Mutations

Anthony LEZIN

8/19/2020

I. Régression linéaire

0. Préparation des données

```
mut=read.csv("/Users/anthonylezin/Desktop/projets_stat/Projet_Inf. baysienne/mutations2.csv",header=TRU
```

Je décide de renommer les variables pour certaines applications ultérieures.

```
nom=c("1","vil","3","4","Mat","Barre","ef_l","ef_es","ef_s","tbrl","tbres","tbrs","tral","traes","tras"

#équivalences entre anciens noms et nouveaux noms
equivalences = cbind(names(mut),(nom))
rownames(equivalences)=c()
```

voici le tableau des équivalences pour les noms des variables :

```
kable(equivalences)
```

code_etablissement	1
ville	vil
etablissement	3
commune	4
Matiere	Mat
Barre	Barre
effectif_presents_serie_l	ef_l
effectif_presents_serie_es	ef_es
effectif_presents_serie_s	ef_s
taux_brut_de_reussite_serie_l	tbrl
taux_brut_de_reussite_serie_es	tbres
taux_brut_de_reussite_serie_s	tbrs
taux_reussite_attendu_serie_l	tral
taux_reussite_attendu_serie_es	traes
taux_reussite_attendu_serie_s	tras
effectif_de_seconde	ef_2
effectif_de_premiere	ef_1
taux_acces_brut_seconde_bac	tab2b
taux_acces_attendu_seconde_bac	taa2b
taux_acces_brut_premiere_bac	tab1b
taux_acces_attendu_premiere_bac	taa1b
taux_brut_de_reussite_total_series	tbrts

taux_reussite_attendu_total_series	trats

1. Régression linaire baysienne et interprétation des coefficients

```
mut2=mut[,6:23]
names(mut2)=nom[6:23]
```

On simule 10 000 itérations de la loi à postériori par la méthode de Monte-Carlo associées aux chaîne de Markov (MCMC). Par cette méthode, j'obtiens une matrice de dimension (10 000,19) dontla colonne i fournit une estimation de la loi à postériori de la i^{me} covariable.

```
reg0=MCMCregress(Barre~., data=mut2)
summary(reg0)
```

```
##
## Iterations = 1001:11000
## Thinning interval = 1
## Number of chains = 1
## Sample size per chain = 10000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                    Mean        SD  Naive SE Time-series SE
## (Intercept) -4.712e+02 5.623e+02 5.623e+00      5.571e+00
## ef_l         7.479e-01 1.639e+00 1.639e-02      1.639e-02
## ef_es        2.930e-01 1.222e+00 1.222e-02      1.222e-02
## ef_s         2.685e-03 1.020e+00 1.020e-02      1.020e-02
## tbrl         3.067e+00 2.546e+00 2.546e-02      2.546e-02
## tbres        4.719e+00 4.208e+00 4.208e-02      4.208e-02
## tbrs         9.281e+00 6.322e+00 6.322e-02      6.070e-02
## tral        -1.423e+01 6.952e+00 6.952e-02      6.952e-02
## traes        3.928e+00 8.264e+00 8.264e-02      8.264e-02
## tras        -4.201e+00 9.597e+00 9.597e-02      9.597e-02
## ef_2         5.518e-02 6.239e-01 6.239e-03      6.239e-03
## ef_1        -3.564e-01 7.169e-01 7.169e-03      7.169e-03
## tab2b        1.071e+01 5.695e+00 5.695e-02      5.695e-02
## taa2b       -6.940e+00 9.029e+00 9.029e-02      8.992e-02
## tab1b       -2.040e+01 1.065e+01 1.065e-01      1.065e-01
## taa1b        3.460e+01 1.916e+01 1.916e-01      1.887e-01
## tbrts       -5.160e+00 1.289e+01 1.289e-01      1.261e-01
## trats       -4.571e+00 2.217e+01 2.217e-01      2.217e-01
## sigma2       1.792e+05 1.147e+04 1.147e+02      1.213e+02
##
## 2. Quantiles for each variable:
##
##                   2.5%        25%        50%        75%      97.5%
## (Intercept) -1.562e+03 -8.509e+02 -4.759e+02 -1.065e+02  6.576e+02
## ef_l        -2.435e+00 -3.562e-01  7.413e-01  1.859e+00  3.948e+00
## ef_es       -2.143e+00 -5.216e-01  2.956e-01  1.109e+00  2.691e+00
## ef_s        -1.953e+00 -7.007e-01 -5.803e-03  6.938e-01  2.017e+00
```

```
## tbrl        -1.928e+00  1.354e+00  3.084e+00  4.780e+00  8.089e+00
## tbres       -3.502e+00  1.911e+00  4.690e+00  7.585e+00  1.300e+01
## tbrs        -3.367e+00  5.048e+00  9.328e+00  1.350e+01  2.175e+01
## tral        -2.782e+01 -1.888e+01 -1.416e+01 -9.566e+00 -7.551e-01
## traes       -1.206e+01 -1.708e+00  4.087e+00  9.441e+00  2.033e+01
## tras        -2.315e+01 -1.077e+01 -4.170e+00  2.229e+00  1.464e+01
## ef_2        -1.158e+00 -3.660e-01  5.770e-02  4.756e-01  1.266e+00
## ef_1        -1.787e+00 -8.401e-01 -3.521e-01  1.309e-01  1.026e+00
## tab2b       -4.918e-01  6.848e+00  1.078e+01  1.454e+01  2.172e+01
## taa2b       -2.501e+01 -1.310e+01 -6.954e+00 -8.326e-01  1.075e+01
## tab1b       -4.107e+01 -2.765e+01 -2.030e+01 -1.317e+01  2.314e-01
## taa1b       -2.656e+00  2.140e+01  3.449e+01  4.743e+01  7.209e+01
## tbrts       -3.044e+01 -1.391e+01 -5.068e+00  3.474e+00  1.986e+01
## trats       -4.774e+01 -1.966e+01 -4.622e+00  1.047e+01  3.839e+01
## sigma2       1.580e+05  1.713e+05  1.788e+05  1.865e+05  2.032e+05
```

Dans le 1er tableau :

- la 1ère colonne fournit une estimation du vecteur $\hat{\beta}$ contenant les valeurs $\hat{\beta}_i$, estimteur des covaraiables obtenus dans la MCMC. Ils sont calculés à partir de la moyenne des estimations.
- la 2nde colonne fournit une estimation de l'erreur, les écart-type des lois à postériori.
- un 2nd tableau fournit les quantiles de la loi à postériori de chacun des paramètres. En effet, l'inférence baysienne permet d'estimer bien plus que des paramètres ponctuels. On obtient une estimation de la densité de la loi à postériori, ce qui permet d'estimer des paramètres de cette loi, des intervalles de confiance, etc.

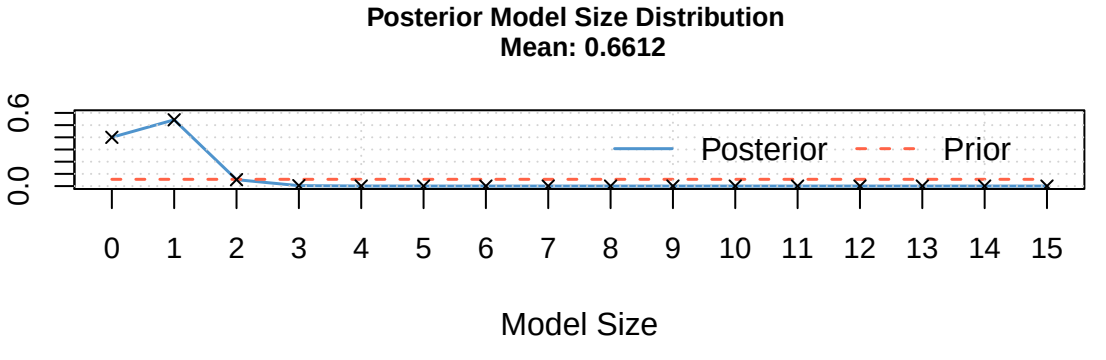
"Naïvement", 0 appartient à tous les intervalles de crédibilité (sauf peut-être "taux_reussite_attendu_serie_l") traduisant indirectement le fait que l'on ne peut pas ôter de manière significative des covariables de notre modèle.

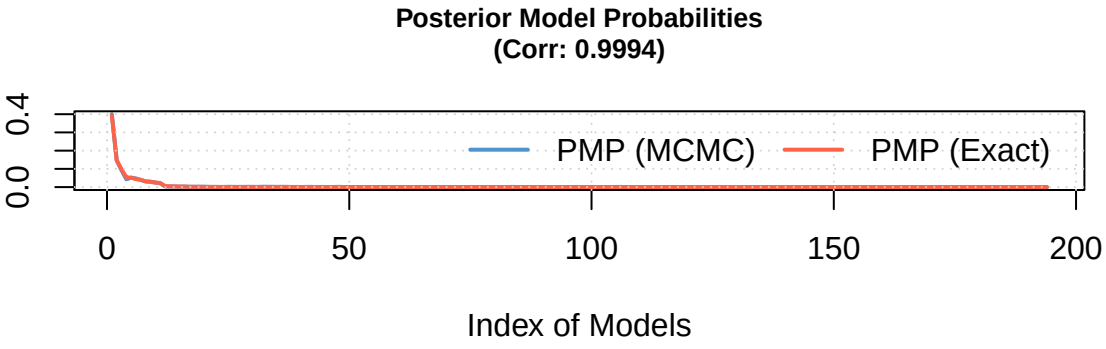
Ce critère n'est pas réellement adapté à la sélection de modèles.

```
reg0.bms=bms(Barre~., data=mut2, burn = 1e4, iter=1e5)
```

```
##           PIP     Post Mean    Post SD Cond.Pos.Sign Idx
## taa1b 0.16825  1.703016e+00 4.15206324     1.0000000  15
## taa2b 0.10290  7.893861e-01 2.51098666     1.0000000  13
## traes 0.06028  3.530452e-01 1.54821613     0.9805906   8
## tras  0.05287  2.745647e-01 1.29689208     0.9807074   9
## tbrts 0.04935  3.361022e-01 1.62155860     0.9993921  16
## trats 0.04610  2.645034e-01 1.63990656     0.9561822  17
## tab2b 0.03766  1.892149e-01 1.06648299     1.0000000  12
## tab1b 0.03514  2.188871e-01 1.34578792     0.9519067  14
## tbrs  0.03114  1.415339e-01 0.92084174     1.0000000   6
## tbres 0.02658  1.123530e-01 0.77112762     1.0000000   5
## tral  0.01030 -4.496032e-02 0.75534709     0.1805825   7
## ef_es 0.00771  2.472388e-03 0.06530594     0.7574578   2
## ef_s  0.00759  2.646741e-03 0.05127420     0.8511199   3
## tbrl  0.00703  1.138334e-02 0.21200379     1.0000000   4
## ef_1  0.00649 -7.200981e-06 0.01600616     0.4098613  11
## ef_l  0.00604  2.243180e-03 0.08009449     0.7417219   1
## ef_2  0.00574  1.266026e-04 0.01224232     0.4930314  10
##
## Mean no. regressors              Draws             Burnins                  Time
```

```
##            "0.6612"             "1e+05"             "10000"     "3.566827 secs"
##  No. models visited      Modelspace 2^K          % visited         % Topmodels
##              "8567"            "131072"              "6.5"               "100"
##           Corr PMP            No. Obs.        Model Prior             g-Prior
##            "0.9994"               "516"      "random / 8.5"               "UIP"
##     Shrinkage-Stats
##         "Av=0.9981"
##
## Time difference of 3.566827 secs
```



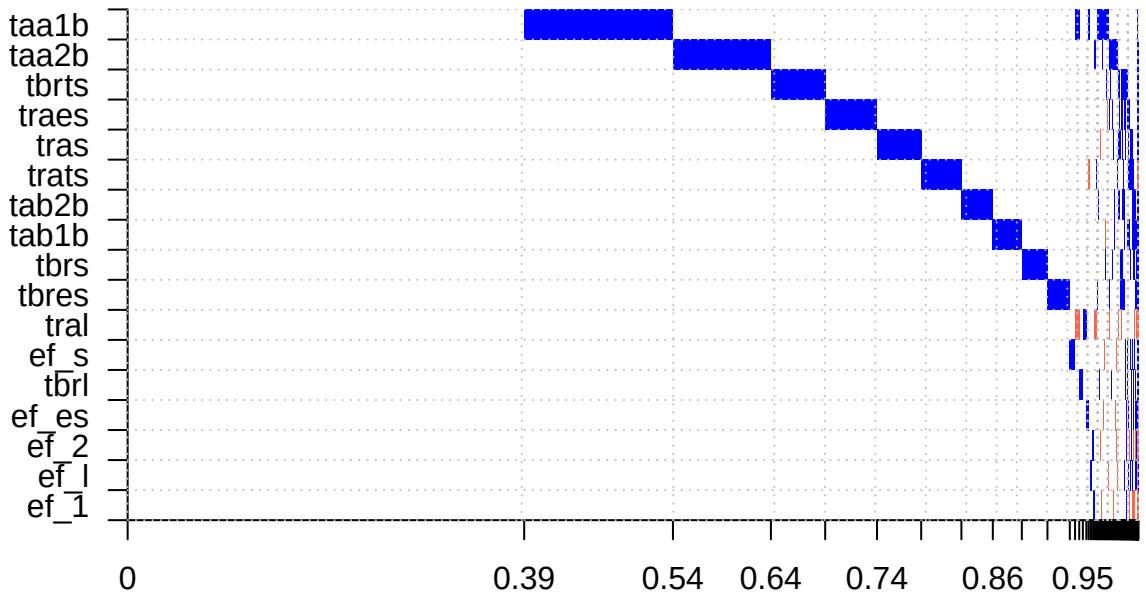


Après avoir "brulé" les premières itérations et augmenté un peu le nombre, l'indice $CorrPMP$ avoisinne la valeur 1. Les probabilités d'inclusion et celles données par l'algorithme MCMC sont donc relativement proches. Il n'est pas nécessaire d'augmenter encore le nombre d'itérations de la chaine.

Malheureusement, les proportions des itérations de l'algorithme MCMC passées par les différents modèles (représentées par les probabilités contenues colonne PIP) sont relativement faibles ! Etant donné qu'elles représentent les probabilités d'inclure les covariables en question dans le modèle, il semble qu'il sera délicat de déterminer une "sélection utile".

Par exemple, la covaiable "taux d'accès attendu en 1ère bac" est la plus "visitée" par la chaine. Elle apparaît dans moins de 16% de l'ensemble des modèles visités.

```
image(reg0.bms)
```




```
kable(topmodels.bma(reg0.bms)[,1:7])
```

	00000	00004	00010	00002	00200	00100	00001
ef_l	0.0000000	0.0000000	0.0000000	0.0000000	0.000000	0.0000000	0.0000000
ef_es	0.0000000	0.0000000	0.0000000	0.0000000	0.000000	0.0000000	0.0000000
ef_s	0.0000000	0.0000000	0.0000000	0.0000000	0.000000	0.0000000	0.0000000
tbrl	0.0000000	0.0000000	0.0000000	0.0000000	0.000000	0.0000000	0.0000000
tbres	0.0000000	0.0000000	0.0000000	0.0000000	0.000000	0.0000000	0.0000000
tbrs	0.0000000	0.0000000	0.0000000	0.0000000	0.000000	0.0000000	0.0000000
tral	0.0000000	0.0000000	0.0000000	0.0000000	0.000000	0.0000000	0.0000000
traes	0.0000000	0.0000000	0.0000000	0.0000000	1.000000	0.0000000	0.0000000
tras	0.0000000	0.0000000	0.0000000	0.0000000	0.000000	1.0000000	0.0000000
ef_2	0.0000000	0.0000000	0.0000000	0.0000000	0.000000	0.0000000	0.0000000
ef_1	0.0000000	0.0000000	0.0000000	0.0000000	0.000000	0.0000000	0.0000000
tab2b	0.0000000	0.0000000	0.0000000	0.0000000	0.000000	0.0000000	0.0000000
taa2b	0.0000000	0.0000000	1.0000000	0.0000000	0.000000	0.0000000	0.0000000
tab1b	0.0000000	0.0000000	0.0000000	0.0000000	0.000000	0.0000000	0.0000000
taa1b	0.0000000	1.0000000	0.0000000	0.0000000	0.000000	0.0000000	0.0000000
tbrts	0.0000000	0.0000000	0.0000000	1.0000000	0.000000	0.0000000	0.0000000
trats	0.0000000	0.0000000	0.0000000	0.0000000	0.000000	0.0000000	1.0000000
PMP (Exact)	0.3924691	0.1471987	0.0969562	0.0537239	0.051309	0.0437093	0.0398128
PMP (MCMC)	0.3998900	0.1475400	0.0918200	0.0427400	0.052040	0.0474200	0.0387600

```
kable(topmodels.bma(reg0.bms)[,8:14])
```

	00020	00008	00800	01000	04000	00404	02000
ef_l	0.00000	0.0000000	0.0000000	0.0000000	0.000000	0.0000000	0.000000
ef_es	0.00000	0.0000000	0.0000000	0.0000000	0.000000	0.0000000	0.000000
ef_s	0.00000	0.0000000	0.0000000	0.0000000	1.000000	0.0000000	0.000000
tbrl	0.00000	0.0000000	0.0000000	0.0000000	0.000000	0.0000000	1.000000
tbres	0.00000	0.0000000	0.0000000	1.0000000	0.000000	0.0000000	0.000000
tbrs	0.00000	0.0000000	1.0000000	0.0000000	0.000000	0.0000000	0.000000
tral	0.00000	0.0000000	0.0000000	0.0000000	0.000000	1.0000000	0.000000
traes	0.00000	0.0000000	0.0000000	0.0000000	0.000000	0.0000000	0.000000
tras	0.00000	0.0000000	0.0000000	0.0000000	0.000000	0.0000000	0.000000
ef_2	0.00000	0.0000000	0.0000000	0.0000000	0.000000	0.0000000	0.000000
ef_1	0.00000	0.0000000	0.0000000	0.0000000	0.000000	0.0000000	0.000000
tab2b	1.00000	0.0000000	0.0000000	0.0000000	0.000000	0.0000000	0.000000
taa2b	0.00000	0.0000000	0.0000000	0.0000000	0.000000	0.0000000	0.000000
tab1b	0.00000	1.0000000	0.0000000	0.0000000	0.000000	0.0000000	0.000000
taa1b	0.00000	0.0000000	0.0000000	0.0000000	0.000000	1.0000000	0.000000
tbrts	0.00000	0.0000000	0.0000000	0.0000000	0.000000	0.0000000	0.000000
trats	0.00000	0.0000000	0.0000000	0.0000000	0.000000	0.0000000	0.000000
PMP (Exact)	0.03108	0.0287174	0.0253728	0.0221161	0.005183	0.0041378	0.003582
PMP (MCMC)	0.03087	0.0297500	0.0242500	0.0209500	0.003500	0.0032500	0.003200

