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 :

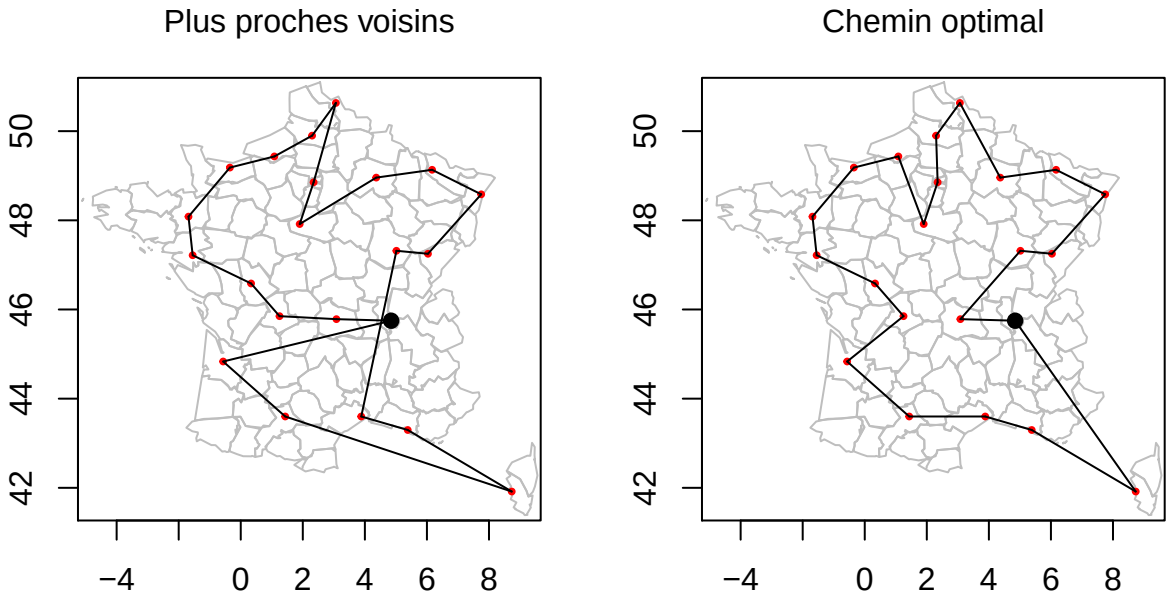
```
## 'data.frame':    22 obs. of  5 variables:
##  $ EU_circo : Factor w/ 7 levels "Centre","Est",..: 6 6 4 2 7 4 2 1 2 4 ...
##  $ region   : Factor w/ 22 levels "Alsace","Aquitaine",..: 22 9 19 10 2 4 8 3 5 17 ...
##  $ ville    : Factor w/ 22 levels "Ajaccio","Amiens",..: 11 1 2 3 4 5 6 7 8 9 ...
##  $ 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

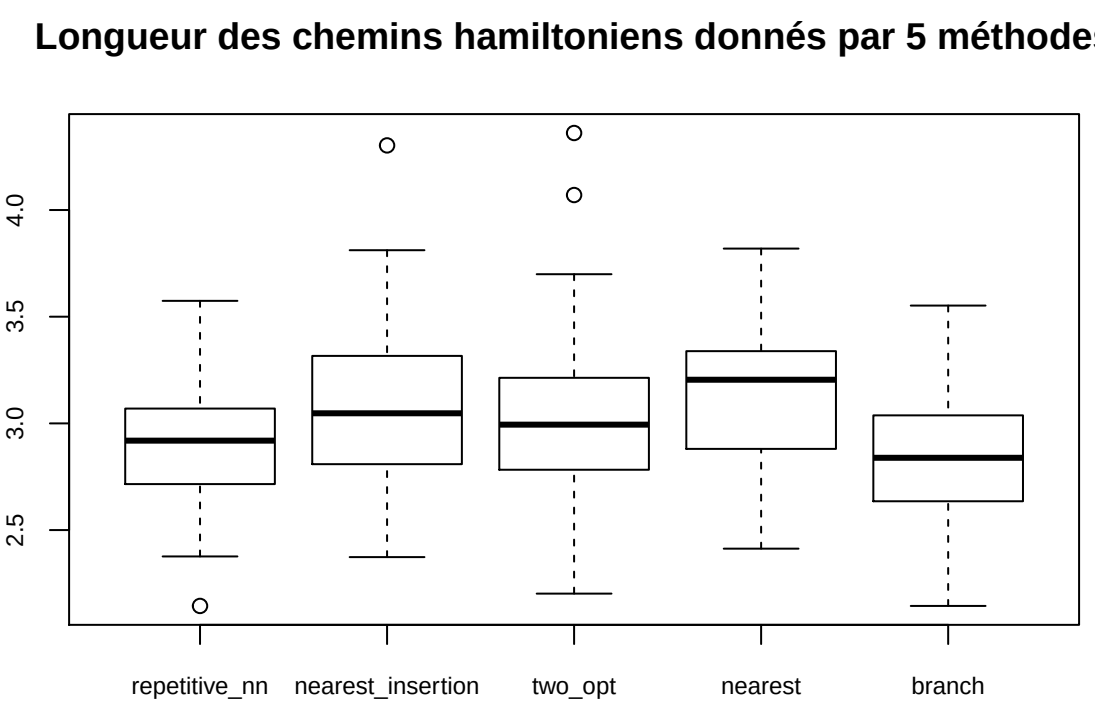
Dans cette partie, nous souhaitons comparer les méthodes repetitive_nn, nearest_insertion, two_opt, nearest, et branch. Pour cela, nous allons générer des graphes aléatoires de 10 sommets, et tester les longueurs des chemins calculés et le temps de calcul des différentes méthodes.

