linéaire de $\log(temps.moy)^2$ en fonction de n.

```
##
## Call:
## lm(formula = vect_temps_moy ~ vect_dim_moy)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -39.299 -12.829  -2.645  16.229  28.830
##
```

```
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   66.9210    12.7079   5.266 9.50e-05 ***
## vect_dim_moy  14.4118     0.9804  14.699 2.58e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.8 on 15 degrees of freedom
## Multiple R-squared:  0.9351, Adjusted R-squared:  0.9308
## F-statistic: 216.1 on 1 and 15 DF,  p-value: 2.583e-10
```

Analyse de la validité du modèle :

- pertinence des coefficients et du modèle



- étude des hypothèses sur les résidus.

PAS SUR QUE LE PARAMETRE SOIT temps.lm_moy

(H0) les résidus suivent une loi normale (H1) les résidus ne suivent pas une loi normale

On prend un risque alpha=5%

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(temps.lm_moy)
## W = 0.9601, p-value = 0.6334
```

```
## [1] "p-valeur >= alpha"
## [1] "On ne peut pas rejeter H0"
```

On ne peut pas rejeter H0. Donc on peut assurer avec un risque de 5% que les résidus suivent une loi normale. Validité du modèle ???

2.3. Comportement par rapport à la structure du graphe

Lecture du fichier 'DonneesTSP.csv'.

```
data.graph <- data.frame(read.csv('DonneesTSP.csv'))
data.graph$dim<-sqrt(data.graph$dim)
str(data.graph)
```

```
## 'data.frame':    70 obs. of  8 variables:
##  $ tps      : num  53692 144081 997803 2553322 6333009 ...
##  $ dim      : num  2 2.45 2.83 3.16 3.46 ...
##  $ mean.long: num  0.391 0.442 0.334 0.276 0.254 ...
##  $ mean.dist: num  0.665 0.592 0.537 0.506 0.502 ...
##  $ sd.dist  : num  0.276 0.259 0.246 0.238 0.227 ...
##  $ mean.deg : num  3 5 7 9 11 13 15 17 19 3 ...
##  $ sd.deg   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ diameter : num  1 1 1 1 1 1 1 1 1 1 ...
```

Ajustement du modèle linéaire de $\log(temps.moy)^2$ en fonction de toutes les variables présentes. Modèle sans constante.

```
model.complete <- lm(log(tps)~., data = data.graph)
```

```
(step(model.complete))
```

```
## Start:  AIC=-165.23
## log(tps) ~ dim + mean.long + mean.dist + sd.dist + mean.deg +
##     sd.deg + diameter
##
##             Df Sum of Sq     RSS      AIC
## - diameter   1    0.0145  5.2711 -167.038
## <none>                    5.2566 -165.230
## - sd.deg     1    0.2182  5.4748 -164.384
## - mean.dist  1    0.3014  5.5581 -163.327
## - mean.deg   1    0.8757  6.1324 -156.444
## - mean.long  1    3.6951  8.9517 -129.965
## - sd.dist    1    4.4335  9.6902 -124.417
## - dim        1   17.3311 22.5877  -65.176
##
## Step:  AIC=-167.04
## log(tps) ~ dim + mean.long + mean.dist + sd.dist + mean.deg +
##     sd.deg
##
##             Df Sum of Sq     RSS      AIC
## <none>                    5.2711 -167.038
## - sd.deg     1    0.2065  5.4776 -166.349
## - mean.dist  1    0.6554  5.9265 -160.835
## - mean.deg   1    0.9820  6.2531 -157.080
## - mean.long  1    3.8220  9.0931 -130.869
## - sd.dist    1    4.9133 10.1844 -122.935
## - dim        1   18.7788 24.0499  -62.785
```

```
##
## Call:
## lm(formula = log(tps) ~ dim + mean.long + mean.dist + sd.dist +
##     mean.deg + sd.deg, data = data.graph)
##
```

```
## Coefficients:
## (Intercept)          dim    mean.long    mean.dist      sd.dist     mean.deg
##    6.396008     3.444077    -4.854857    -0.002284     0.004883    -0.140823
##      sd.deg
##    0.126916
```

```
new_model <- lm(formula = log(tps) ~ dim + mean.long + mean.dist + sd.dist +
    mean.deg + sd.deg, data = data.graph)
shapiroTest_aic<-shapiro.test(residuals(new_model))
print(shapiroTest_aic)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(new_model)
## W = 0.98094, p-value = 0.3641
```

Mise en œuvre d'une sélection de variables pour ne garder que les variables pertinentes.

Analyse de la validité du modèle :

- pertinence des coefficients et du modèle,
- étude des hypothèses sur les résidus.