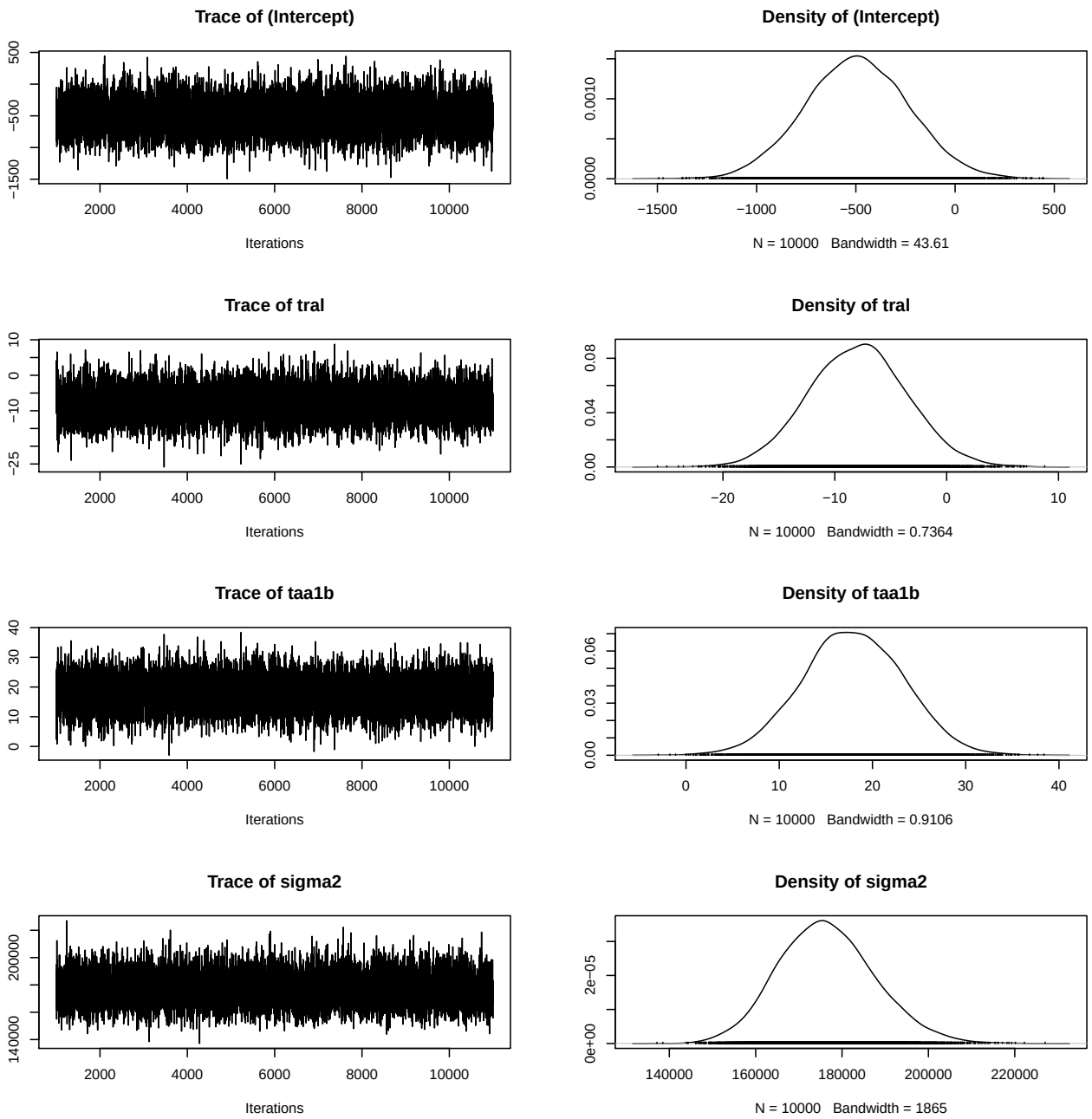
Le modèle le plus probable est celui qui ne contient aucune covariable, et ce, dans 39% des cas !
Les 12 premiers modèles proposés ne contiennent qu'une seule covariable! Difficile de choisir…
Sans conviction et bien qu'il n'apparaisse que dans 0.4% des cas, je choisis le 1er modèle qui contient plus d'une covariable.

```
reg1=MCMCregress(Barre~tral+taa1b , data=mut2)
summary(reg1)
```

```
##
## Iterations = 1001:11000
## Thinning interval = 1
## Number of chains = 1
## Sample size per chain = 10000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                    Mean        SD  Naive SE Time-series SE
## (Intercept)    -494.040   259.556   2.59556        2.60252
## tral             -7.933     4.383   0.04383        0.04383
## taa1b            17.881     5.420   0.05420        0.05420
## sigma2       176491.569 11117.790 111.17790      111.17790
##
## 2. Quantiles for each variable:
##
##                    2.5%       25%       50%        75%     97.5%
## (Intercept)    -997.051   -670.19   -495.14   -318.325 1.735e+01
## tral            -16.441    -10.92     -7.88     -5.018 7.354e-01
## taa1b             7.291     14.26     17.85     21.592 2.840e+01
## sigma2       155917.623 168739.50 176023.98 183617.262 1.998e+05
```

Examinons la trace et le graphique des densités marginales à postériori.

```
plot(reg1)
```



Il ne semble y avoir aucune structure particulière dans le graph. des traces. Ces dernières semblent "osciller convenablement", signe que l'algorithme fonctionne normalement en "allant chercher des points de la densité à postériori ni trop près, ni trop loin".

```
raftery.diag(reg1)
```

```
##
## Quantile (q) = 0.025
## Accuracy (r) = +/- 0.005
## Probability (s) = 0.95
```

```
##
##              Burn-in  Total Lower bound  Dependence
##              (M)      (N)   (Nmin)       factor (I)
##  (Intercept) 2        3741  3746         0.999
##  tral        2        3802  3746         1.010
##  taa1b       2        3771  3746         1.010
##  sigma2      2        3834  3746         1.020
```

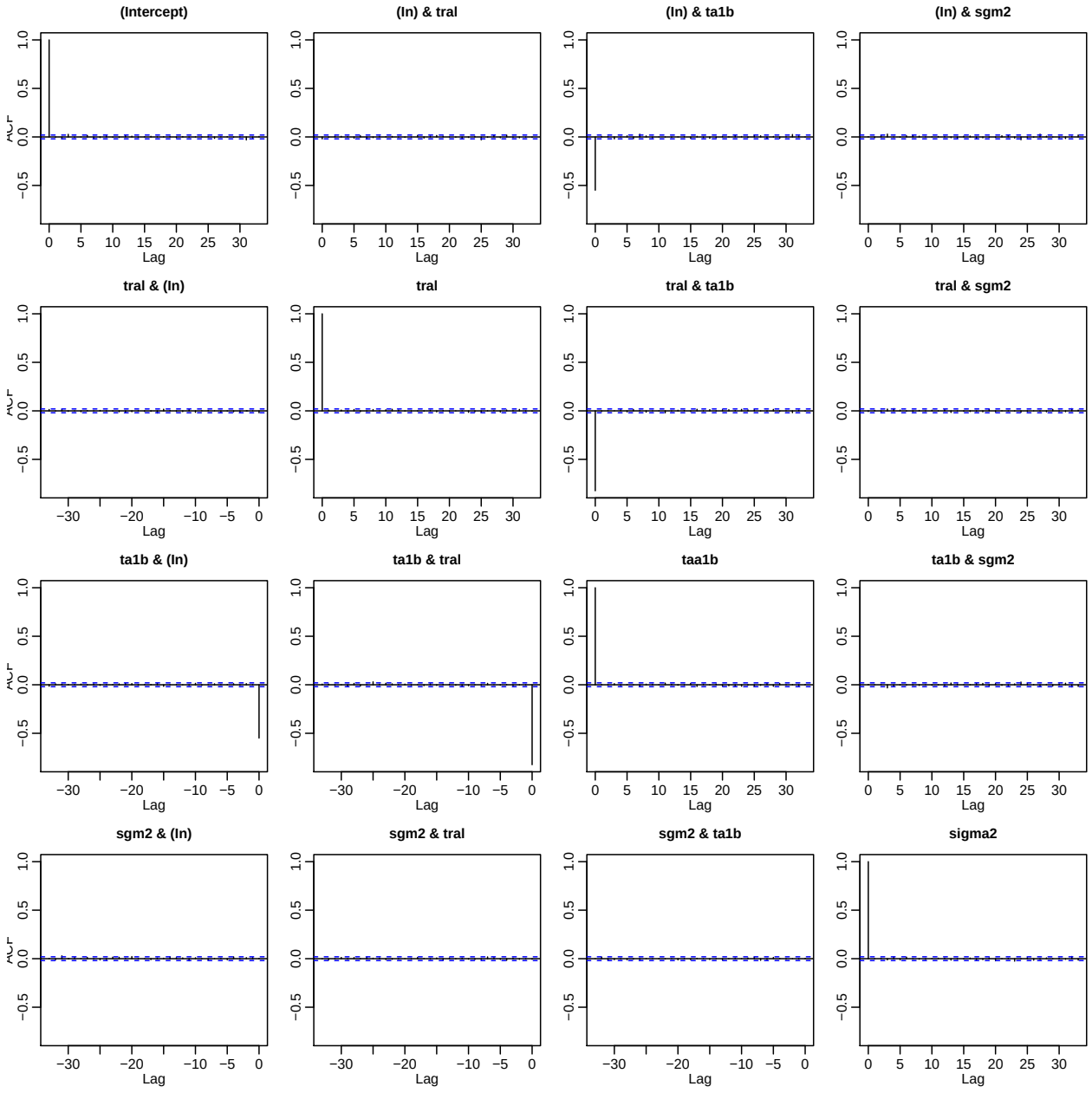
C'est quasi-pafait ! Il aurait fallu entre 3740 et 3850 itérations pour faire fonctionner convenablement l'algorithlme de MCMC, ce qui est bien inférieur au 10 000 générées précédemment.
Il n' y a quasiment aucun "temps de chauffe" (Burn-in = 2), Il ne me semble pas nécessaire de modifier la point de départ, la taille des pas ("tune"), et donc le Burn-IN pour l'instant.

Obeservons l'autocorrélation potentielle afin de vérifier si la chaine de markov "oublie rapidement son passé".

```
effectiveSize(reg1)
```

```
## (Intercept)        tral       taa1b      sigma2
##    9946.623   10000.000   10000.000   10000.000
```

```
acf(reg1)
```



En résumé, l'algortihme MCMC converge, la sélection BMS fonctionne, mais la qualité du modèle proposé est pauvre.
Je décide de ne pas poursuivre plus en détail l'étude de ce modèle.

2. Analyse fréquentiste/baysienne

```
reg_freq=lm(Barre~., data=mut2)
summary(reg_freq)
```

```
##
## Call:
## lm(formula = Barre ~ ., data = mut2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -429.72 -205.90 -122.25   -8.55 1645.96
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -4.725e+02  5.586e+02  -0.846   0.3980
## ef_l         7.781e-01  1.638e+00   0.475   0.6351
## ef_es        2.924e-01  1.234e+00   0.237   0.8128
## ef_s         9.694e-03  1.019e+00   0.010   0.9924
## tbrl         3.122e+00  2.559e+00   1.220   0.2232
## tbres        4.811e+00  4.205e+00   1.144   0.2531
## tbrs         9.385e+00  6.383e+00   1.470   0.1421
## tral        -1.428e+01  6.879e+00  -2.077   0.0383 *
## traes        3.814e+00  8.261e+00   0.462   0.6445
## tras        -4.299e+00  9.586e+00  -0.448   0.6540
## ef_2         4.306e-02  6.229e-01   0.069   0.9449
## ef_1        -3.521e-01  7.182e-01  -0.490   0.6242
## tab2b        1.074e+01  5.655e+00   1.900   0.0580 .
## taa2b       -7.077e+00  9.038e+00  -0.783   0.4340
## tab1b       -2.039e+01  1.071e+01  -1.904   0.0575 .
## taa1b        3.444e+01  1.916e+01   1.797   0.0729 .
## tbrts       -5.392e+00  1.288e+01  -0.419   0.6757
## trats       -4.072e+00  2.202e+01  -0.185   0.8534
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 422.4 on 498 degrees of freedom
## Multiple R-squared:  0.04068,    Adjusted R-squared:  0.007931
## F-statistic: 1.242 on 17 and 498 DF,  p-value: 0.2267
```

En l'état, il n'y a pas grand chose de significatif et le "Adjusted R-squared" ($R^2_{adj} = 0.007931$) est tout simplement catastrophique…

Etudions d'éventuelles corrélations dans les variable susceptible de minimiser la qualité de l'estimation de $\hat{\beta}$ due à un mauvais coditionnement de la matrice à inverser.

J'utilise la fonction *symnum* (le package "corrplot" semble en travaux aujourd'hui…).

```
C2 = cor(mut2)
symnum(C2,symbols = c(" ", ".", "*", "**", "***", "****"),abbr.colnames = F)
```

```
##       Barre ef_l ef_es ef_s tbrl tbres tbrs tral traes tras ef_2 ef_1 tab2b
## Barre 1
## ef_l        1
## ef_es       *    1
## ef_s        *    **    1
## tbrl                      1
## tbres                .    .    1
## tbrs                 .    .    .     1
## tral            .    .    .    .     *    1
## traes           .    .    .    *     *    **   1
## tras            .    .    .    *     **   **   ***   1
## ef_2        *    **    **                   .    .          1
## ef_1        *    **    **                   .    .     .    **** 1
## tab2b           .    .    .    *     .    *    *     *    .    .    1
## taa2b           .    *    .    .     *    *    **    **   .    .    **
## tab1b           .    .    .    *     *    *    **    **   .         **
## taa1b           .    *    .    *     *    **   ***   ***  .    .    *
## tbrts           .    .    .    *     **   *    **    **             *
## trats           .    .    .    *     **   **   ***   **** .    .    *
##       taa2b tab1b taa1b tbrts trats
## Barre
## ef_l
## ef_es
## ef_s
## tbrl
## tbres
## tbrs
## tral
## traes
## tras
## ef_2
## ef_1
## tab2b
## taa2b 1
## tab1b *     1
## taa1b ***   **    1
## tbrts *     ***   **    1
## trats **    **    ****  **    1
## attr(,"legend")
## [1] 0 ' ' 0.3 '.' 0.6 '*' 0.8 '**' 0.9 '***' 0.95 '****' 1
```

Ici, on voit clairement des groupes de variables corrélées, voire fortement corrélées. Cela a pour conséquence d'induire du bruit. Je tente de sélectionner une covariable pour chaque groupe de variables "assez corrélées" et faiblement corrélé avec les autres.

```
reg_freq2=lm(Barre~ ef_s+ef_2+ taa1b, data=mut2)
summary(reg_freq2)
```

```
##
## Call:
## lm(formula = Barre ~ ef_s + ef_2 + taa1b, data = mut2)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -373.97 -199.10 -129.07  -25.07 1688.23
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -500.84895  354.83289  -1.412   0.1587
## ef_s           0.05336    0.76592   0.070   0.9445
## ef_2          -0.05808    0.27321  -0.213   0.8317
## taa1b          9.94851    4.33300   2.296   0.0221 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 421.2 on 512 degrees of freedom
## Multiple R-squared:  0.01935,    Adjusted R-squared:  0.0136
## F-statistic: 3.367 on 3 and 512 DF,  p-value: 0.01847
```

```
AIC(reg_freq2)
```

```
## [1] 7706.719
```

Il n'y a rien de probant. Voyons ce que fournit une sélection automatique.

```
modselect_f=stepAIC(reg_freq,~., data=mut2,trace=F,direction=c("both"))
summary(modselect_f)
```

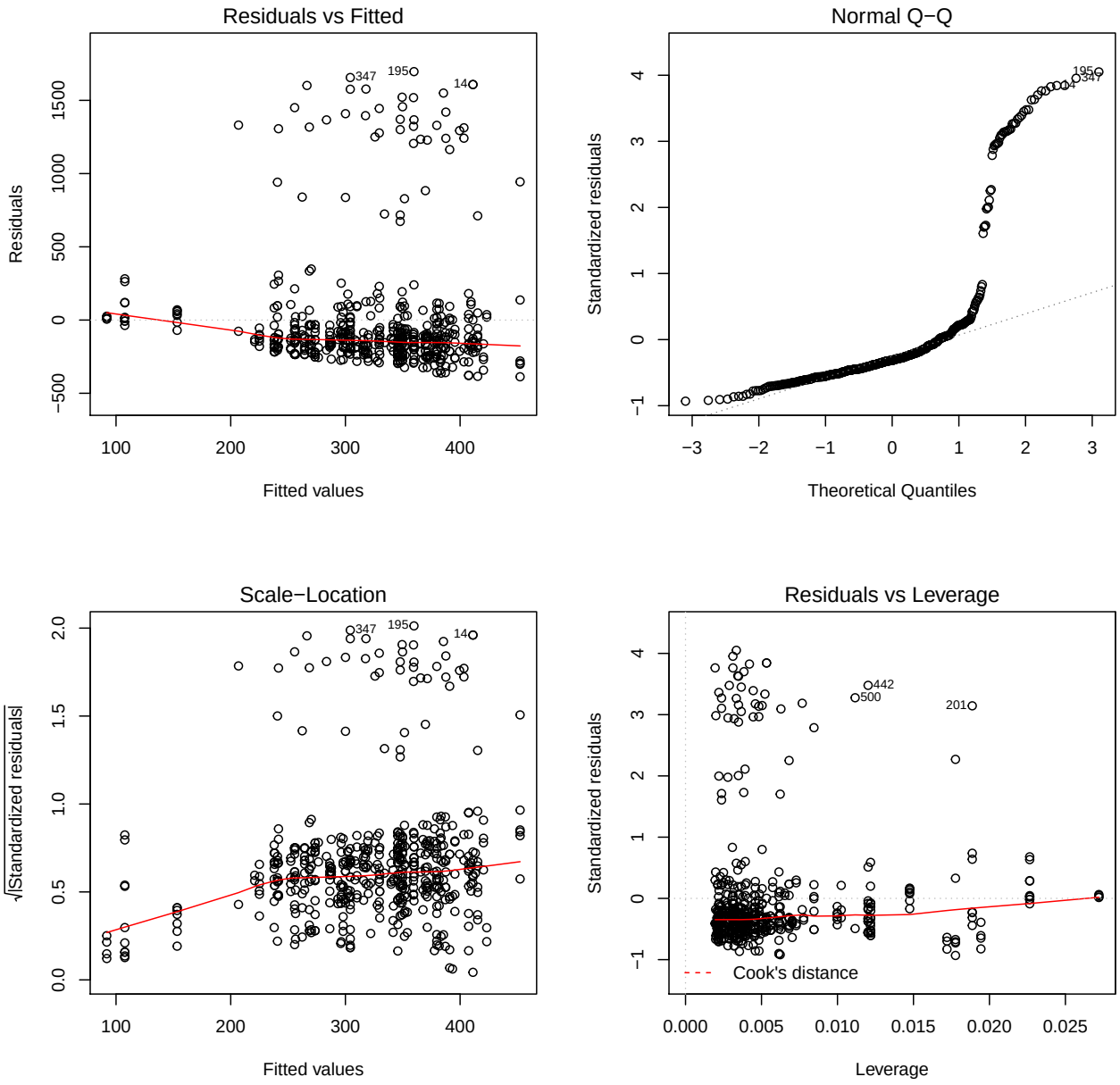
```
##
## Call:
## lm(formula = Barre ~ tral + taa1b, data = mut2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -387.32 -196.56 -130.83  -14.95 1696.20
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -494.324    260.593  -1.897  0.05840 .
## tral          -7.882      4.360  -1.808  0.07124 .
## taa1b         17.833      5.407   3.298  0.00104 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 419.5 on 513 degrees of freedom
## Multiple R-squared:  0.02539,    Adjusted R-squared:  0.02159
## F-statistic: 6.681 on 2 and 513 DF,  p-value: 0.001366
```

```
AIC(modselect_f)
```

```
## [1] 7701.531
```

Il reste peu de covariables. Cela risque de ne pas être concluant.

```
par(mfrow=c(2,2))
plot(modselect_f)
```



C'est franchement mauvais :

- les résidus semblent structurés et assez mal disséminés autour des régressions locales (graph.1 et 3) qui, de fait, ne sont pas "horizontales"
- le QQ-plot traduit une "non-gaussianité" manifeste
- enfin, ce modèles induit un nombres conséquents de "points atypiques" avec des distances de Cook élevées.

Bref, le modèle gaussien semble totalement inadapté ici.