```
      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 médiane 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.

(H_0) $m_{nn} - m_b \leq 0 \Leftrightarrow m_{nn} \leq m_b$
(H_1) $m_{nn} - m_b > 0 \Leftrightarrow m_{nn} > m_b$

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 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 de Branch&Bound.

Tests 2 à 2

On souhaite maintenant comparer 2 à 2 les longueurs moyennes des chemins hamiltoniens obtenus par les 5 méthodes vues précédemment.

On réalise donc un test sur l'espérance de chaque méthode.
Soit i, j deux méthodes différentes .Notre hypothèse nulle (H0) est que la moyenne des chemins hamiltoniens obtenus avec la méthode i est égale à la moyenne des chemins hamiltoniens obtenus avec la méthode j. Notre hypothèse alternative (H1) est que la moyenne des chemins hamiltoniens obtenus avec la méthode i est différente de la moyenne des chemins hamiltoniens obtenus avec la méthode j.

$(H_0) mi = mj$
$(H_1) mi \neq mj$

Nous avons lancé 10 tests simultanés, et obtenus les résulats suivants:

```
##
##  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
```

Nous allons tester si au seuil de 5%, notre hypothèse H0 est vérifiée.

Si on accepte un risque alpha=5%, on rejette notre hypothèse nulle (H_0) si la p_{valeur} obtenue à l'indice [i,j] est inférieure à alpha.
Donc, si la valeur à l'indice [i,j] est inférieure à alpha, nous pouvons affirmer avec un risque de 5% que la moyenne des chemins hamiltoniens obtenus avec la méthode i est différente de celle obtenue avec la méthode j.

En appliquant ce principe à nos résultats, nous pouvons dire que :

- les méthodes nearest et branch ont des moyennes de chemins calculés différentes
- les méthodes nearest_insertion et branch ont des moyennes de chemins calculés différentes - les méthodes nearest et repetitive_nn ont des moyennes de chemins calculés différentes

Pour les autres méthodes, nous ne pouvons pas rejeter l'hypothèse d'après laquelle la moyenne des chemins hamiltoniens obtenus avec la méthode i est égale à la moyenne des chemins hamiltoniens obtenus avec la méthode j.

1.2. Temps de calcul

Nous souhaitons maintenant comparer les temps d'éxécution des différentes méthodes de calcul de longueur de chemin hamiltonien sur 20 graphes de 10 sommets générés aléatoirement.

Nous avons utilisé la fonction benchmark pour réaliser des statistiques d'exécution pour chaque méthode.

Nous avons réalisé des tests sur les temps moyens d'exécution de chaque méthode :
Soit i, j deux méthodes différentes .Notre hypothèse nulle H_0 est que le temps moyen d'exécution de la méthode i est égale au temps moyen d'exécution de la méthode j. Notre hypothèse alternative (H_1) est que le temps moyen d'exécution de la méthode i est différent du temps moyen d'exécution de la méthode j.

$(H_0) mi = mj$
$(H_1) mi != mj$

Le résultat de ces tests est représenté par une lettre dans la colonne cld du tableau ci-dessous. Une même lettre est attribuée aux méthodes pour lequelles H0 n'est pas rejetée. Les lettres sont classées par ordre croissant de temps d'exécution, ainsi un algorithme classé 'a' est plus rapide qu'un algoriyhme classé 'b', etc ...
Deux méthodes ayant des lettres différentes ont donc des temps d'exécution moyens différents.

```
## Unit: microseconds
##                                 expr      min       lq        mean     median
##      TSPsolve(couts, "repetitive_nn") 6509.765 8484.247 10490.37725 10518.8725
##  TSPsolve(couts, "nearest_insertion")  965.159 1455.647  2099.74300  1707.7525
##            TSPsolve(couts, "two_opt")  721.619  820.076  1313.45995  1003.1485
##            TSPsolve(couts, "nearest")   19.053   25.534    34.45095    29.5375
##             TSPsolve(couts, "branch") 2092.894 3056.671  5382.26545  4421.8375
##          uq       max neval  cld
##  11820.6865 15908.255    20    d
##   2193.2970  6287.669    20  b
##   1217.1685  4485.868    20 ab
##     35.9465   115.935    20 a
##   6249.1015 16234.025    20   c
```

Nous pouvons remarquer que les méthodes nearest_insertion, two_opt et nearest ont des temps d'exécution moyens similaires. Les méthodes sont classées 'a', ce qui montre que ce sont les méthodes les plus rapides des 5 méthodes proposées. De plus, les 3 méthodes ayant le même classement, on ne peut pas rejeter le fait que

les temps moyens d'exécution de ces méthode sont équivalents. Il n'a, en tout cas, pas été mis en évidence que ces méthodes avaient des temps d'exécution significativement différents.
Les méthodes repetitive_nn et branch ont une durée d'exécution moyenne supérieure aux autres.

La méthode repetitive_nn est classée 'c', ce qui fait que l'on peut affirmer que le temps moyen d'exécution de cette méthode est différent du temps moyen d'exécution des autres méthodes. De plus, comme les lettres {a, b, c, ...} sont attribuées en fonction du temps d'exécution ('a' pour la plus rapide, ...), on peut en déduire que le temps moyen d'exécution de repetitive_nn est plus important que le temps moyens des autres méthodes.

Par la même réflexion que celle faite précédemment, nous pouvons remarquer que branch est classé 'b', et a un temps d'exécution moyen plus long que les trois méthodes classées 'a'. Cependant, il a un temps moyen d'exécution inférieur à la méthode repetitive_nn.

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

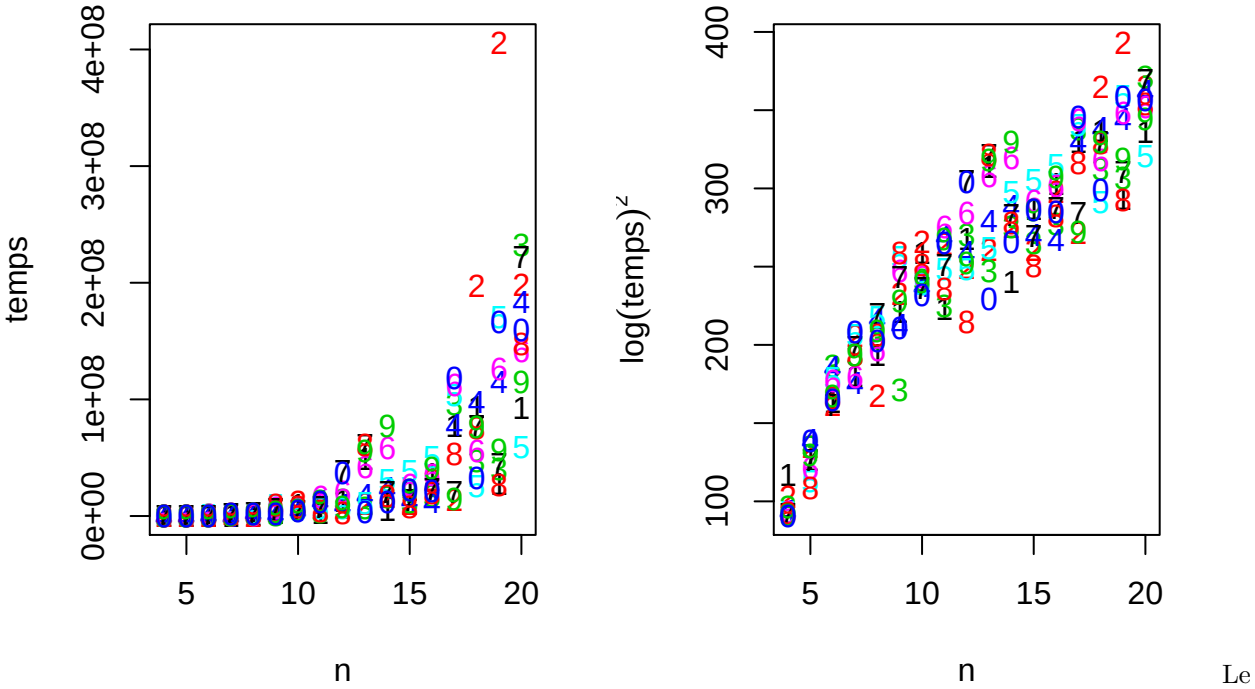
Nous nous intéressons désormais à l'algorithme Branch&Bound. Nous allons étudier sa complexité en fonction du graphe sur lequel il est appliqué.