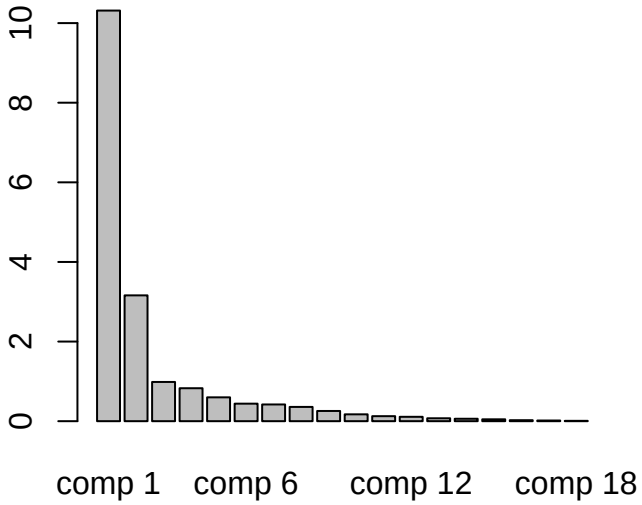
Essai d'une ACP

```
par(mfrow=c(1,2))
# réalisation de l'ACP
library('FactoMineR')
mut2.sc=scale(mut2)
res.pca = PCA(mut2.sc,graph=FALSE)
```

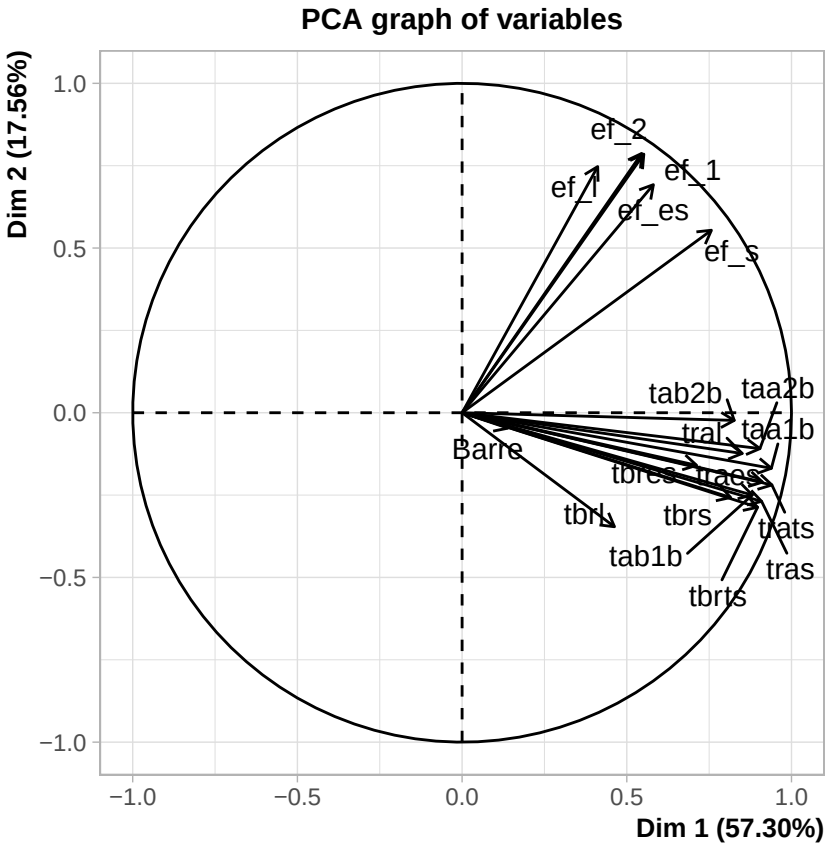
```
#valeurs propres et composantes
kable(res.pca$eig)
```

	eigenvalue	percentage of variance	cumulative percentage of variance
comp 1	10.3145940	57.3032998	57.30330
comp 2	3.1603543	17.5575237	74.86082
comp 3	0.9851796	5.4732198	80.33404
comp 4	0.8277788	4.5987709	84.93281
comp 5	0.5990090	3.3278276	88.26064
comp 6	0.4398799	2.4437774	90.70442
comp 7	0.4203393	2.3352184	93.03964
comp 8	0.3584607	1.9914484	95.03109
comp 9	0.2562714	1.4237300	96.45482
comp 10	0.1720688	0.9559375	97.41075
comp 11	0.1259999	0.6999995	98.11075
comp 12	0.1097055	0.6094749	98.72023
comp 13	0.0728137	0.4045205	99.12475
comp 14	0.0612767	0.3404262	99.46517
comp 15	0.0467095	0.2594974	99.72467
comp 16	0.0247059	0.1372548	99.86193
comp 17	0.0164592	0.0914400	99.95337
comp 18	0.0083939	0.0466329	100.00000

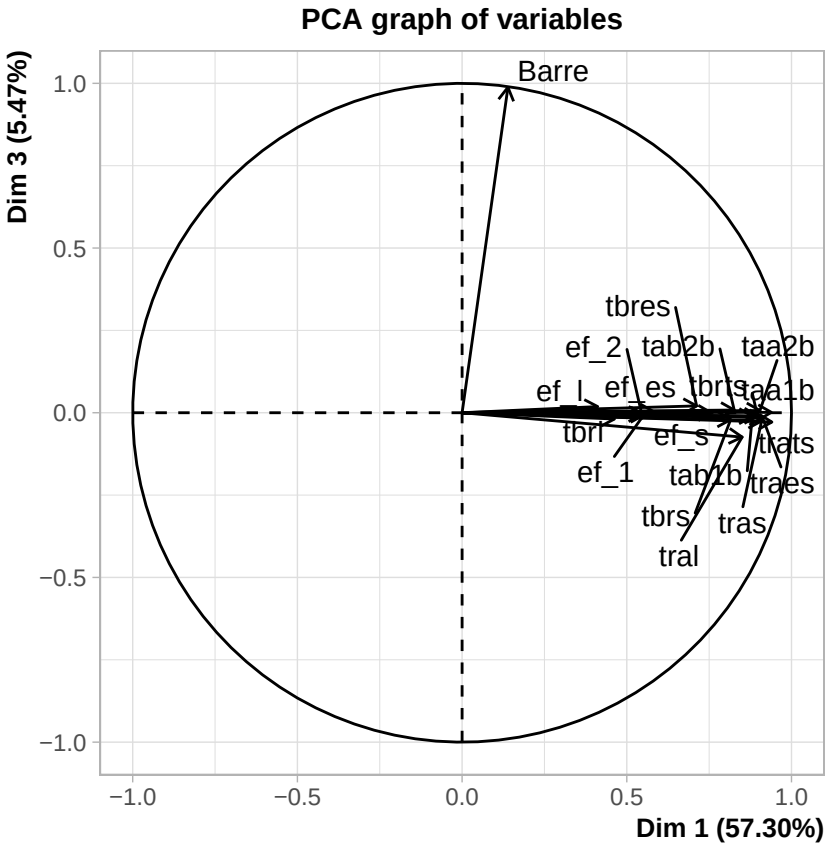
```
barplot(res.pca$eig[,1])
```



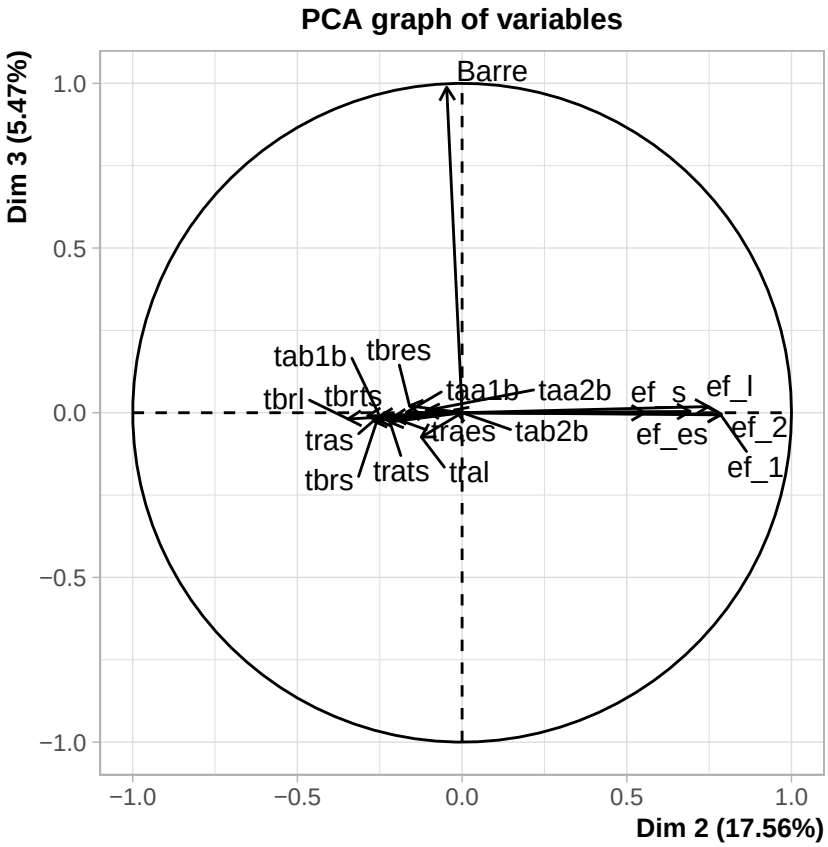

```
par(mfrow=c(1,3))
plot(res.pca,choix="varcor",axes=c(1,2))
```




```
plot(res.pca,choix="varcor",axes=c(1,3))
```




```
plot(res.pca,choix="varcor",axes=c(2,3))
```



Il y a des groupement de variables intéressantes.
Les 3 premiers axes concentrent 80% de la variance totale. Ils sont assez représentatifs pour effectuer une étude.
Je pourrai tenter de grouper les variables (selon ces 3 axes), expliquer les axes, regarder ce qu'ils opposent (selon les varaiables et les "individus extrêmes"), puis effectuer une régression PLS (par exemple), mais je vais m'en passer dans ce projet.

3. Maths et Anglais

Reprenons l'étude précédente en ciblant sur les 2 disciplines.

```
mut_mat=mut[(mut$Matiere=="MATHS"),]
mut_ang=mut[(mut$Matiere=="ANGLAIS"),]
# restrictions aux varaiables quantitatives
mut_mat2=mut_mat[,6:23]
mut_ang2=mut_ang[,6:23]
```

3.1 Maths

```
reg_mat2.0=lm(mut_mat2$Barre~., data=mut_mat2)
summary(reg_mat2.0)
```

regression linéaire

```
##
## Call:
## lm(formula = mut_mat2$Barre ~ ., data = mut_mat2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -173.08  -69.00  -21.11   75.56  273.03
##
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                        65.7874   489.3234   0.134   0.8937
## effectif_presents_serie_l           0.4061     1.2463   0.326   0.7462
## effectif_presents_serie_es         -0.3858     0.9616  -0.401   0.6903
## effectif_presents_serie_s           1.1721     0.8454   1.386   0.1731
## taux_brut_de_reussite_serie_l      -3.6683     2.1240  -1.727   0.0917 .
## taux_brut_de_reussite_serie_es      7.2467     4.1641   1.740   0.0893 .
## taux_brut_de_reussite_serie_s       1.4955     5.5932   0.267   0.7905
## taux_reussite_attendu_serie_l      -4.1272     5.6497  -0.731   0.4692
## taux_reussite_attendu_serie_es     -6.2008     7.1700  -0.865   0.3922
## taux_reussite_attendu_serie_s      -2.0550     8.3611  -0.246   0.8071
## effectif_de_seconde                 0.8035     0.5236   1.535   0.1325
## effectif_de_premiere               -1.4042     0.6320  -2.222   0.0319 *
## taux_acces_brut_seconde_bac        13.0976     6.2043   2.111   0.0409 *
## taux_acces_attendu_seconde_bac     -9.2312     7.2998  -1.265   0.2132
## taux_acces_brut_premiere_bac      -12.5976    10.4840  -1.202   0.2364
## taux_acces_attendu_premiere_bac     5.6413    14.9814   0.377   0.7084
## taux_brut_de_reussite_total_series  1.4551    11.0152   0.132   0.8956
## taux_reussite_attendu_total_series 11.3206    17.7198   0.639   0.5265
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 116.4 on 41 degrees of freedom
## Multiple R-squared:  0.3826, Adjusted R-squared:  0.1267
## F-statistic: 1.495 on 17 and 41 DF,  p-value: 0.1451
```

```r
# sélection automatique
reg_mat2.1=stepAIC(reg_mat2.0,~., data=mut_mat2,trace=F,direction=c("backward"))
summary(reg_mat2.1)
```

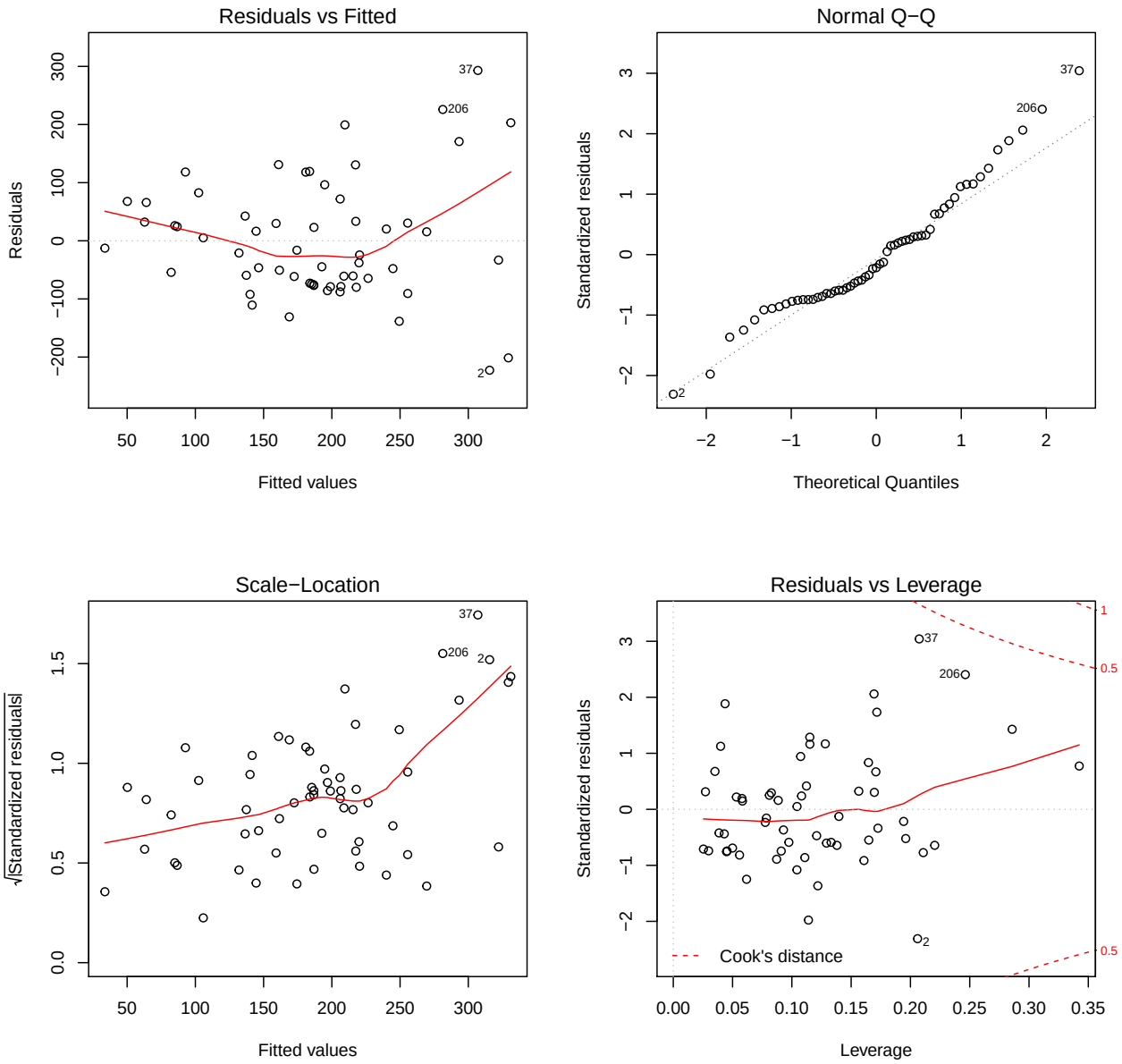
```
##
## Call:
## lm(formula = mut_mat2$Barre ~ effectif_presents_serie_s + taux_brut_de_reussite_serie_l +
##     taux_brut_de_reussite_serie_es + effectif_de_seconde + effectif_de_premiere +
##     taux_acces_brut_seconde_bac, data = mut_mat2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -222.62  -73.59  -20.98   54.23  292.99
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   -77.4081   182.6176  -0.424   0.6734
## effectif_presents_serie_s       0.7892     0.5009   1.575   0.1212
## taux_brut_de_reussite_serie_l  -3.9693     1.5624  -2.541   0.0141 *
## taux_brut_de_reussite_serie_es  3.8648     2.2119   1.747   0.0865 .
## effectif_de_seconde             0.6536     0.3883   1.683   0.0983 .
## effectif_de_premiere           -1.1044     0.4286  -2.576   0.0129 *
## taux_acces_brut_seconde_bac     4.2086     2.6145   1.610   0.1135
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 108.2 on 52 degrees of freedom
## Multiple R-squared:  0.3242, Adjusted R-squared:  0.2462
## F-statistic: 4.157 on 6 and 52 DF,  p-value: 0.001744
```

```r
AIC(reg_mat2.1)
```

```
## [1] 728.6768
```

Sans être extraordinaire, le R^2_{adj} et AIC s'améliorent un peu en sélectionnant quelques covariables.

```r
par(mfrow=c(2,2))
plot(reg_mat2.1)
```



C'est un peu mieux que précédemment. Le QQ-plot demeure non probant. les résidus sont plutôt bien répartis ,mais les régressions locales "se tordent un peu" et semblent s'éloigner de la direction horizontale idéale.
Enfin, les distances de Cook semblent acceptables sans être extraordinaires.

Algorithme MCMC On simule 10 000 itérations de la loi à postériori par la méthode de Monte-Carlo associées aux chaîne de Markov (MCMC).