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

Création du modèle linéaire

Dans un premier temps nous allons créer une matrice temps calculant les temps mis par l'algorithme Branch&Bound pour calculer les chemins hamiltoniens sur un graphe de n sommets généré aléatoirement. Nous réalisons 10 fois l'algorithme sur un graphe à n sommets, et nous faisons varier n de 4 à 20. La matrice obtenue est donc de dimension $17 * 10$.

A l'aide de ces données, nous pouvons afficher le graphe du temps mis par l'algorithme en fonction du nombre de noeuds du graphe, et le graphe de $\log(temps)^2$ en fonction nombre de noeuds du graphe. Cela nous donne les résultats ci-dessous.



Le graphe du temps mis par l'algorithme en fonction du nombre de noeuds du graphe semble suivre une courbe exponentielle. Cette hypothèse est soutenue par le 2ème graphe.

Nous avons ensuite ajusté le modèle linéaire de $\log(temps)^2$ en fonction de n, pour en récupérer les principales charactéristiques. Nous avons obtenu le résultat suivant :

```
##
## Call:
## lm(formula = vect_temps ~ vect_dim)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -54.418 -19.850   0.029  17.821  57.279
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   83.938      5.483   15.31   <2e-16 ***
## vect_dim      13.846      0.423   32.73   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 27.02 on 168 degrees of freedom
## Multiple R-squared:  0.8645, Adjusted R-squared:  0.8636
## F-statistic:  1071 on 1 and 168 DF,  p-value: < 2.2e-16
```

Nous pouvons remarquer qu'il y a une relation linéaire entre $\log(temps)$š et n, puisque le test de Fisher ne rejette pas le modèle linéaire. En effet ce test permet de rejeter le fait que tous les coefficients du modèle sont nuls.
De plus, $R^2 =$

```
## [1] 0.8644508
```

Rš étant proche de 1, une grande partie des données suivent le modéle linéaire. On peut donc en conclure qu'il y a une relation linéaire entre $\log(temps)^2$ et n.

De ce fait, on peut en déduire que :
$\exists \alpha, \beta$ tels que $\log(temps)$š $= \alpha * n + \beta$
soit $\log(temps) = \pm \sqrt{(\alpha * n + \beta)}$
donc $temps = \exp(\pm \sqrt{(\alpha * n + \beta)})$

On peut en déduire que Branch&Bound semble avoir une complexité temporelle en $\exp(n)$.

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 à savoir si les coéfficients et donc le modèle sont pertinents. Nous allons tous d'abord analyser la pertinence des coefficients puis celle du modèle en géneral.
* Sois $\alpha = 5\%$