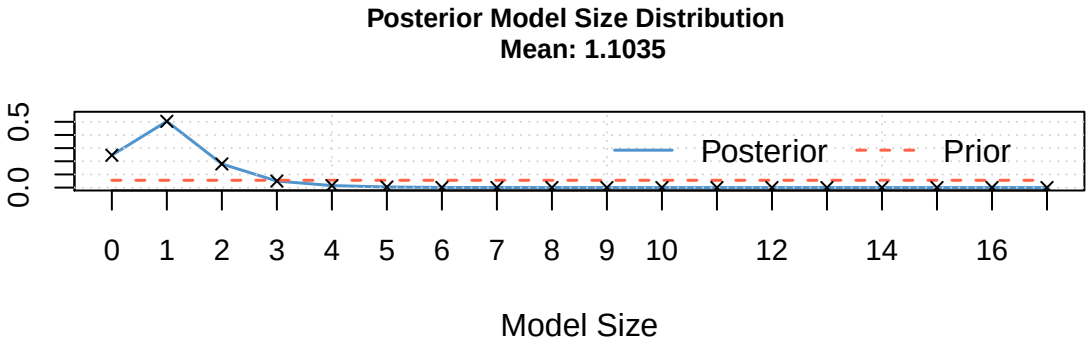
```
reg0.mat=MCMCregress(Barre~., data=mut_mat2)
summary(reg0.mat)
```

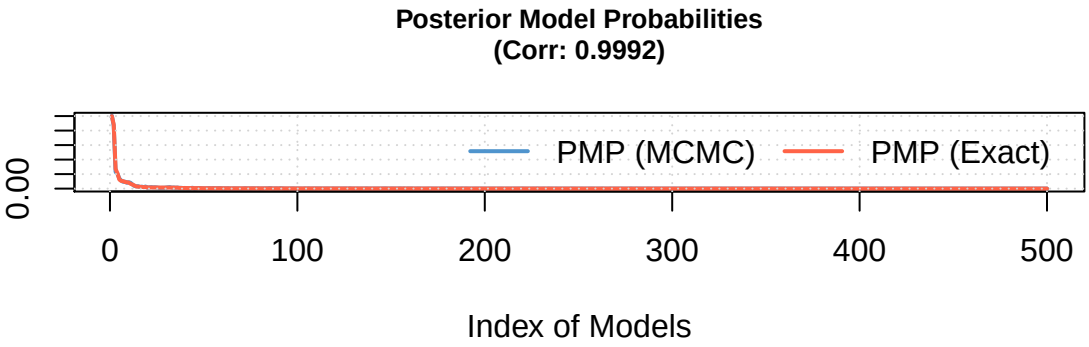
```
##
## Iterations = 1001:11000
## Thinning interval = 1
## Number of chains = 1
## Sample size per chain = 10000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                                      Mean        SD  Naive SE
## (Intercept)                       67.4523  500.0076  5.000076
## effectif_presents_serie_l          0.4013    1.2837  0.012837
## effectif_presents_serie_es        -0.3867    0.9701  0.009701
## effectif_presents_serie_s          1.1703    0.8663  0.008663
## taux_brut_de_reussite_serie_l     -3.6940    2.1484  0.021484
## taux_brut_de_reussite_serie_es     7.2346    4.2445  0.042445
## taux_brut_de_reussite_serie_s      1.4298    5.6936  0.056936
## taux_reussite_attendu_serie_l     -4.1673    5.8405  0.058405
## taux_reussite_attendu_serie_es    -6.1565    7.3670  0.073670
## taux_reussite_attendu_serie_s     -2.0245    8.5632  0.085632
## effectif_de_seconde                0.8110    0.5426  0.005426
## effectif_de_premiere              -1.4121    0.6550  0.006550
## taux_acces_brut_seconde_bac       13.2278    6.3089  0.063089
## taux_acces_attendu_seconde_bac    -9.2566    7.4632  0.074632
## taux_acces_brut_premiere_bac     -12.8203   10.6022  0.106022
## taux_acces_attendu_premiere_bac    5.8948   15.2374  0.152374
## taux_brut_de_reussite_total_series 1.5784   11.1902  0.111902
## taux_reussite_attendu_total_series 11.1336  18.1748  0.181748
## sigma2                         14280.1921 3322.8942 33.228942
##                                Time-series SE
## (Intercept)                          5.043451
## effectif_presents_serie_l            0.012837
## effectif_presents_serie_es           0.009701
## effectif_presents_serie_s            0.008408
## taux_brut_de_reussite_serie_l        0.021048
## taux_brut_de_reussite_serie_es       0.042126
## taux_brut_de_reussite_serie_s        0.056936
## taux_reussite_attendu_serie_l        0.058405
## taux_reussite_attendu_serie_es       0.072475
## taux_reussite_attendu_serie_s        0.083186
## effectif_de_seconde                  0.005426
## effectif_de_premiere                 0.006700
## taux_acces_brut_seconde_bac          0.063089
## taux_acces_attendu_seconde_bac       0.074632
## taux_acces_brut_premiere_bac         0.107550
## taux_acces_attendu_premiere_bac      0.155409
## taux_brut_de_reussite_total_series   0.111902
## taux_reussite_attendu_total_series   0.181748
```

```
## sigma2                                  48.369544
##
## 2. Quantiles for each variable:
##
##                                      2.5%       25%       50%       75%
## (Intercept)                      -914.3191  -269.3885   66.4187   397.3215
## effectif_presents_serie_l          -2.1486    -0.4416    0.4041     1.2642
## effectif_presents_serie_es         -2.3009    -1.0322   -0.3905     0.2578
## effectif_presents_serie_s          -0.5231     0.5877    1.1595     1.7513
## taux_brut_de_reussite_serie_l      -7.9048    -5.1343   -3.6841    -2.2533
## taux_brut_de_reussite_serie_es     -1.1000     4.3966    7.2001    10.0823
## taux_brut_de_reussite_serie_s      -9.7921    -2.3034    1.3743     5.1666
## taux_reussite_attendu_serie_l     -15.6817    -7.9943   -4.2055    -0.2629
## taux_reussite_attendu_serie_es    -20.6382   -11.0327   -6.1085    -1.2763
## taux_reussite_attendu_serie_s     -18.7786    -7.7847   -2.1505     3.6123
## effectif_de_seconde                -0.2657     0.4638    0.8155     1.1645
## effectif_de_premiere               -2.7096    -1.8413   -1.4103    -0.9760
## taux_acces_brut_seconde_bac         0.8995     9.0584   13.2321    17.4312
## taux_acces_attendu_seconde_bac    -24.0529   -14.2979   -9.2388    -4.3279
## taux_acces_brut_premiere_bac      -33.5284   -19.9334  -13.0226    -5.7717
## taux_acces_attendu_premiere_bac   -24.1264    -4.4210    6.0930    16.1689
## taux_brut_de_reussite_total_series -20.3869   -5.9408    1.7265     9.0156
## taux_reussite_attendu_total_series -24.6605   -0.8632   11.2726    23.3063
## sigma2                           9173.5050 11937.6471 13806.5158 16141.6306
##                                     97.5%
## (Intercept)                      1063.0583
## effectif_presents_serie_l           2.8670
## effectif_presents_serie_es          1.5367
## effectif_presents_serie_s           2.9052
## taux_brut_de_reussite_serie_l       0.4996
## taux_brut_de_reussite_serie_es     15.5475
## taux_brut_de_reussite_serie_s      12.7086
## taux_reussite_attendu_serie_l       7.3722
## taux_reussite_attendu_serie_es      8.4961
## taux_reussite_attendu_serie_s      15.0643
## effectif_de_seconde                 1.8809
## effectif_de_premiere               -0.1325
## taux_acces_brut_seconde_bac        25.6278
## taux_acces_attendu_seconde_bac      5.2425
## taux_acces_brut_premiere_bac        8.1238
## taux_acces_attendu_premiere_bac    35.8679
## taux_brut_de_reussite_total_series 23.1985
## taux_reussite_attendu_total_series 46.6924
## sigma2                           21923.2032
```

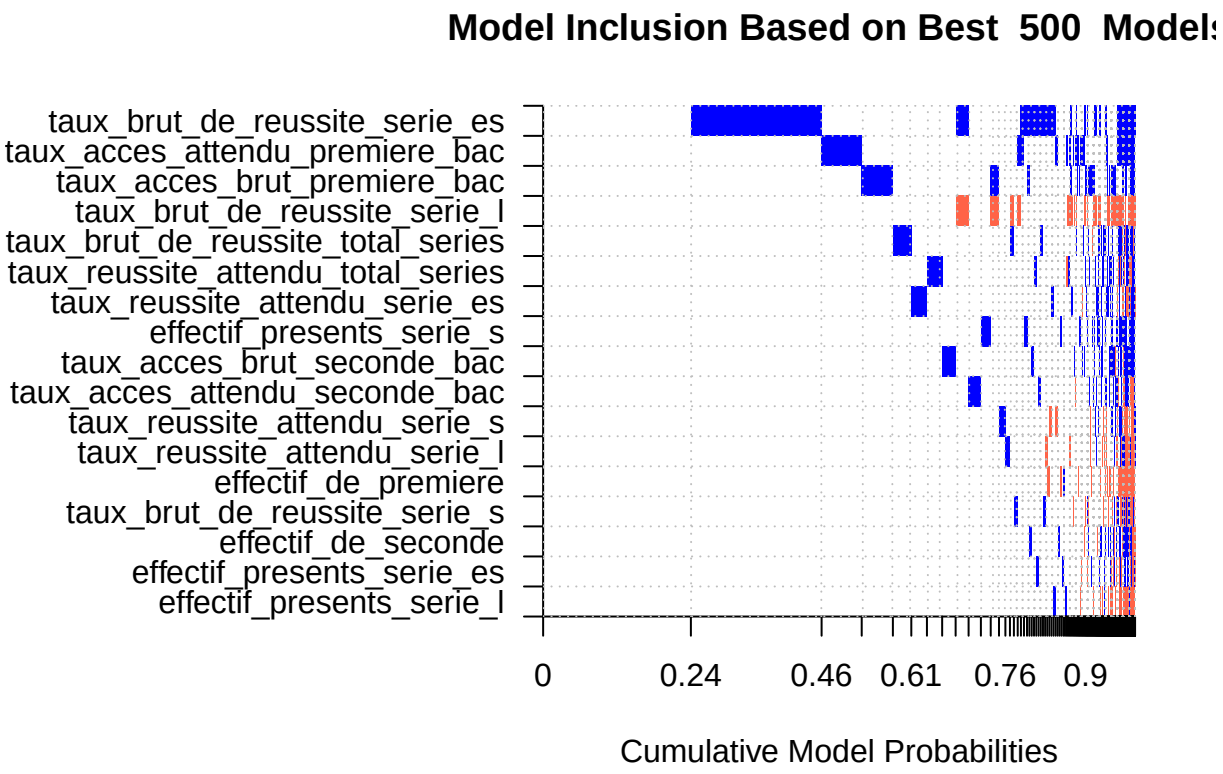
```
reg0.mat.bms=bms(Barre~., data=mut_mat2, burn = 1e4, iter=1e5)
```

```
##                                      PIP     Post Mean    Post SD
## taux_brut_de_reussite_serie_es     0.33004  1.7562436500 2.73361865
## taux_acces_attendu_premiere_bac    0.10595  0.7826252246 2.78454737
## taux_acces_brut_premiere_bac       0.09607  0.5882417810 2.10341217
## taux_brut_de_reussite_serie_l      0.09368 -0.3021452352 1.08021371
## taux_brut_de_reussite_total_series 0.06587  0.3009597352 1.45000800
## taux_reussite_attendu_serie_es     0.05421  0.1670727574 1.07388046
## effectif_presents_serie_s          0.05399  0.0277964006 0.15967229
## taux_acces_brut_seconde_bac        0.04875  0.1425363098 0.87799884
## taux_reussite_attendu_total_series 0.04358  0.1201228721 1.24634944
## taux_acces_attendu_seconde_bac     0.04233  0.1020799232 0.97101753
## taux_reussite_attendu_serie_s      0.02871 -0.0056482039 0.75996091
## taux_reussite_attendu_serie_l      0.02563  0.0127911315 0.64911668
## effectif_de_seconde                0.02481  0.0024402604 0.04955020
## effectif_de_premiere               0.02467 -0.0055809611 0.06682025
## effectif_presents_serie_es         0.02455 -0.0001038855 0.10297773
## taux_brut_de_reussite_serie_s      0.02099  0.0002434097 0.47416449
## effectif_presents_serie_l          0.01972 -0.0025186851 0.11764406
##                                    Cond.Pos.Sign Idx
## taux_brut_de_reussite_serie_es         1.0000000   5
## taux_acces_attendu_premiere_bac        1.0000000  15
## taux_acces_brut_premiere_bac           0.9964609  14
## taux_brut_de_reussite_serie_l          0.0000000   4
## taux_brut_de_reussite_total_series     0.9725216  16
## taux_reussite_attendu_serie_es         0.9173584   8
## effectif_presents_serie_s              0.9987035   3
## taux_acces_brut_seconde_bac            0.9698461  12
## taux_reussite_attendu_total_series     0.8854979  17
## taux_acces_attendu_seconde_bac         0.8634538  13
## taux_reussite_attendu_serie_s          0.4754441   9
## taux_reussite_attendu_serie_l          0.4728833   7
## effectif_de_seconde                    0.7404272  10
## effectif_de_premiere                   0.1333604  11
## effectif_presents_serie_es             0.6729124   2
## taux_brut_de_reussite_serie_s          0.5583611   6
## effectif_presents_serie_l              0.4751521   1
##
## Mean no. regressors              Draws             Burnins                Time
##            "1.1036"             "1e+05"             "10000"      "3.75146 secs"
##  No. models visited      Modelspace 2^K          % visited         % Topmodels
##             "13709"            "131072"               "10"               "98"
##           Corr PMP           No. Obs.         Model Prior             g-Prior
##            "0.9992"               "59"      "random / 8.5"               "UIP"
##     Shrinkage-Stats
##         "Av=0.9833"
##
## Time difference of 3.75146 secs
```






```
image(reg0.mat.bms)
```




```
topmodels.bma(reg0.mat.bms)[,1:8]
```

```
##                                 00000      01000      00004      00008
```

```
## effectif_presents_serie_l          0.00000 0.0000000 0.00000000 0.00000000
## effectif_presents_serie_es         0.00000 0.0000000 0.00000000 0.00000000
## effectif_presents_serie_s          0.00000 0.0000000 0.00000000 0.00000000
## taux_brut_de_reussite_serie_l      0.00000 0.0000000 0.00000000 0.00000000
## taux_brut_de_reussite_serie_es     0.00000 1.0000000 0.00000000 0.00000000
## taux_brut_de_reussite_serie_s      0.00000 0.0000000 0.00000000 0.00000000
## taux_reussite_attendu_serie_l      0.00000 0.0000000 0.00000000 0.00000000
## taux_reussite_attendu_serie_es     0.00000 0.0000000 0.00000000 0.00000000
## taux_reussite_attendu_serie_s      0.00000 0.0000000 0.00000000 0.00000000
## effectif_de_seconde                0.00000 0.0000000 0.00000000 0.00000000
## effectif_de_premiere               0.00000 0.0000000 0.00000000 0.00000000
## taux_acces_brut_seconde_bac        0.00000 0.0000000 0.00000000 0.00000000
## taux_acces_attendu_seconde_bac     0.00000 0.0000000 0.00000000 0.00000000
## taux_acces_brut_premiere_bac       0.00000 0.0000000 0.00000000 1.00000000
## taux_acces_attendu_premiere_bac    0.00000 0.0000000 1.00000000 0.00000000
## taux_brut_de_reussite_total_series 0.00000 0.0000000 0.00000000 0.00000000
## taux_reussite_attendu_total_series 0.00000 0.0000000 0.00000000 0.00000000
## PMP (Exact)                        0.24433 0.2161623 0.06632039 0.05157653
## PMP (MCMC)                         0.24592 0.2122300 0.05610000 0.05012000
##                                        00002      00200      00001      00020
## effectif_presents_serie_l          0.00000000 0.00000000 0.00000000 0.00000000
## effectif_presents_serie_es         0.00000000 0.00000000 0.00000000 0.00000000
## effectif_presents_serie_s          0.00000000 0.00000000 0.00000000 0.00000000
## taux_brut_de_reussite_serie_l      0.00000000 0.00000000 0.00000000 0.00000000
## taux_brut_de_reussite_serie_es     0.00000000 0.00000000 0.00000000 0.00000000
## taux_brut_de_reussite_serie_s      0.00000000 0.00000000 0.00000000 0.00000000
## taux_reussite_attendu_serie_l      0.00000000 0.00000000 0.00000000 0.00000000
## taux_reussite_attendu_serie_es     0.00000000 1.00000000 0.00000000 0.00000000
## taux_reussite_attendu_serie_s      0.00000000 0.00000000 0.00000000 0.00000000
## effectif_de_seconde                0.00000000 0.00000000 0.00000000 0.00000000
## effectif_de_premiere               0.00000000 0.00000000 0.00000000 0.00000000
## taux_acces_brut_seconde_bac        0.00000000 0.00000000 0.00000000 1.00000000
## taux_acces_attendu_seconde_bac     0.00000000 0.00000000 0.00000000 0.00000000
## taux_acces_brut_premiere_bac       0.00000000 0.00000000 0.00000000 0.00000000
## taux_acces_attendu_premiere_bac    0.00000000 0.00000000 0.00000000 0.00000000
## taux_brut_de_reussite_total_series 1.00000000 0.00000000 0.00000000 0.00000000
## taux_reussite_attendu_total_series 0.00000000 0.00000000 1.00000000 0.00000000
## PMP (Exact)                        0.03045483 0.02583959 0.02518429 0.02224611
## PMP (MCMC)                         0.03401000 0.02737000 0.02281000 0.02375000
```

```
topmodels.bma(reg0.mat.bms)[,9:16]
```

```
##                                        03000      00010      04000      02008
## effectif_presents_serie_l          0.00000000 0.00000000 0.00000000 0.00000000
## effectif_presents_serie_es         0.00000000 0.00000000 0.00000000 0.00000000
## effectif_presents_serie_s          0.00000000 0.00000000 1.00000000 0.00000000
## taux_brut_de_reussite_serie_l      1.00000000 0.00000000 0.00000000 1.00000000
## taux_brut_de_reussite_serie_es     1.00000000 0.00000000 0.00000000 0.00000000
## taux_brut_de_reussite_serie_s      0.00000000 0.00000000 0.00000000 0.00000000
## taux_reussite_attendu_serie_l      0.00000000 0.00000000 0.00000000 0.00000000
## taux_reussite_attendu_serie_es     0.00000000 0.00000000 0.00000000 0.00000000
## taux_reussite_attendu_serie_s      0.00000000 0.00000000 0.00000000 0.00000000
## effectif_de_seconde                0.00000000 0.00000000 0.00000000 0.00000000
## effectif_de_premiere               0.00000000 0.00000000 0.00000000 0.00000000
## taux_acces_brut_seconde_bac        0.00000000 0.00000000 0.00000000 0.00000000
```

```
## taux_acces_attendu_seconde_bac     0.00000000 1.00000000 0.00000000 0.00000000
## taux_acces_brut_premiere_bac       0.00000000 0.00000000 0.00000000 1.00000000
## taux_acces_attendu_premiere_bac    0.00000000 0.00000000 0.00000000 0.00000000
## taux_brut_de_reussite_total_series 0.00000000 0.00000000 0.00000000 0.00000000
## taux_reussite_attendu_total_series 0.00000000 0.00000000 0.00000000 0.00000000
## PMP (Exact)                        0.02146376 0.02029974 0.01630646 0.01364421
## PMP (MCMC)                         0.01950000 0.02181000 0.01945000 0.01380000
##                                         00100       00400       02002
## effectif_presents_serie_l          0.00000000 0.000000000 0.000000000
## effectif_presents_serie_es         0.00000000 0.000000000 0.000000000
## effectif_presents_serie_s          0.00000000 0.000000000 0.000000000
## taux_brut_de_reussite_serie_l      0.00000000 0.000000000 1.000000000
## taux_brut_de_reussite_serie_es     0.00000000 0.000000000 0.000000000
## taux_brut_de_reussite_serie_s      0.00000000 0.000000000 0.000000000
## taux_reussite_attendu_serie_l      0.00000000 1.000000000 0.000000000
## taux_reussite_attendu_serie_es     0.00000000 0.000000000 0.000000000
## taux_reussite_attendu_serie_s      1.00000000 0.000000000 0.000000000
## effectif_de_seconde                0.00000000 0.000000000 0.000000000
## effectif_de_premiere               0.00000000 0.000000000 0.000000000
## taux_acces_brut_seconde_bac        0.00000000 0.000000000 0.000000000
## taux_acces_attendu_seconde_bac     0.00000000 0.000000000 0.000000000
## taux_acces_brut_premiere_bac       0.00000000 0.000000000 0.000000000
## taux_acces_attendu_premiere_bac    0.00000000 0.000000000 0.000000000
## taux_brut_de_reussite_total_series 0.00000000 0.000000000 1.000000000
## taux_reussite_attendu_total_series 0.00000000 0.000000000 0.000000000
## PMP (Exact)                        0.01058561 0.007557905 0.006514058
## PMP (MCMC)                         0.00929000 0.007260000 0.008300000
##                                         00800
## effectif_presents_serie_l          0.000000000
## effectif_presents_serie_es         0.000000000
## effectif_presents_serie_s          0.000000000
## taux_brut_de_reussite_serie_l      0.000000000
## taux_brut_de_reussite_serie_es     0.000000000
## taux_brut_de_reussite_serie_s      1.000000000
## taux_reussite_attendu_serie_l      0.000000000
## taux_reussite_attendu_serie_es     0.000000000
## taux_reussite_attendu_serie_s      0.000000000
## effectif_de_seconde                0.000000000
## effectif_de_premiere               0.000000000
## taux_acces_brut_seconde_bac        0.000000000
## taux_acces_attendu_seconde_bac     0.000000000
## taux_acces_brut_premiere_bac       0.000000000
## taux_acces_attendu_premiere_bac    0.000000000
## taux_brut_de_reussite_total_series 0.000000000
## taux_reussite_attendu_total_series 0.000000000
## PMP (Exact)                        0.006319245
## PMP (MCMC)                         0.005020000
```

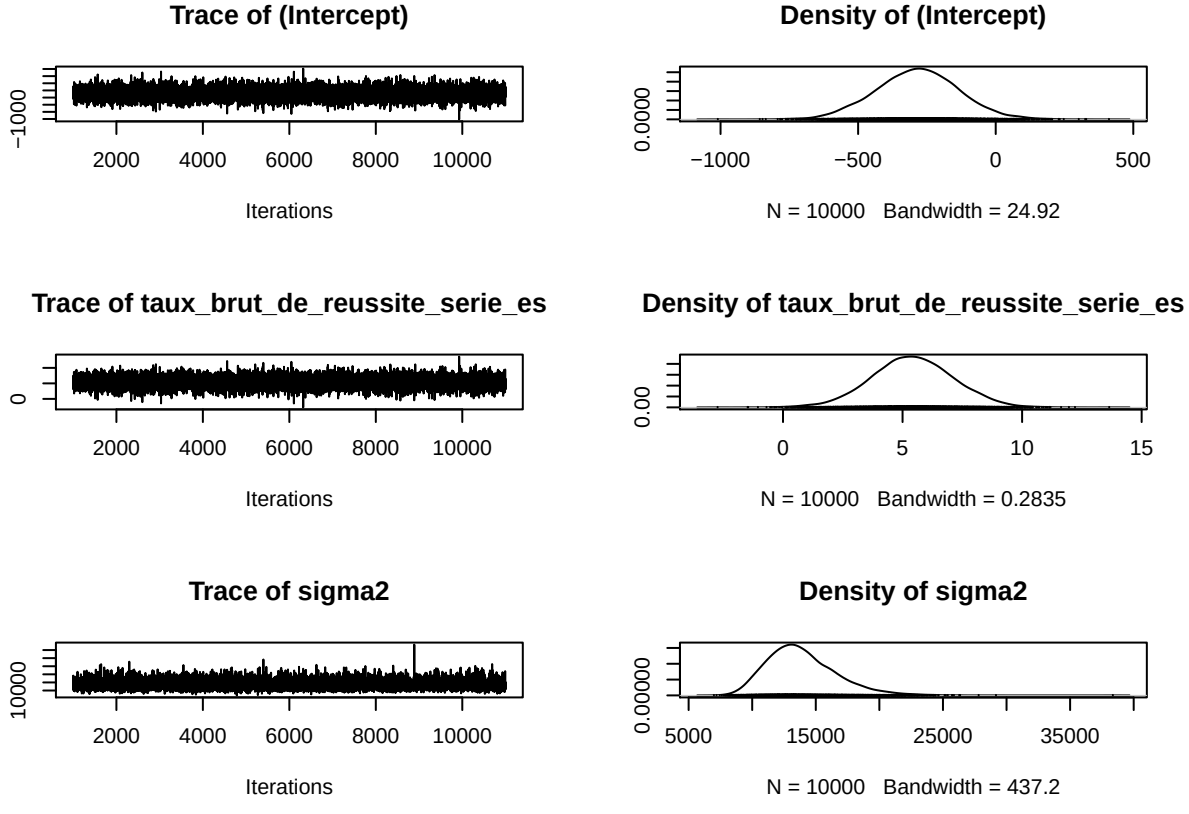
Le modèle le plus probable est celui qui ne contient aucune covariable !! Cela n'annonce rien de bon..
Je sélectionne le modèle non vide le plus probable. En l'occurence, celui contenant une seule covariable "taux_brut_de_reussite_serie_es".

```
reg0.mat.select=MCMCregress(Barre~taux_brut_de_reussite_serie_es, data=mut_mat2)
```

```
summary(reg0.mat.select)
```

```
##
## Iterations = 1001:11000
## Thinning interval = 1
## Number of chains = 1
## Sample size per chain = 10000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                                    Mean      SD Naive SE Time-series SE
## (Intercept)                    -285.072  150.72   1.5072         1.5012
## taux_brut_de_reussite_serie_es    5.418    1.72   0.0172         0.0172
## sigma2                        13884.427 2719.84  27.1984        27.6607
##
## 2. Quantiles for each variable:
##
##                                    2.5%      25%       50%       75%     97.5%
## (Intercept)                    -580.206  -384.25  -283.195  -185.503     7.103
## taux_brut_de_reussite_serie_es    2.028     4.29     5.403     6.551     8.782
## sigma2                         9580.197 11969.98 13532.124 15457.317 20146.375
```

```
plot(reg0.mat.select)
```



Il ne semble y avoir aucune structure particulière dans le graph. des traces. Ces dernières semblent “osciller convenablement”, signe que l’algorithme fonctionne normalement en “allant chercher des points de la densité à postériori ni trop près, ni trop loin”.

```
effectiveSize(reg0.mat.select)
```

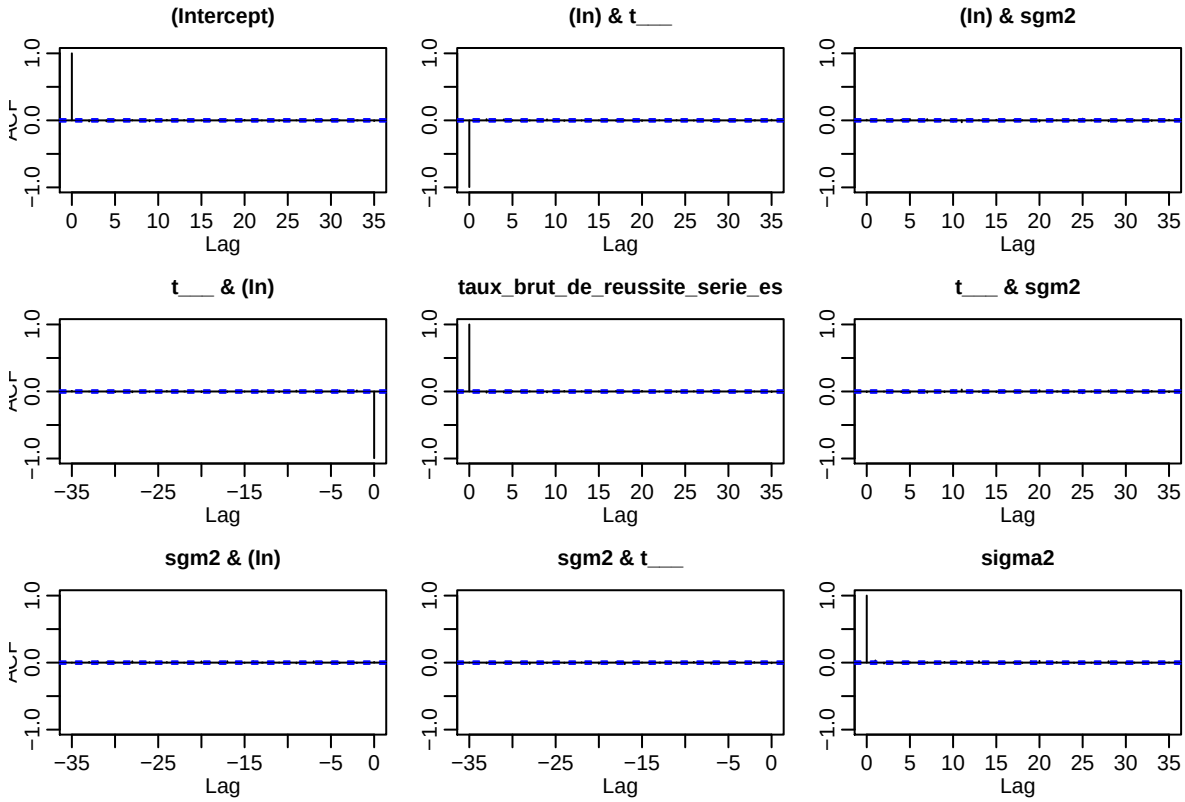
```
##                     (Intercept) taux_brut_de_reussite_serie_es
##                       10079.583                       10000.000
##                          sigma2
##                        9668.484
```

```
raftery.diag(reg0.mat.select)
```

```
##
## Quantile (q) = 0.025
## Accuracy (r) = +/- 0.005
## Probability (s) = 0.95
##
##                                 Burn-in  Total Lower bound  Dependence
##                                 (M)      (N)   (Nmin)       factor (I)
##  (Intercept)                    2        3802  3746         1.01
##  taux_brut_de_reussite_serie_es 2        3834  3746         1.02
##  sigma2                         2        3710  3746         0.99
```

Il n'y a pas de souci pour le «taux de change» entre les échantillons provenant de la MCMC et des échantillons indépendants.

```
acf(reg0.mat.select)
```



Les graphiques de convergence de la MCMC indiquent que l'algorithme a convergé. Ceux d'autocorrelation des résidus sont assez bons. La chaîne oublie très rapidement son passé. Néanmoins le modèle, lui, reste inadapté.

3.2 Anglais

```
reg_ang2.0=lm(mut_ang2$Barre~., data=mut_ang2)
summary(reg_ang2.0)
```

regression linéaire

```
##
## Call:
## lm(formula = mut_ang2$Barre ~ ., data = mut_ang2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -363.03 -133.21  -24.75   97.93 1044.32
##
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      -1286.4382  1860.5131  -0.691   0.4940
## effectif_presents_serie_l           -5.2435     5.6318  -0.931   0.3584
## effectif_presents_serie_es          -1.6538     2.7501  -0.601   0.5516
## effectif_presents_serie_s            0.7911     2.3383   0.338   0.7372
## taux_brut_de_reussite_serie_l       -5.1155     5.8121  -0.880   0.3850
## taux_brut_de_reussite_serie_es       5.9949     8.9053   0.673   0.5054
## taux_brut_de_reussite_serie_s        3.0029    14.7626   0.203   0.8400
## taux_reussite_attendu_serie_l       -3.7900    19.4963  -0.194   0.8470
## taux_reussite_attendu_serie_es      30.9369    17.5094   1.767   0.0862 .
## taux_reussite_attendu_serie_s      -24.6363    20.1117  -1.225   0.2290
## effectif_de_seconde                  1.2467     1.5277   0.816   0.4202
## effectif_de_premiere                -1.0171     1.9638  -0.518   0.6079
## taux_acces_brut_seconde_bac          8.7706    13.6909   0.641   0.5261
## taux_acces_attendu_seconde_bac     -30.4301    23.0086  -1.323   0.1948
## taux_acces_brut_premiere_bac       -42.0258    25.2706  -1.663   0.1055
## taux_acces_attendu_premiere_bac    103.0319    52.3436   1.968   0.0572 .
## taux_brut_de_reussite_total_series  32.8251    31.4412   1.044   0.3038
## taux_reussite_attendu_total_series -62.3386    54.2693  -1.149   0.2587
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 287.9 on 34 degrees of freedom
## Multiple R-squared:  0.3659, Adjusted R-squared:  0.0489
## F-statistic: 1.154 on 17 and 34 DF,  p-value: 0.3492
```

```r
# sélection automatique
reg_ang2.1=stepAIC(reg_ang2.0,~., data=mut_ang2,trace=F,direction=c("backward"))
summary(reg_ang2.1)
```

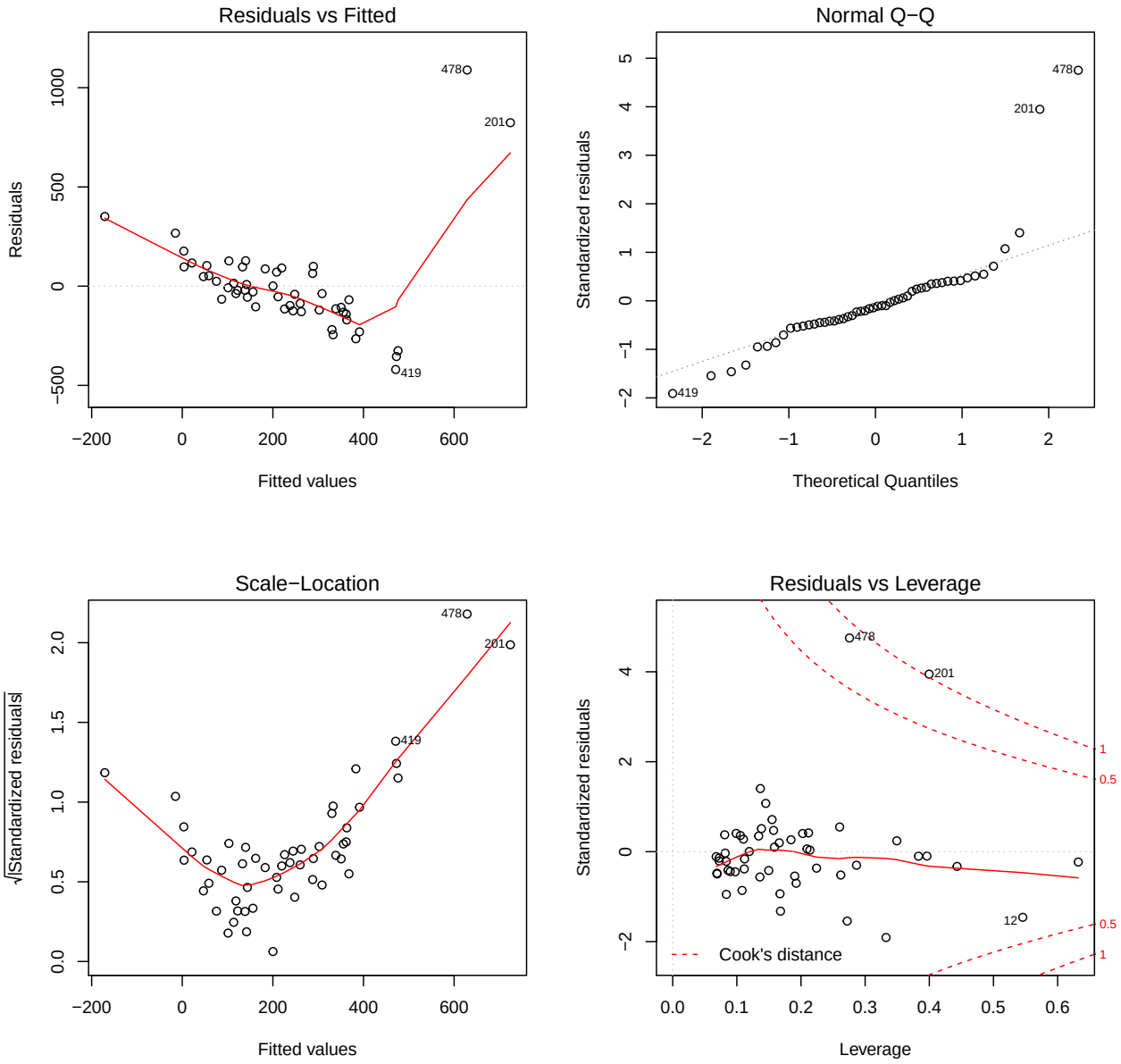
```
##
## Call:
## lm(formula = mut_ang2$Barre ~ effectif_presents_serie_l + taux_brut_de_reussite_serie_l +
##     taux_reussite_attendu_serie_es + taux_reussite_attendu_serie_s +
##     taux_acces_attendu_seconde_bac + taux_acces_brut_premiere_bac +
##     taux_acces_attendu_premiere_bac + taux_brut_de_reussite_total_series +
##     taux_reussite_attendu_total_series, data = mut_ang2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -420.1 -116.3  -33.2   88.1 1089.2
##
## Coefficients:
##                                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)                       -1696.950    822.088  -2.064  0.04521 *
## effectif_presents_serie_l            -3.095      2.219  -1.395  0.17038
## taux_brut_de_reussite_serie_l        -6.030      4.225  -1.427  0.16087
## taux_reussite_attendu_serie_es       37.414     13.888   2.694  0.01010 *
## taux_reussite_attendu_serie_s       -26.011     15.296  -1.700  0.09643 .
## taux_acces_attendu_seconde_bac      -26.632     15.037  -1.771  0.08381 .
## taux_acces_brut_premiere_bac        -29.358     16.041  -1.830  0.07432 .
## taux_acces_attendu_premiere_bac     101.217     34.272   2.953  0.00513 **
## taux_brut_de_reussite_total_series   44.714     17.511   2.553  0.01439 *
## taux_reussite_attendu_total_series  -74.814     32.156  -2.327  0.02488 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 269.3 on 42 degrees of freedom
## Multiple R-squared:  0.3146, Adjusted R-squared:  0.1677
## F-statistic: 2.142 on 9 and 42 DF,  p-value: 0.04689
```

```r
AIC(reg_ang2.1)
```

```
## [1] 740.4428
```

le R^2_{adj} possède toujours une valeur relativement faible...

```
par(mfrow=c(2,2))
plot(reg_ang2.1)
```



2 outliers perturbent totalement les graphs de gregression. Il s'agit des individus 201 et 478. Le QQ-plot n'est pas si mauvais si on enlève valeurs ces valeurs extrêmes. Bien qu'ils soient plutôt bien répartis, ils semblent avoir une "tendance" remettant en cause l'hypothèse d'indépendance des résidus.