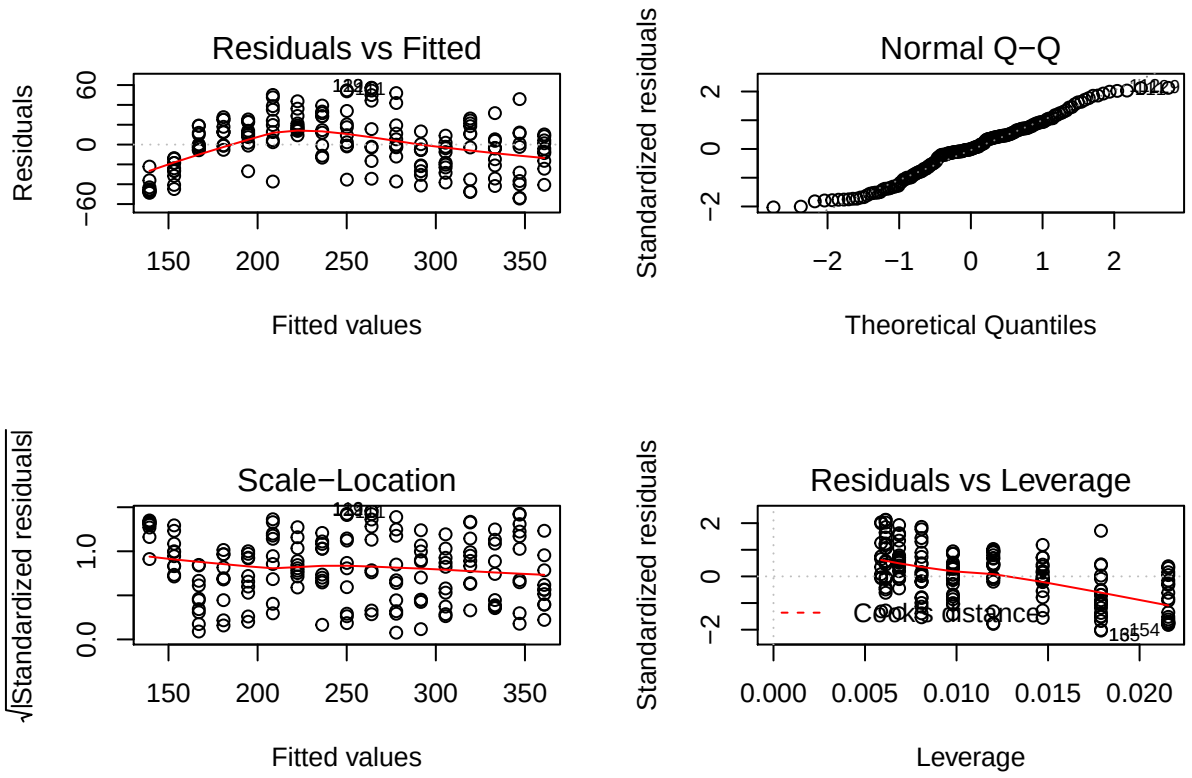
- L'analyse de a, permet d'établir un premier résultat quantifiant la significativité du modèle. En effet nous allons tester la significativité de a via le test statistique: $(H_0) : a = 0$ contre $(H_1) : a \neq 0$. La p-value de celui-ci ce retrouve dans le tableau summary(temps.lm) et est

```
##     vect_dim
## 8.241621e-75
```

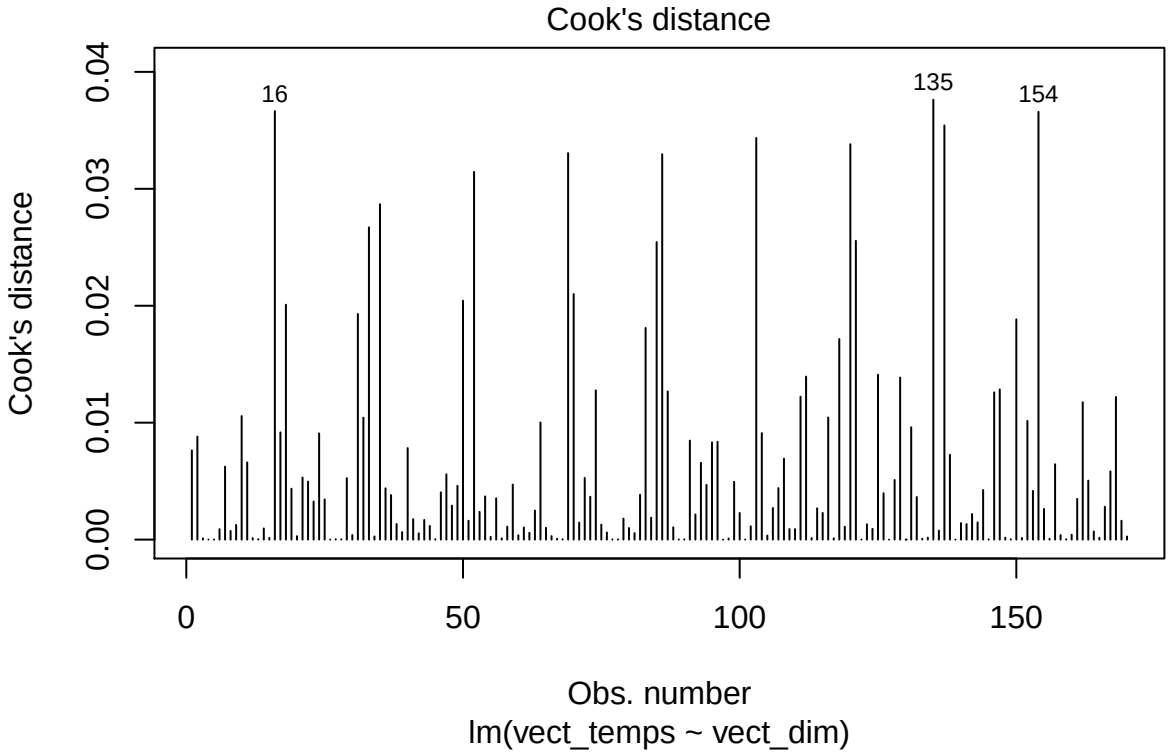
Nous pouvons donc rejeter $H0$ et affirmer avec 5% de risque que a est significatif. * L'analyse de b est la moins importante. Il nous indique seulement l'importance de l'intercept. Le test statistique est analogique à a. Sa p-value est

```
##  (Intercept)
## 1.097929e-33
```