Que se passe-t-il en enlevant ces individus extrêmes ?

```
mut_ang.bis=mut[-c(201,478),]
mut_ang.bis=mut_ang.bis[(mut_ang.bis$Matiere=="ANGLAIS"),]
# restrictions aux varaiables quantitatives
mut_ang2.bis=mut_ang.bis[,6:23]
```

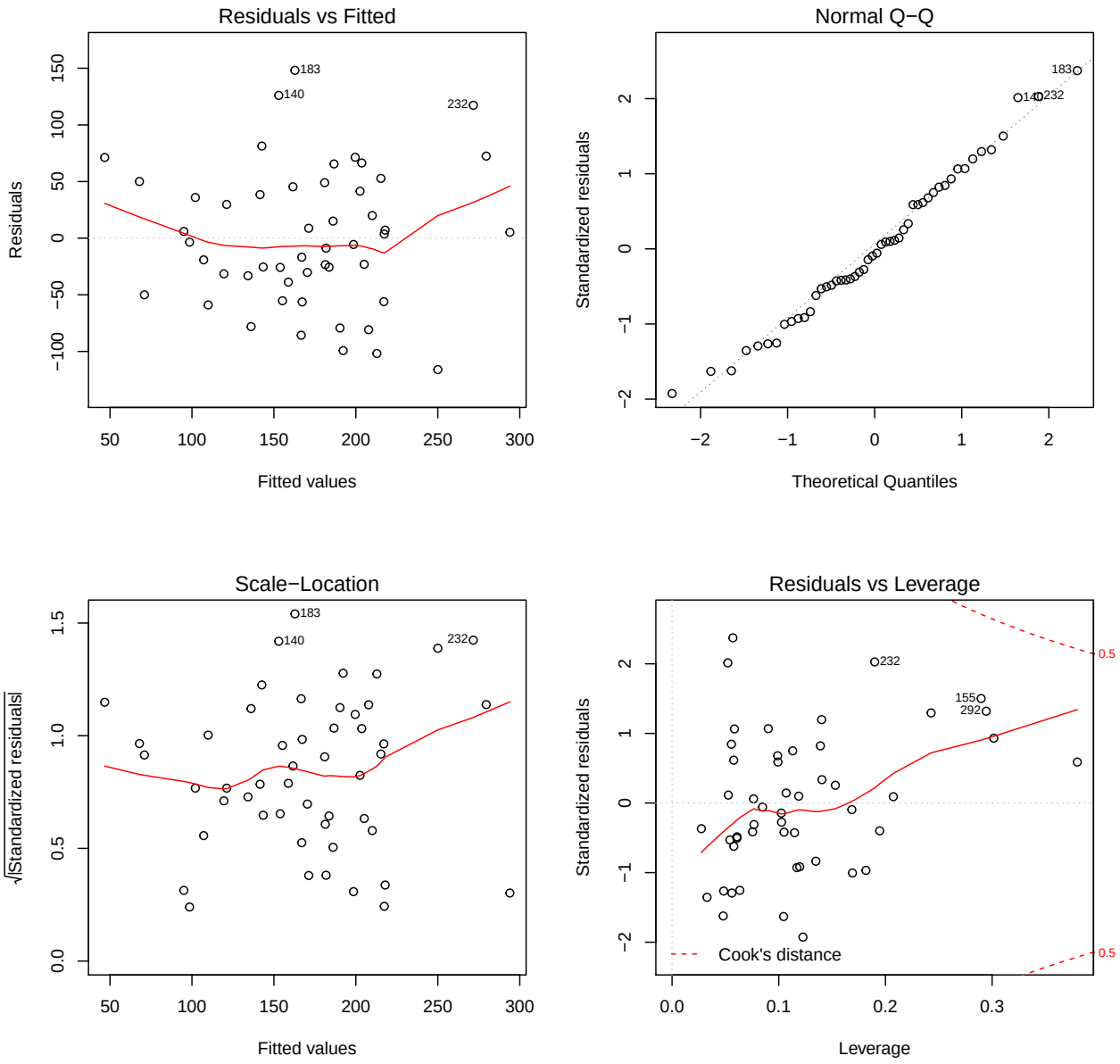
```
reg_ang2.2.select=stepAIC(lm(mut_ang2.bis$Barre~., data=mut_ang2.bis)
                          ,~., data=mut_ang2.bis,trace=F,direction=c("backward"))
summary(reg_ang2.2.select)
```

```
##
## Call:
## lm(formula = mut_ang2.bis$Barre ~ effectif_presents_serie_es +
##     effectif_presents_serie_s + taux_brut_de_reussite_serie_es +
##     taux_acces_attendu_seconde_bac + taux_acces_attendu_premiere_bac,
##     data = mut_ang2.bis)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -116.029  -37.402   -4.549   44.391  148.163
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     -821.3430   195.9371  -4.192 0.000131 ***
## effectif_presents_serie_es         1.4666     0.5341   2.746 0.008705 **
## effectif_presents_serie_s         -0.9206     0.3924  -2.346 0.023554 *
## taux_brut_de_reussite_serie_es     1.7136     1.0884   1.574 0.122561
## taux_acces_attendu_seconde_bac    -7.8245     3.5668  -2.194 0.033586 *
## taux_acces_attendu_premiere_bac   16.3131     4.3141   3.781 0.000467 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 64.3 on 44 degrees of freedom
## Multiple R-squared:  0.4239, Adjusted R-squared:  0.3585
## F-statistic: 6.476 on 5 and 44 DF,  p-value: 0.0001365
```

```
AIC(reg_ang2.2.select)
```

```
## [1] 565.8573
```

```
par(mfrow=c(2,2))
plot(reg_ang2.2.select)
```



Les graphs sont un peu meilleurs d'une manière générale. Le QQ-plot devient plus qu'acceptable.La répartition des résidus se fait plutôt bien de part et d'autres des régressions locales.

Néanmoins la faible valeur du R^2_{adj} m'enclint a délaisser ce modèle.

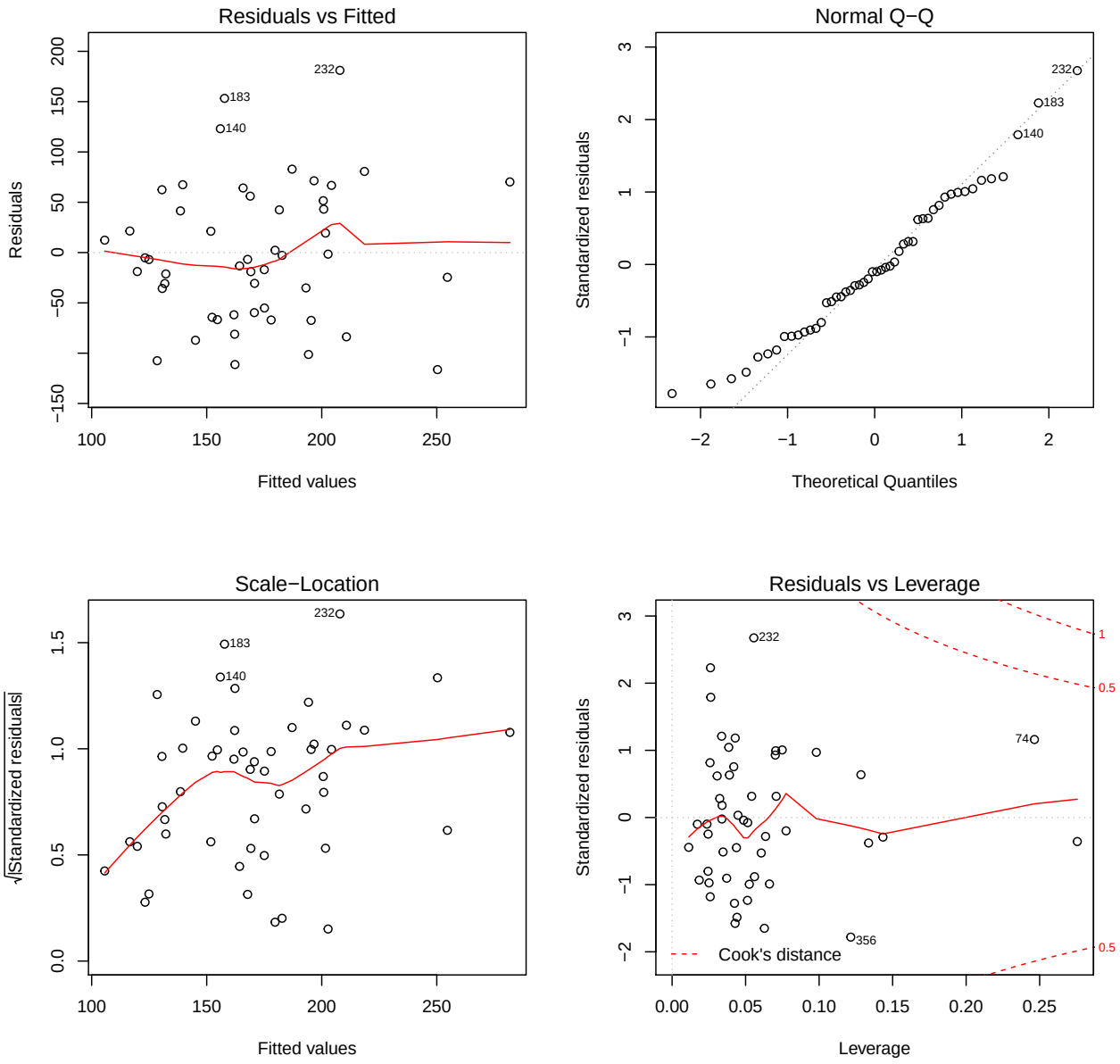
Pour le fun.. Après avoir ôté du modèle quelques covariables trop corrélées, j'obtiens les modèles suivants :

```
par(mfrow=c(2,2))
# avec sélection de quelques covariables trop correlés aux autres
reg_ang2.1=lm(mut_ang2.bis$Barre~. -1-(taux_brut_de_reussite_serie_s +effectif_presents_serie_l +taux_r

# sélection automatique
reg_ang2.2=stepAIC(reg_ang2.1,~., data=mut_ang2.bis,trace=F,direction=c("backward"))
summary(reg_ang2.2)
```

```
##
## Call:
## lm(formula = mut_ang2.bis$Barre ~ effectif_presents_serie_es +
##     taux_reussite_attendu_serie_s + effectif_de_premiere - 1,
##     data = mut_ang2.bis)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -116.311  -58.549   -6.843   49.441  181.102
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## effectif_presents_serie_es      1.4428     0.4978   2.898  0.00568 **
## taux_reussite_attendu_serie_s   1.8685     0.3247   5.754 6.34e-07 ***
## effectif_de_premiere           -0.3189     0.1405  -2.270  0.02786 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 69.67 on 47 degrees of freedom
## Multiple R-squared:  0.8707, Adjusted R-squared:  0.8624
## F-statistic: 105.5 on 3 and 47 DF,  p-value: < 2.2e-16
```

```
plot(reg_ang2.2)
```



En sélectionnant quelques covariables, le R^2_{adj} s'améliore grandement. Il est peut-être plus facile de trouver un modèle linéaire pour cette discipline que dans le cadre "pluridisciplinaire"

Néanmoins, en étudiant les graphiques, les résidus sont plutôt bien répartis et les distances de Cook semblent acceptables. C'est cette fois-ci la gaussianité des résidus qui pose problème. Le QQ-plott n'est pas bon. Dommage….

```
reg0.ang=MCMCregress(Barre~., data=mut_ang2.bis)
summary(reg0.ang)
```

Algorithme MCMC

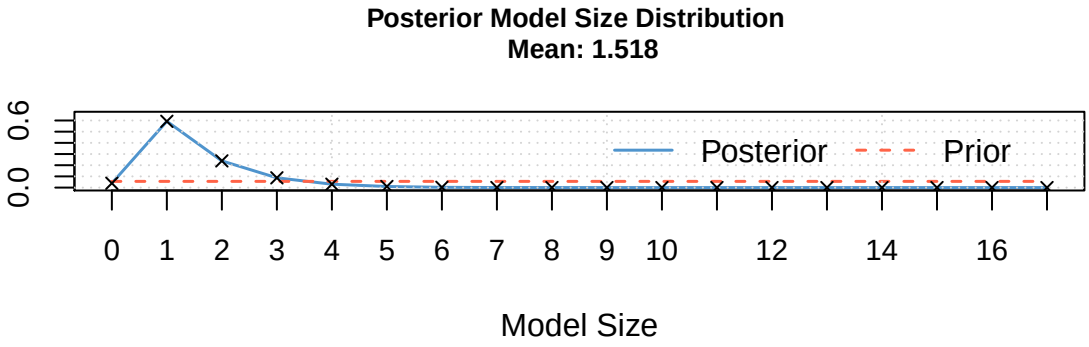
```
##
## Iterations = 1001:11000
## Thinning interval = 1
## Number of chains = 1
## Sample size per chain = 10000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                                       Mean        SD  Naive SE
## (Intercept)                     -1.140e+03  491.9911  4.919911
## effectif_presents_serie_l       -9.470e-01    1.4661  0.014661
## effectif_presents_serie_es       1.537e+00    0.7231  0.007231
## effectif_presents_serie_s       -8.231e-01    0.6168  0.006168
## taux_brut_de_reussite_serie_l   -6.052e-01    1.6094  0.016094
## taux_brut_de_reussite_serie_es   1.043e+00    2.2947  0.022947
## taux_brut_de_reussite_serie_s   -2.258e+00    3.8121  0.038121
## taux_reussite_attendu_serie_l    3.970e+00    5.0384  0.050384
## taux_reussite_attendu_serie_es  -2.077e+00    4.7490  0.047490
## taux_reussite_attendu_serie_s    5.041e-01    5.2897  0.052897
## effectif_de_seconde             -4.361e-01    0.4053  0.004053
## effectif_de_premiere             5.656e-01    0.5158  0.005158
## taux_acces_brut_seconde_bac     -3.178e+00    3.5313  0.035313
## taux_acces_attendu_seconde_bac  -9.406e+00    5.9387  0.059387
## taux_acces_brut_premiere_bac     7.285e+00    6.8628  0.068628
## taux_acces_attendu_premiere_bac  2.460e+01   14.0266  0.140266
## taux_brut_de_reussite_total_series  5.301e-02    8.1389  0.081389
## taux_reussite_attendu_total_series -6.679e+00   14.1556  0.141556
## sigma2                           5.436e+03 1456.5165 14.565165
##                                 Time-series SE
## (Intercept)                           5.067089
## effectif_presents_serie_l             0.014661
## effectif_presents_serie_es            0.007231
## effectif_presents_serie_s             0.006168
## taux_brut_de_reussite_serie_l         0.015589
## taux_brut_de_reussite_serie_es        0.022947
## taux_brut_de_reussite_serie_s         0.037226
## taux_reussite_attendu_serie_l         0.048846
## taux_reussite_attendu_serie_es        0.047065
## taux_reussite_attendu_serie_s         0.052115
## effectif_de_seconde                   0.004053
## effectif_de_premiere                  0.005158
## taux_acces_brut_seconde_bac           0.035313
## taux_acces_attendu_seconde_bac        0.059387
## taux_acces_brut_premiere_bac          0.069703
## taux_acces_attendu_premiere_bac       0.145488
## taux_brut_de_reussite_total_series    0.081389
## taux_reussite_attendu_total_series    0.144644
```

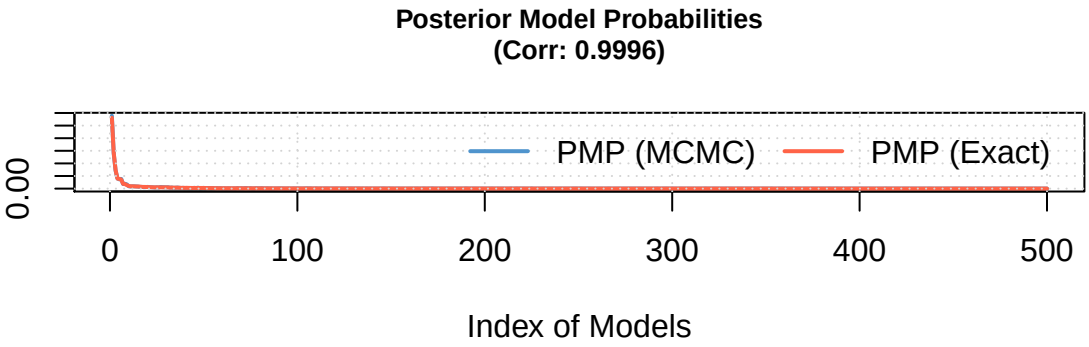
```
## sigma2                                  21.900828
##
## 2. Quantiles for each variable:
##
##                                             2.5%        25%        50%        75%
## (Intercept)                            -2114.6501 -1470.3131 -1.143e+03 -818.84758
## effectif_presents_serie_l                 -3.8571    -1.9096 -9.391e-01    0.03799
## effectif_presents_serie_es                 0.1131     1.0620  1.542e+00    2.01237
## effectif_presents_serie_s                 -2.0316    -1.2300 -8.310e-01   -0.40938
## taux_brut_de_reussite_serie_l             -3.7686    -1.6802 -6.072e-01    0.45708
## taux_brut_de_reussite_serie_es            -3.4840    -0.4533  1.033e+00    2.54309
## taux_brut_de_reussite_serie_s             -9.6982    -4.7672 -2.273e+00    0.22267
## taux_reussite_attendu_serie_l             -5.8682     0.6735  3.957e+00    7.27098
## taux_reussite_attendu_serie_es           -11.4394    -5.1997 -2.050e+00    1.04454
## taux_reussite_attendu_serie_s             -9.8126    -3.0343  4.199e-01    3.98526
## effectif_de_seconde                       -1.2176    -0.7003 -4.382e-01   -0.16587
## effectif_de_premiere                      -0.4633     0.2257  5.746e-01    0.91142
## taux_acces_brut_seconde_bac              -10.0680    -5.5305 -3.162e+00   -0.76145
## taux_acces_attendu_seconde_bac           -21.1433   -13.4395 -9.427e+00   -5.49549
## taux_acces_brut_premiere_bac              -6.3395     2.7976  7.269e+00   11.82706
## taux_acces_attendu_premiere_bac           -3.4003    15.2989  2.475e+01   33.74500
## taux_brut_de_reussite_total_series       -15.8749    -5.4000 -4.990e-03    5.51390
## taux_reussite_attendu_total_series       -34.2953   -16.2113 -6.573e+00    2.65189
## sigma2                                  3295.9040  4412.7871  5.187e+03 6234.54076
##                                            97.5%
## (Intercept)                            -166.2137
## effectif_presents_serie_l                 1.8995
## effectif_presents_serie_es                2.9597
## effectif_presents_serie_s                 0.3932
## taux_brut_de_reussite_serie_l             2.5718
## taux_brut_de_reussite_serie_es            5.5851
## taux_brut_de_reussite_serie_s             5.2096
## taux_reussite_attendu_serie_l            14.0765
## taux_reussite_attendu_serie_es            7.3413
## taux_reussite_attendu_serie_s            11.1418
## effectif_de_seconde                       0.3539
## effectif_de_premiere                      1.5702
## taux_acces_brut_seconde_bac               3.6587
## taux_acces_attendu_seconde_bac            2.3180
## taux_acces_brut_premiere_bac             20.7075
## taux_acces_attendu_premiere_bac          52.1366
## taux_brut_de_reussite_total_series       15.9768
## taux_reussite_attendu_total_series       21.7595
## sigma2                                 8903.7860
```

```
reg0.ang.bms=bms(Barre~., data=mut_ang2.bis, burn = 1e4, iter=1e5)
```