Nous pouvons donc rejeter $H0$ et affirmer avec 5% de risque que b est significatif.
* Nous pouvons maintenant passer à l'analyse des résidus: * Pour ceci nous allons tous d'abord nous intéresser à plusieurs graphique:



- Residuals vs Fitted: La courbe n'est pas complètement horizontale. Il y'a donc un léger effet d'échelle.
- Normal Q-Q: les points sont proches de la bissectrice, la distribution des résidus est donc similaire à la distribution normale. Nous voyons une légère séparation au niveau des queues des distribution.
- Scale Location: cette courbe représente la même chose que la première seulement avec des résisus normalisés. On remarque que la courbe est bien honrizontale et que le léger effet d'échelle disparaît.
- Pour la distance de cook nous avons préféré prendre le graphique suivant: Nous voyons qu'aucun résidu a une distance plus grande que 0.05 et que la plupart ont une distance inferieure à 0.01. Ceci montre que le modèle est bien choisi.



- Il est aussi possible d'éffectuer un test statistique sur les résidus. En effet le fait qu'ils suivent une loi normale indique la qualité du modèle.
- Définissons:
  - $(H0)$ les résidus suivent une loi normale
  - $(H1)$ les résidus ne suivent pas une loi normale
- Pour tester ceci nous pouvons efféctuer un test de shapiro.

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(temps.lm)
## W = 0.97925, p-value = 0.01202
```

```
## [1] "p-valeur < alpha"
## [1] "On peut rejeter H0"
```

On ne peut pas rejetter H_0, donc nous pouvons affirmer avec un risque de 5% que les résidus ne suivent pas une loi normale. Ce qui indique un modèle pertinent.

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

L'explication des résultats étant similaire à 2.1, nous allons simplement afficher nos résultats.
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
## -43.66 -13.27   2.79  15.76  28.98
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)     84.81      13.10   6.477 1.04e-05 ***
## vect_dim_moy    14.21       1.01  14.063 4.81e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 20.41 on 15 degrees of freedom
## Multiple R-squared:  0.9295, Adjusted R-squared:  0.9248
## F-statistic: 197.8 on 1 and 15 DF,  p-value: 4.81e-10
```

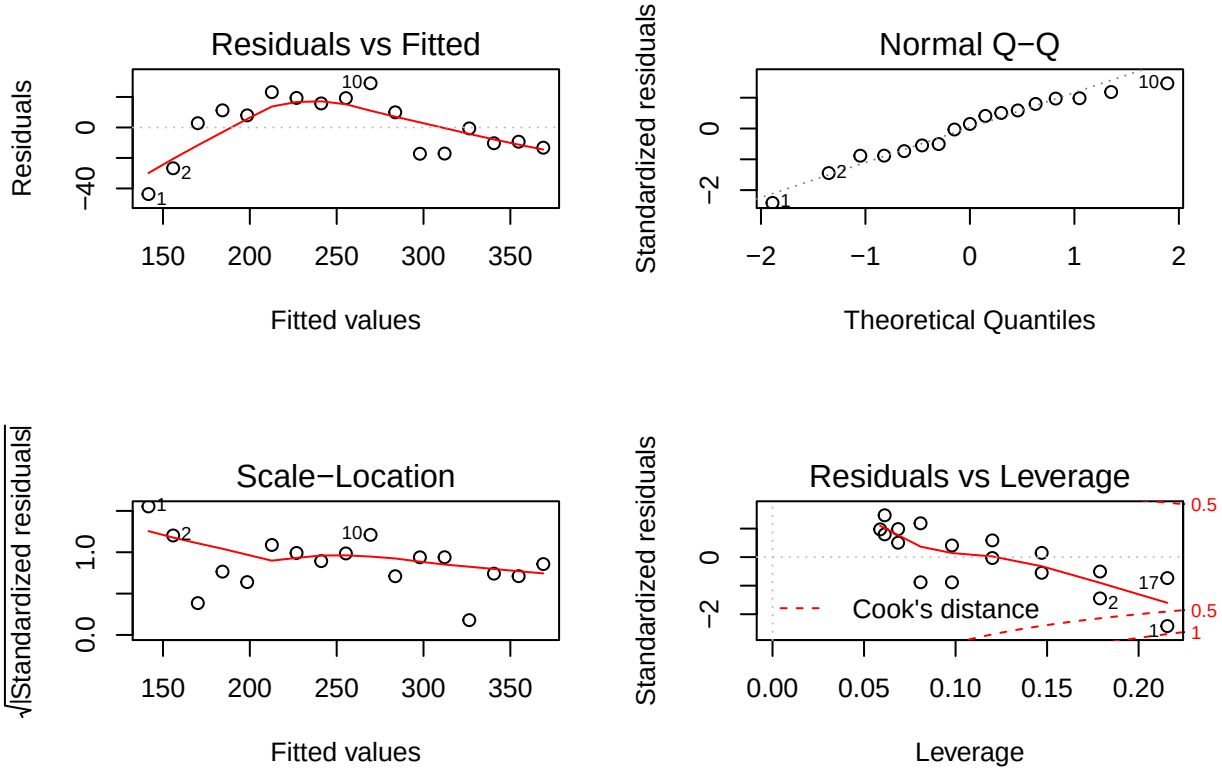
Analyse de la validité du modèle :