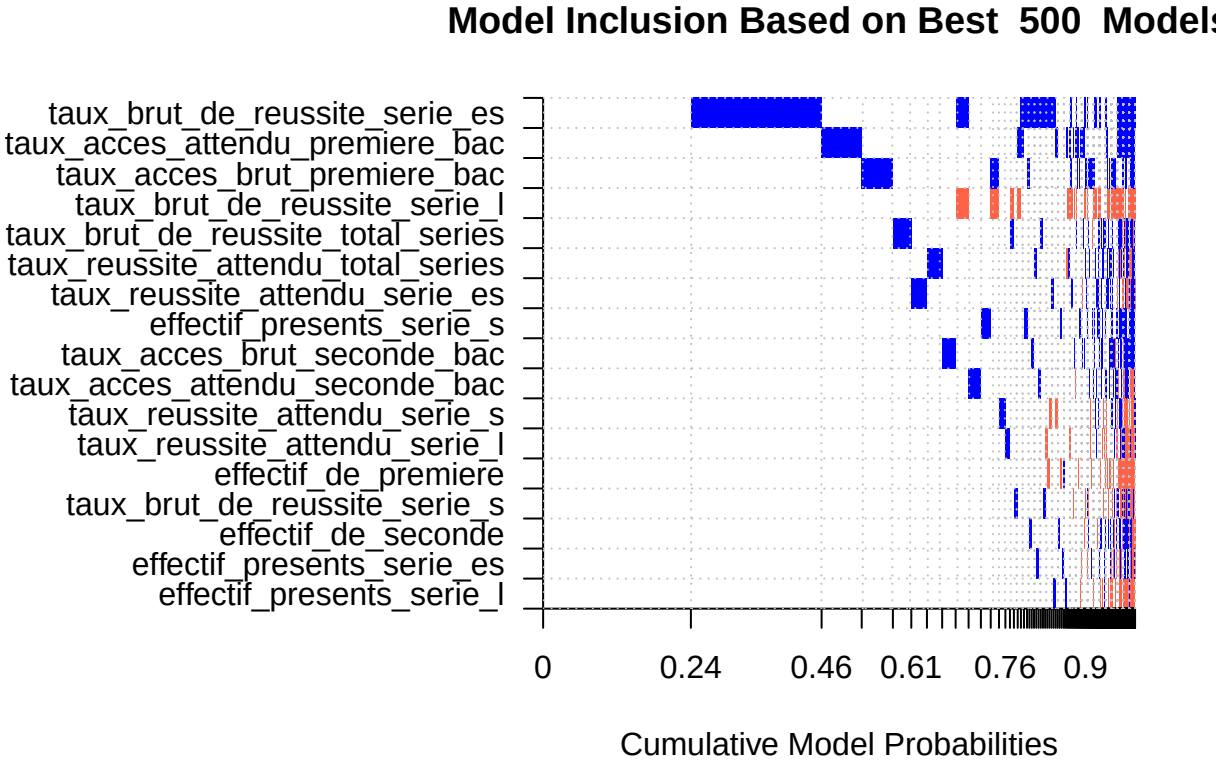
```
##                                        PIP     Post Mean    Post SD
## taux_acces_attendu_premiere_bac    0.49708  4.1748501368 5.22722868
## taux_reussite_attendu_total_series 0.22318  1.1327571545 2.54914045
## taux_reussite_attendu_serie_s      0.12386  0.4370889790 1.46846163
## taux_acces_attendu_seconde_bac     0.10326 -0.5771005615 2.40812901
## effectif_presents_serie_es         0.08115  0.0625631214 0.26646179
## taux_reussite_attendu_serie_es     0.07035  0.2067471371 1.05302468
```

```
## taux_brut_de_reussite_serie_s      0.06544 -0.1706248148 0.81956128
## taux_acces_brut_premiere_bac       0.04982  0.0913630899 0.97974696
## taux_brut_de_reussite_total_series 0.04745  0.0181647206 0.94635143
## taux_brut_de_reussite_serie_l      0.03813 -0.0395133141 0.27889954
## taux_brut_de_reussite_serie_es     0.03773  0.0542498050 0.37797761
## taux_reussite_attendu_serie_l      0.03543  0.0490936731 0.60399963
## effectif_presents_serie_s          0.03388 -0.0068847217 0.10485789
## effectif_de_premiere               0.03275 -0.0031327150 0.03977263
## effectif_de_seconde                0.02950 -0.0022239835 0.03022962
## taux_acces_brut_seconde_bac        0.02653 -0.0137977399 0.33373692
## effectif_presents_serie_l          0.02244 -0.0003956839 0.11072492
##                                    Cond.Pos.Sign Idx
## taux_acces_attendu_premiere_bac       1.00000000  15
## taux_reussite_attendu_total_series    0.95125011  17
## taux_reussite_attendu_serie_s         0.88801873   9
## taux_acces_attendu_seconde_bac        0.14129382  13
## effectif_presents_serie_es            1.00000000   2
## taux_reussite_attendu_serie_es        0.82103767   8
## taux_brut_de_reussite_serie_s         0.03468826   6
## taux_acces_brut_premiere_bac          0.62364512  14
## taux_brut_de_reussite_total_series    0.48451001  16
## taux_brut_de_reussite_serie_l         0.00839234   4
## taux_brut_de_reussite_serie_es        1.00000000   5
## taux_reussite_attendu_serie_l         0.62348292   7
## effectif_presents_serie_s             0.35242031   3
## effectif_de_premiere                  0.20335878  11
## effectif_de_seconde                   0.16915254  10
## taux_acces_brut_seconde_bac           0.23105918  12
## effectif_presents_serie_l             0.42959002   1
##
## Mean no. regressors               Draws             Burnins                Time
##            "1.5180"             "1e+05"             "10000"    "3.922986 secs"
##  No. models visited      Modelspace 2^K           % visited         % Topmodels
##             "12902"            "131072"               "9.8"                "97"
##            Corr PMP           No. Obs.         Model Prior             g-Prior
##            "0.9996"                "50"      "random / 8.5"               "UIP"
##     Shrinkage-Stats
##         "Av=0.9804"
##
## Time difference of 3.922986 secs
```






```
image(reg0.mat.bms)
```



```
topmodels.bma(reg0.ang.bms)[,1:8]
```

```
##                                      00004     00001      00100      00000
## effectif_presents_serie_l          0.0000000 0.0000000 0.00000000 0.00000000
## effectif_presents_serie_es         0.0000000 0.0000000 0.00000000 0.00000000
## effectif_presents_serie_s          0.0000000 0.0000000 0.00000000 0.00000000
## taux_brut_de_reussite_serie_l      0.0000000 0.0000000 0.00000000 0.00000000
## taux_brut_de_reussite_serie_es     0.0000000 0.0000000 0.00000000 0.00000000
## taux_brut_de_reussite_serie_s      0.0000000 0.0000000 0.00000000 0.00000000
## taux_reussite_attendu_serie_l      0.0000000 0.0000000 0.00000000 0.00000000
## taux_reussite_attendu_serie_es     0.0000000 0.0000000 0.00000000 0.00000000
## taux_reussite_attendu_serie_s      0.0000000 0.0000000 1.00000000 0.00000000
## effectif_de_seconde                0.0000000 0.0000000 0.00000000 0.00000000
## effectif_de_premiere               0.0000000 0.0000000 0.00000000 0.00000000
## taux_acces_brut_seconde_bac        0.0000000 0.0000000 0.00000000 0.00000000
## taux_acces_attendu_seconde_bac     0.0000000 0.0000000 0.00000000 0.00000000
## taux_acces_brut_premiere_bac       0.0000000 0.0000000 0.00000000 0.00000000
## taux_acces_attendu_premiere_bac    1.0000000 0.0000000 0.00000000 0.00000000
## taux_brut_de_reussite_total_series 0.0000000 0.0000000 0.00000000 0.00000000
## taux_reussite_attendu_total_series 0.0000000 1.0000000 0.00000000 0.00000000
## PMP (Exact)                        0.2715726 0.1367755 0.07060375 0.03844561
## PMP (MCMC)                         0.2791200 0.1345700 0.06901000 0.03876000
##                                      00014      00200      00804      00008
## effectif_presents_serie_l          0.00000000 0.00000000 0.00000000 0.00000000
## effectif_presents_serie_es         0.00000000 0.00000000 0.00000000 0.00000000
## effectif_presents_serie_s          0.00000000 0.00000000 0.00000000 0.00000000
## taux_brut_de_reussite_serie_l      0.00000000 0.00000000 0.00000000 0.00000000
## taux_brut_de_reussite_serie_es     0.00000000 0.00000000 0.00000000 0.00000000
## taux_brut_de_reussite_serie_s      0.00000000 0.00000000 1.00000000 0.00000000
## taux_reussite_attendu_serie_l      0.00000000 0.00000000 0.00000000 0.00000000
## taux_reussite_attendu_serie_es     0.00000000 1.00000000 0.00000000 0.00000000
## taux_reussite_attendu_serie_s      0.00000000 0.00000000 0.00000000 0.00000000
## effectif_de_seconde                0.00000000 0.00000000 0.00000000 0.00000000
## effectif_de_premiere               0.00000000 0.00000000 0.00000000 0.00000000
## taux_acces_brut_seconde_bac        0.00000000 0.00000000 0.00000000 0.00000000
## taux_acces_attendu_seconde_bac     1.00000000 0.00000000 0.00000000 0.00000000
## taux_acces_brut_premiere_bac       0.00000000 0.00000000 0.00000000 1.00000000
## taux_acces_attendu_premiere_bac    1.00000000 0.00000000 1.00000000 0.00000000
## taux_brut_de_reussite_total_series 0.00000000 0.00000000 0.00000000 0.00000000
## taux_reussite_attendu_total_series 0.00000000 0.00000000 0.00000000 0.00000000
## PMP (Exact)                        0.03703993 0.03454282 0.01839784 0.01725959
## PMP (MCMC)                         0.03530000 0.03743000 0.01738000 0.01841000
```

```
topmodels.bma(reg0.ang.bms)[,9:16]
```

```
##                                       00002      00010       00801
## effectif_presents_serie_l          0.00000000 0.00000000 0.000000000
## effectif_presents_serie_es         0.00000000 0.00000000 0.000000000
## effectif_presents_serie_s          0.00000000 0.00000000 0.000000000
## taux_brut_de_reussite_serie_l      0.00000000 0.00000000 0.000000000
## taux_brut_de_reussite_serie_es     0.00000000 0.00000000 0.000000000
## taux_brut_de_reussite_serie_s      0.00000000 0.00000000 1.000000000
## taux_reussite_attendu_serie_l      0.00000000 0.00000000 0.000000000
## taux_reussite_attendu_serie_es     0.00000000 0.00000000 0.000000000
## taux_reussite_attendu_serie_s      0.00000000 0.00000000 0.000000000
## effectif_de_seconde                0.00000000 0.00000000 0.000000000
## effectif_de_premiere               0.00000000 0.00000000 0.000000000
## taux_acces_brut_seconde_bac        0.00000000 0.00000000 0.000000000
## taux_acces_attendu_seconde_bac     0.00000000 1.00000000 0.000000000
## taux_acces_brut_premiere_bac       0.00000000 0.00000000 0.000000000
## taux_acces_attendu_premiere_bac    0.00000000 0.00000000 0.000000000
## taux_brut_de_reussite_total_series 1.00000000 0.00000000 0.000000000
## taux_reussite_attendu_total_series 0.00000000 0.00000000 1.000000000
## PMP (Exact)                        0.01337454 0.01033895 0.008902224
## PMP (MCMC)                         0.01602000 0.00892000 0.008190000
##                                       08004       00400       00006
## effectif_presents_serie_l          0.000000000 0.000000000 0.000000000
## effectif_presents_serie_es         1.000000000 0.000000000 0.000000000
## effectif_presents_serie_s          0.000000000 0.000000000 0.000000000
## taux_brut_de_reussite_serie_l      0.000000000 0.000000000 0.000000000
## taux_brut_de_reussite_serie_es     0.000000000 0.000000000 0.000000000
## taux_brut_de_reussite_serie_s      0.000000000 0.000000000 0.000000000
## taux_reussite_attendu_serie_l      0.000000000 1.000000000 0.000000000
## taux_reussite_attendu_serie_es     0.000000000 0.000000000 0.000000000
## taux_reussite_attendu_serie_s      0.000000000 0.000000000 0.000000000
## effectif_de_seconde                0.000000000 0.000000000 0.000000000
## effectif_de_premiere               0.000000000 0.000000000 0.000000000
## taux_acces_brut_seconde_bac        0.000000000 0.000000000 0.000000000
## taux_acces_attendu_seconde_bac     0.000000000 0.000000000 0.000000000
## taux_acces_brut_premiere_bac       0.000000000 0.000000000 0.000000000
## taux_acces_attendu_premiere_bac    1.000000000 0.000000000 1.000000000
## taux_brut_de_reussite_total_series 0.000000000 0.000000000 1.000000000
## taux_reussite_attendu_total_series 0.000000000 0.000000000 0.000000000
## PMP (Exact)                        0.008724055 0.008638115 0.008147989
## PMP (MCMC)                         0.008820000 0.008550000 0.008200000
##                                       02004       01000
## effectif_presents_serie_l          0.000000000 0.000000000
## effectif_presents_serie_es         0.000000000 0.000000000
## effectif_presents_serie_s          0.000000000 0.000000000
## taux_brut_de_reussite_serie_l      1.000000000 0.000000000
## taux_brut_de_reussite_serie_es     0.000000000 1.000000000
## taux_brut_de_reussite_serie_s      0.000000000 0.000000000
## taux_reussite_attendu_serie_l      0.000000000 0.000000000
## taux_reussite_attendu_serie_es     0.000000000 0.000000000
## taux_reussite_attendu_serie_s      0.000000000 0.000000000
## effectif_de_seconde                0.000000000 0.000000000
## effectif_de_premiere               0.000000000 0.000000000
```

```
## taux_acces_brut_seconde_bac        0.000000000 0.000000000
## taux_acces_attendu_seconde_bac     0.000000000 0.000000000
## taux_acces_brut_premiere_bac       0.000000000 0.000000000
## taux_acces_attendu_premiere_bac    1.000000000 0.000000000
## taux_brut_de_reussite_total_series 0.000000000 0.000000000
## taux_reussite_attendu_total_series 0.000000000 0.000000000
## PMP (Exact)                        0.008083378 0.007481387
## PMP (MCMC)                         0.007750000 0.006460000
```

Je sélectionne le modèle non vide le plus probable. En l'occurence, celui contenant une seule covariable "taux_brut_de_reussite_serie_es".

```
reg0.ang.select=MCMCregress(Barre~taux_brut_de_reussite_serie_es, data=mut_ang2.bis)
```

```
raftery.diag(reg0.ang.select)
```

```
##
## Quantile (q) = 0.025
## Accuracy (r) = +/- 0.005
## Probability (s) = 0.95
##
##                                Burn-in  Total Lower bound  Dependence
##                                (M)      (N)   (Nmin)       factor (I)
##  (Intercept)                   2        3771  3746         1.01
##  taux_brut_de_reussite_serie_es 2        3897  3746         1.04
##  sigma2                        2        3802  3746         1.01
```

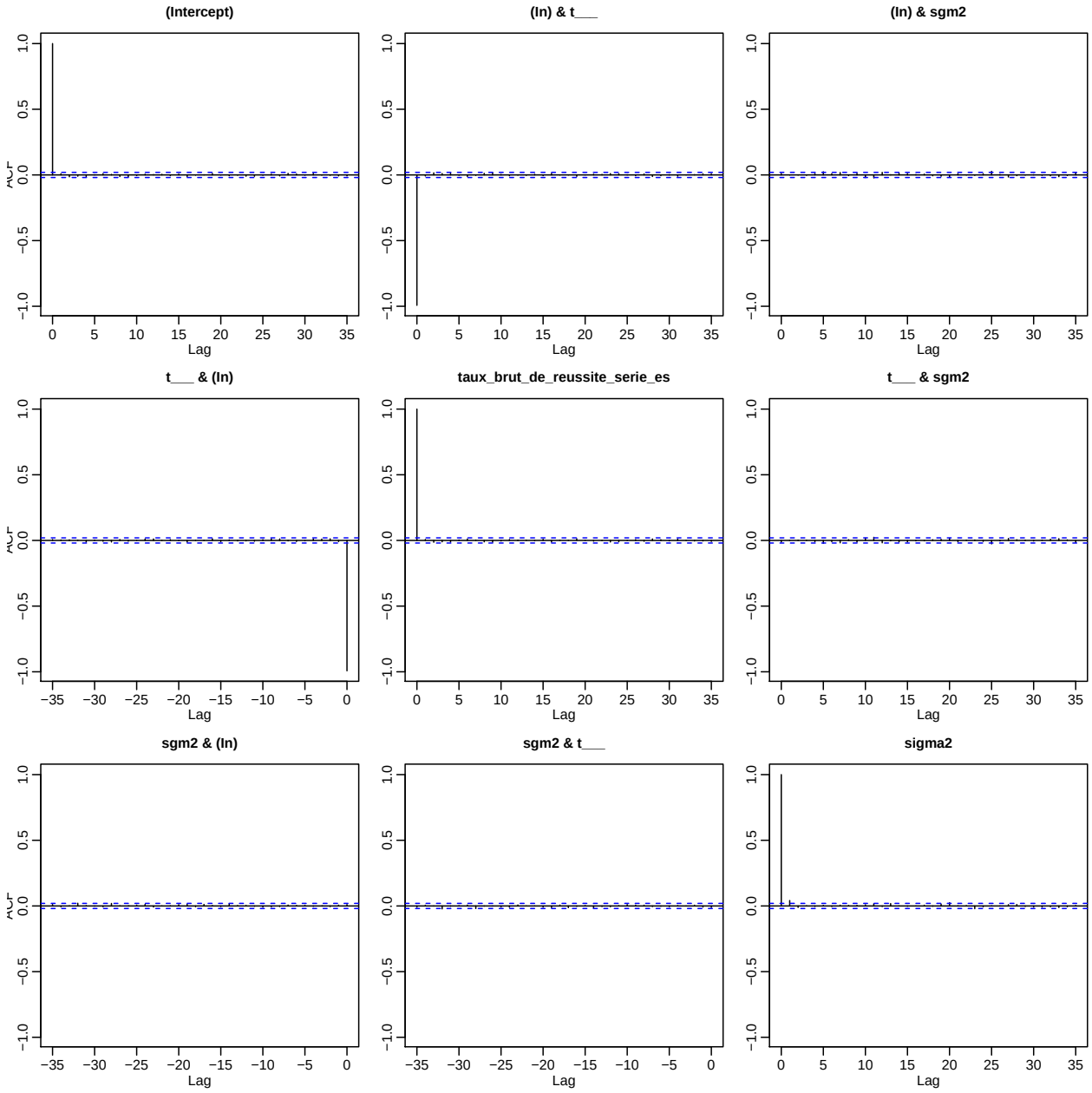
Il aurait fallu environ 3800 itérations pour faire fonctionner convenablement l'algorithlme de MCMC, ce qui est bien inférieur au 10 000 générées précédemment.
Il n' y a quasiment aucun "temps de chauffe" (Burn-in = 2),

Obeservons l'autocorrélation potentielle afin de vérifier si la chaine de markov "oublie rapidement son passé".

```
effectiveSize(reg0.ang.select)
```

```
##                    (Intercept) taux_brut_de_reussite_serie_es
##                      10000.000                       10000.000
##                         sigma2
##                       9206.437
```

```
acf(reg0.ang.select)
```



Il n'y a pas de souci pour le «taux de change» entre les échantillons provenant de la MCMC et des échantillons indépendants. Les graphiques de convergence de la MCMC indiquent que l'algorithme a convergé. Ceux d'autocorrelation des résidus sont assez bons. La chaîne oublie très rapidement son passé. Néanmoins le modèle, lui, reste inadapté.

3.3 “Maths==Anglais” ?

l'hypotèse que les covariables agissent de la même manière dans les deux disciplines semble improbable au vu des résultats précédents. Tentons de le vérifier.

```
par(mfrow=c(1, 2))
r=reg0.mat.select/reg0.ang.select
mean(r)
```

```
## [1] 2.529155
```

```
quantile(r, c(0.025,0.975))
```

```
##      2.5%     97.5%
## -13.44721  16.75311
```

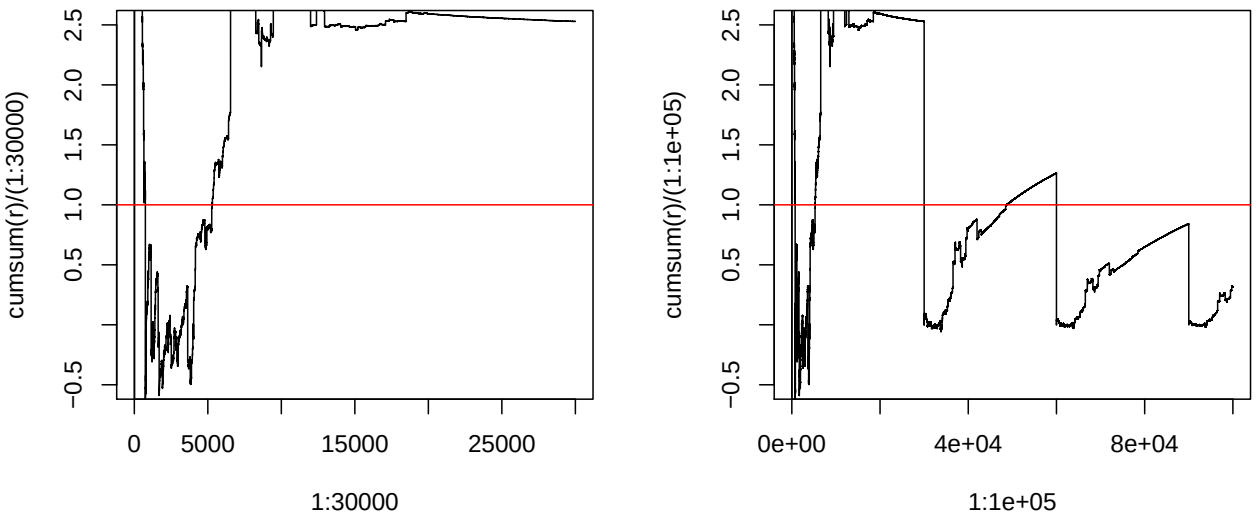
```
length(r)
```

```
## [1] 30000
```

```
plot(1:30000,cumsum(r)/(1:30000),type="l", ylim=c(-0.5,2.5))
abline(h=1,col="red")
plot(1:100000,cumsum(r)/(1:100000),type="l", ylim=c(-0.5,2.5))
```

```
## Warning in cumsum(r)/(1:1e+05): la taille d'un objet plus long n'est pas
## multiple de la taille d'un objet plus court
```

```
abline(h=1,col="red")
```



La moyenne de r s'éloigne de la valeur 1 à mesure que n grandit. L'hypothèse selon laquelle la covariable sélectionnée agit de la même manière sur les différentes matières semble déraisonnable.