- *a* pertinent $p_{value} = $

```
## vect_dim_moy
## 4.809554e-10
```

- *b* pertinent $p_{value} = $

```
##  (Intercept)
## 1.044521e-05
```



- Residuals vs Fitted: La courbe n'est pas du tout horizontale. Il y'a donc un important effet d'échelle.
- Normal Q-Q: Les distributions sont identiques.
- Scale Location: L'effet d'échelle disparait. Le nuage de points est sans structure. Ce qui indique la qualité du modèle.

- Il est aussi possible d'effectuer un test statistique sur les résidus. En effet le fait q'ils suivent une loi normale indique le bon fit du modèle.
- Définissons:
  - $(H0)$ les résidus suivent une loi normale
  - $(H1)$ les résidus ne suivent pas une loi normale
- Pour tester ceci nous pouvons effectuer un test de shapiro.

On prend un risque alpha=5%

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(temps.lm_moy)
## W = 0.9606, p-value = 0.6427
```

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

On ne peut rejetter H_0, donc nous pouvons affirmer avec un risque de 5% que les résidus suivent une loi normale. Ce qui indique un modèle pertinent.

2.3. Comportement par rapport à la structure du graphe

- D'après nous les variables non pertinentes sont: diameter, mean.dist, sd.dist, mean.long. En effet tous ces variables s'intérèssent seulement aux coûts des arrêtes. Ces derniers n'ont pas d'importance dans le temps de calcul (calculer le chemin avec une arrete de 5 ou de 1000 reviens à la même chose).
- Ajustement du modèle linéaire de log(*temps.moy*) en fonction de toutes les variables présentes. Modèle sans constante. Nous allons d'abord procéder à un test de fisher avec toutes les variables.

```
##
## Call:
## lm(formula = log(tps) ~ ., data = data.graph)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.78776 -0.15715  0.01542  0.17260  0.65036
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.4903426  0.5450715  11.907  < 2e-16 ***
## dim          3.4191719  0.2391476  14.297  < 2e-16 ***
## mean.long   -4.8152962  0.7294055  -6.602 1.05e-08 ***
## mean.dist   -0.0020048  0.0010633  -1.886  0.06404 .
## sd.dist      0.0048105  0.0006652   7.231 8.55e-10 ***
## mean.deg    -0.1367369  0.0425459  -3.214  0.00208 **
## sd.deg       0.1399515  0.0872430   1.604  0.11376
## diameter    -0.0646816  0.1566329  -0.413  0.68107
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2912 on 62 degrees of freedom
## Multiple R-squared:  0.986,  Adjusted R-squared:  0.9844
## F-statistic: 622.6 on 7 and 62 DF,  p-value: < 2.2e-16
```

- Soit $\alpha = 1\%$

- Le test de fisher à une p_{value} de $2x10^{-}16$ et nous permet donc de rejeter $H0$ avec un risque de 1%.

Calcul AIC

- Dans l'étape suivante nous allons procéder au calcul de l'AIC pour les variables de notre modèle. Ceci nous permettera de supprimer les variables non pertinentes.

```
## 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
```

- La variable diameter a été supprimée du modèle. Il reste sd.dist, mean.dist et mean.long, qui nous paraissent peut pertinente.

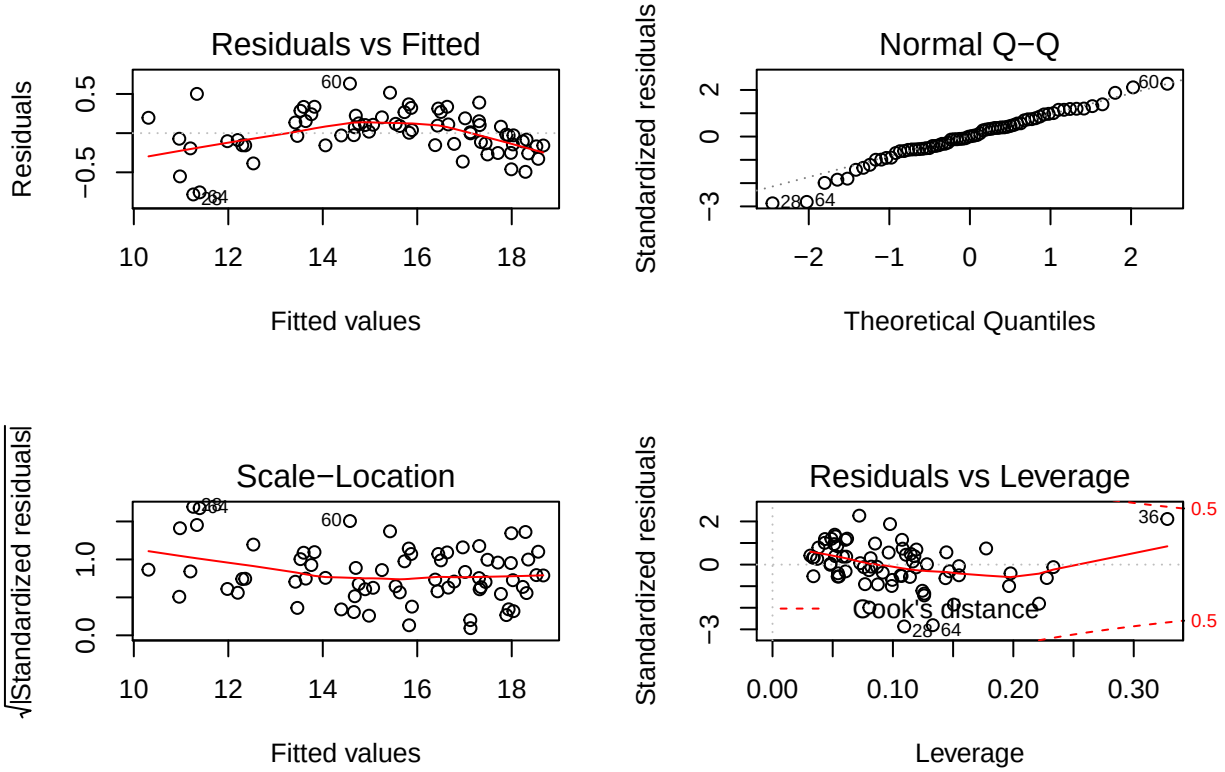
Test Fisher

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

```
##      Min       1Q   Median       3Q      Max
## -0.78246 -0.15445  0.00111  0.18027  0.63156
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.3960078  0.4916227  13.010  < 2e-16 ***
## dim          3.4440771  0.2298893  14.981  < 2e-16 ***
## mean.long   -4.8548566  0.7183110  -6.759 5.25e-09 ***
## mean.dist   -0.0022837  0.0008160  -2.799  0.00680 **
## sd.dist      0.0048833  0.0006372   7.663 1.39e-10 ***
## mean.deg    -0.1408227  0.0411061  -3.426  0.00108 **
## sd.deg       0.1269156  0.0807943   1.571  0.12123
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2893 on 63 degrees of freedom
## Multiple R-squared:  0.9859, Adjusted R-squared:  0.9846
## F-statistic:   736 on 6 and 63 DF,  p-value: < 2.2e-16
```

- Sois $\alpha = 1\%$.
- D'après le test de fisher nous pouvons affirmer que le test est pertinent avec un risque de 1% ($p_{value} = 2.2 * 10^{-16}$). Cependant la p_{value} n'a pas changé en supprimant la variable diameter.

Plots



- Residuals vs Fitted: La courbe n'est pas horizontale. Il y'a donc un effet d'échelle. Ceci indique un fit moyen.
- Normal Q-Q: Les distributions sont identiques, avec de légères différences sur les queues.
- Scale Location: L'effet d'échelle disparaît. Le nuage de points est sans structure. Ce qui indique la qualité du modèle.

- Cook Distance: les points sont à une distance de cook faible, ceci indique un bon fit.

Test Shapiro

- Nous pouvons aussi analyser les résidus, pour voir s'ils suivent une distribution normale, avec le test de Shapiro.

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

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

On ne peut pas rejetter H_0, donc nous pouvons affirmer avec un risque de 5% que les résidus suivent une loi normale. Ce qui indique un modèle pertinent.