II - Loi de Pareto

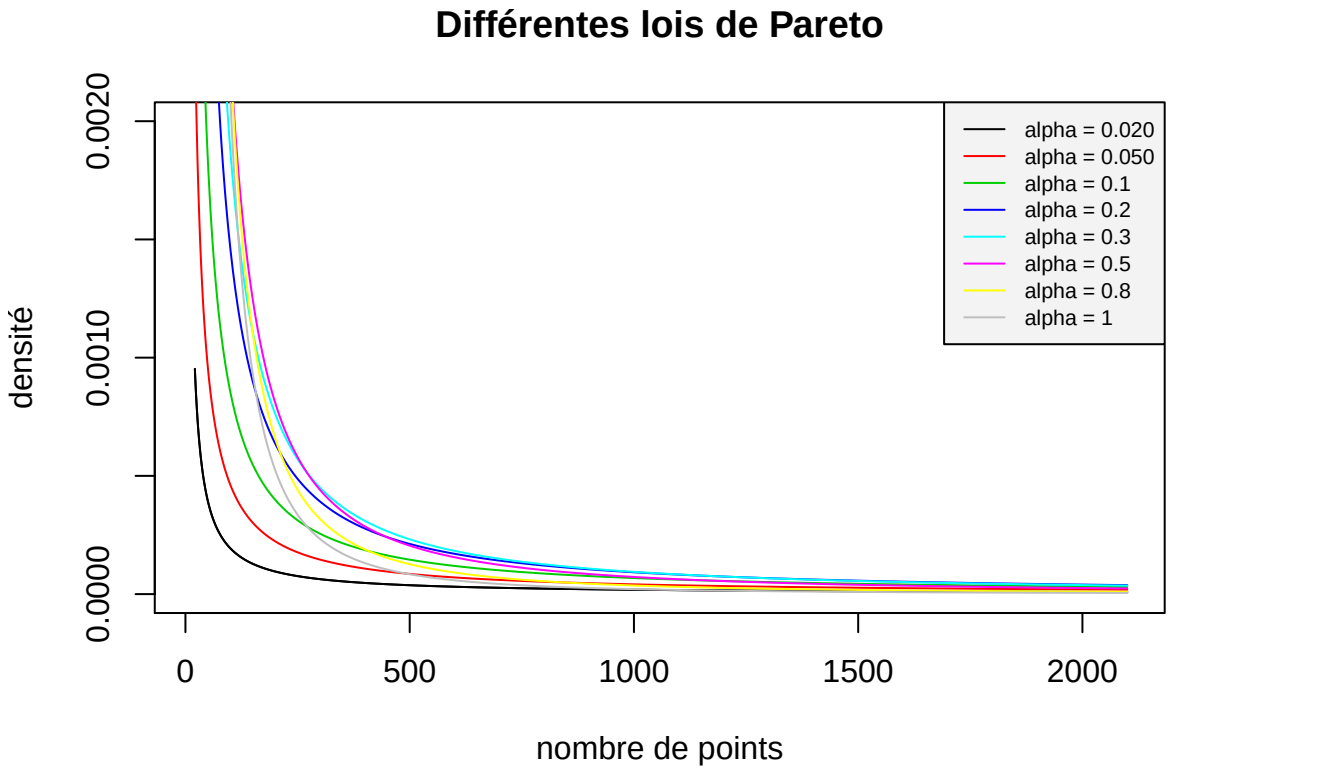
4. Choix du package

J'utilise le package *actuar*. La fonction de densité de probabilité de la loi de Pareto de paramètres α et m est $f_Z(z; m, \alpha) = \alpha \frac{m^\alpha}{z^{\alpha+1}}$.

On choisit ici $m = 21$ (nombre de points minimums de la barre d'admission)

```
library(actuar)
alpha_legend=c("0.020","0.050","0.1","0.2","0.3","0.5","0.8","1")
alpha=as.numeric(alpha_legend)
x = 21:2100
y = dpareto1(x, alpha[1], 21)

plot(x, y,xlim=c(15,2100),ylim=c(0,0.002) ,xlab="nombre de points", ylab="densité",
      main = "Différentes lois de Pareto", type="l")
for (i in 1:8){
  lines(x, dpareto1(x, alpha[i], 21) , col = i)
}
legend("topright", paste("alpha = ",alpha_legend,sep=""), col=1:8, lty=1,cex=0.70,bg="#f3f3f3")
```



5. Choix de la loi à priori de alpha

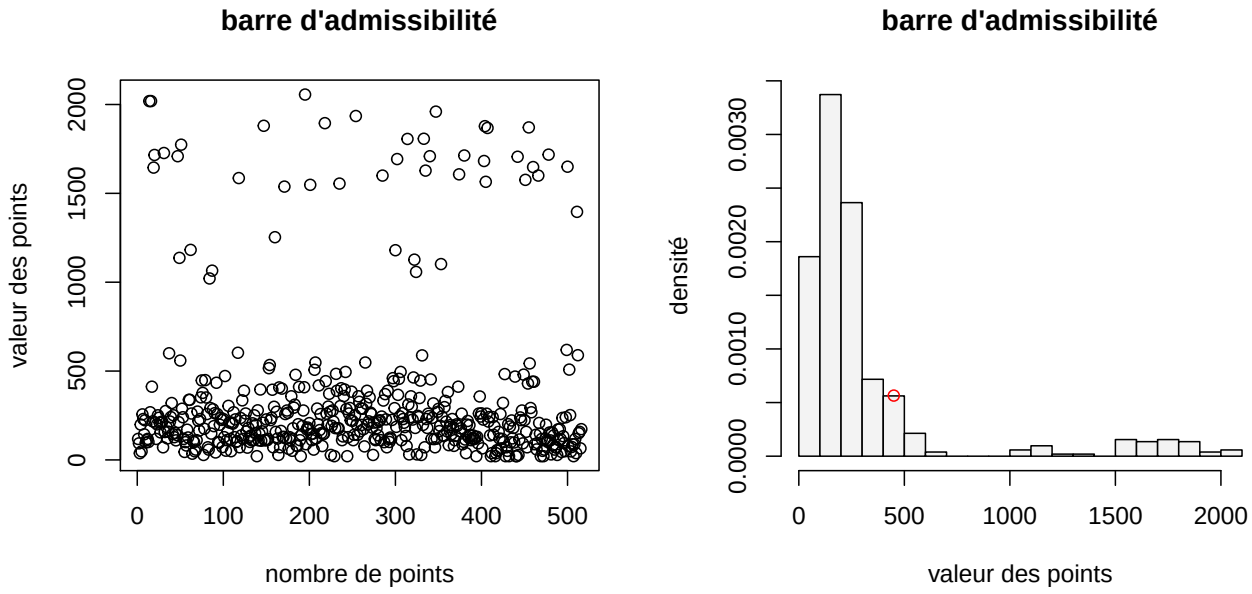
Une 1ère approche

Pour déterminer α, je commence par visualiser les données.

```
par(mfrow=c(1,2))
summary(mut$Barre)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    21.0   111.0   196.0   321.9   292.0  2056.0
```

```
plot(mut$Barre,,xlab="nombre de points", ylab="valeur des points",main = "barre d'admissibilité" )
hist(mut$Barre,freq = FALSE, breaks=20,xlab="valeur des points", ylab="densité",
     main = "barre d'admissibilité",col=("#f3f3f3"))
points(450,0.000565,col="red",lty=1)
```



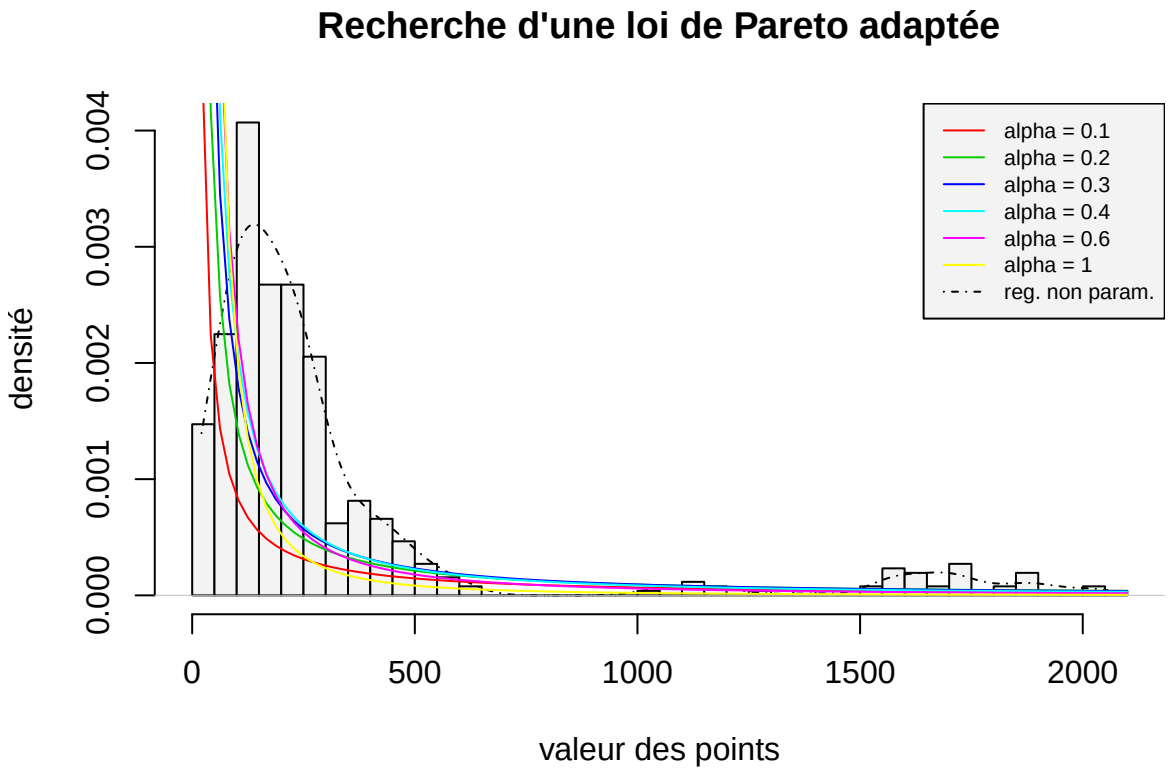
Une majorité des points semble être contenue dans l'intervalle [21 ; 500].
En regardant l'histogramme, on pourrait essayer de déterminer une distribution de Pareto dont la courbe passant par le point marqué en rouge.

Avec une régression non paramétrique

Je cherche une loi à priori de la forme d'une densité de Pareto. L'idée serait de se rapprocher de la courbe obtenue par régression non paramétrique (en pointillés sur le graph. ci-dessous).

```
library("kdensity")
hist(mut$Barre,freq = FALSE, breaks = 30,xlab="valeur des points", ylab="densité",
     main = "Recherche d'une loi de Pareto adaptée",col=("#f3f3f3"))
kde=kdensity(mut$Barre,,1.5,kernel = "gaussian",support = c(21,2056))
lines(kde, xlim=c(195,2100), col=9, lty=4)
val_param_legend=c("0.1","0.2","0.3","0.4","0.6","1")
val_param=as.numeric(val_param_legend)
for (i in 1:6){
```

```
  curve(dpareto1(x, val_param[i],21), col = i+1, lty = 1, xlim=c(21,2100), add = TRUE)
}
legend("topright", c(paste("alpha = ",val_param_legend,sep=""),"reg. non param."), col=c(2:7,1), lty=c(
```



Des valeurs de α trop petites (inférieures à $0,20$) ou trop grandes (supérieures à 1) semblent inadaptées. $\alpha \in [0.20\,;\,0.6]$ (à postériori) semble convenir.

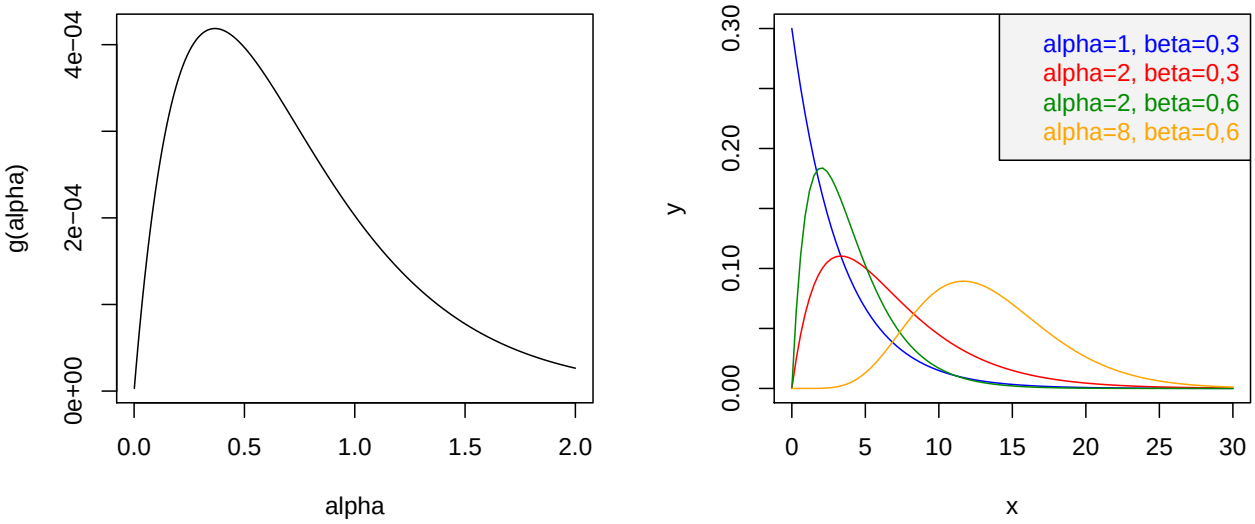
Un essai "heuristique"

Je décide de procéder autrement. Je choisis de "restreindre" la covariable *barre* à sa moyenne et cherche à maximiser la fonction $\alpha \to \dfrac{\alpha m^\alpha}{\bar{x}_{barre}^{\alpha+1}}$ (où $\bar{x}$ est la moyenne des x_i) afin de me faire une idée sur "un choix raisonnable de loi à priori" et sur un ordre de grandeur dans lequel α évolue.

J'obtiens la représentation suivante :

```
par(mfrow=c(1,2))
alph=(1:2000)/1000
g=alph*21^alph/mean(mut$Barre)^(alph+1)
plot(alph,g, type="l",xlab="alpha", ylab="g(alpha)" )
plot( function(x) dgamma(x,shape=1,rate=.3),0 , 30, xlab="x", ylab="y",
main="Fig. 1. diverses densités Gamma.",col="blue");
curve( dgamma(x,shape=2,rate=.3), add=TRUE, col="red");
curve( dgamma(x,shape=2,rate=.5), add=TRUE, col="green4");
curve( dgamma(x,shape=8,rate=.6), add=TRUE, col="orange")
legend( x="topright", y=NULL, text.col= c("blue","red","green4","orange"), legend=
c("alpha=1, beta=0,3","alpha=2, beta=0,3","alpha=2, beta=0,6","alpha=8, beta=0,6"),bg=("#f3f3f3"))
```

Fig. 1. diverses densités Gamma.



En réduisant les données à leur moyenne, la courbe décrivant g ressemble à une loi *gamma* dont le maximum serait obtenu en $\alpha \approx 0.366$. Cela pourrait tendre à confirmer que :

- le choix d'une loi Γ semble indiqué pour une prior de α
- l'idée que la loi à postériori doit accorder des valeurs plus fréquentes pour $\alpha \in [0.20\,;\,0.60]$ à postériori reste cohérente

Sans plus d'informations et pour ne pas biaiser l'exercice (cf. questions suivantes), une "prior" suivant une loi $\Gamma(1,1)$ semble convenable.

6. Détermination de la loi à postériori de alpha

On sait que $\pi(\alpha/X)\ \Theta\ \pi(\alpha) \times L(X_1, ..., X_n/\alpha)$ (où Θ indique une "relation de proportionnalité").

- D'une part, la vraissemblance L s'écrit sous la forme :

$$\forall x_i \geq m,\ L(X_1, ..., X_n/\alpha) = \prod_{i=1}^{i=n} \alpha \frac{m^\alpha}{x_i^{\alpha+1}} = \alpha^n \frac{m^{n\alpha}}{\left(\prod_{i=1}^{i=n} x_i\right)^{\alpha+1}} = \alpha^n \frac{m^{n\alpha}\left(\prod_{i=1}^{i=n} x_i\right)^{-\alpha}}{\left(\prod_{i=1}^{i=n} x_i\right)}$$

- D'autre part, α suit une loi du type $\Gamma(a, b)$, donc $\pi(\alpha)$ peut s'écrire sous la forme :
  $\pi(\alpha) = \frac{b^a}{\Gamma(a)} e^{-b\alpha} \alpha^{a-1} \mathbf{1}_{\{\mathbf{z} \geq \mathbf{m}\}}$.

Dans ce cas, $\pi(\alpha/X)\ \Theta\ \frac{b^a}{\Gamma(a)} e^{-b\alpha} \alpha^{a-1} \alpha^n \frac{m^{n\alpha}\left(\prod_{i=1}^{i=n} x_i\right)^{-\alpha}}{\left(\prod_{i=1}^{i=n} x_i\right)}$, d'où $\pi(\alpha/X)\ \Theta\ e^{-b\alpha} \alpha^{a-1} \alpha^n m^{n\alpha} \left(\prod_{i=1}^{i=n} x_i\right)^{-\alpha}$

soit : $\pi(\alpha/X)\ \Theta\ \alpha^{n+a-1} e^{-\alpha\left(b + \sum_{i=1}^{i=n} ln(x_i) - nln(m)\right)}$.

Ainsi, la loi à postériori est une loi $\Gamma\left(a + n, b + \sum_{i=1}^{i=n} ln(x_i) - nln(m)\right)$.

Les lois sont conjuguées !

Cela s'avère relativement pratique pour tirer un échantillon de la loi à postériori et éviter un MCMC (par exemple).

```
n=length(mut$Barre)
c(1+n,1+sum(log(mut$Barre))-n*log(21))
```

```
## [1]  517.000 1147.141
```

–> Ma loi à postériori pour α est donc la loi $\Gamma(517, 1147.141)$

7. Echantillon et Intervalle de crédibilité

```
# Un échantillon
ech=rgamma(1000,1+n,1+sum(log(mut$Barre))-n*log(21))

# un intervalle de crédibilité associé à cet échantillon
cred.echan=c(sort(ech)[2.5/100*1000],sort(ech)[97.5/100*1000])
cred.echan
```

```
## [1] 0.4123771 0.4915267
```

```
mean(cred.echan)
```

```
## [1] 0.4519519
```

```
# un intervalle de crédibilité associé à ma loi à postériori
cred=qgamma(c(.025, .975), 1+n, 1+sum(log(mut$Barre))-n*log(21))
cred
```

```
## [1] 0.4126690 0.4903532
```

```
mean(cred)
```

```
## [1] 0.4515111
```

Les intervalles sont relativement proches. C'est rassurant...

Et en vrai ? (maximum de vraisemblance)

La vraissemblance des X_i s'écrit $L(m, \alpha, z_1, ..., z_n) = \dfrac{(\alpha m^\alpha)^n}{\prod_{i=1}^{n} (z_i)^{\alpha+1}} \mathbf{1}_{\{\mathbf{z} \geq \mathbf{m}\}}$

Après passage par la log-vraissemblance, annulation de sa dérivée partielle et vérification de la nature de l'extrema, le maximum de vraissemblance est défini par $\hat{\alpha}_{m,v} = \dfrac{n}{\sum_{i=1}^{n} ln(x_i) - ln(m)}$.

```
alpha=length(mut$Barre)/sum(log(mut$Barre)-log(21))
alpha
```

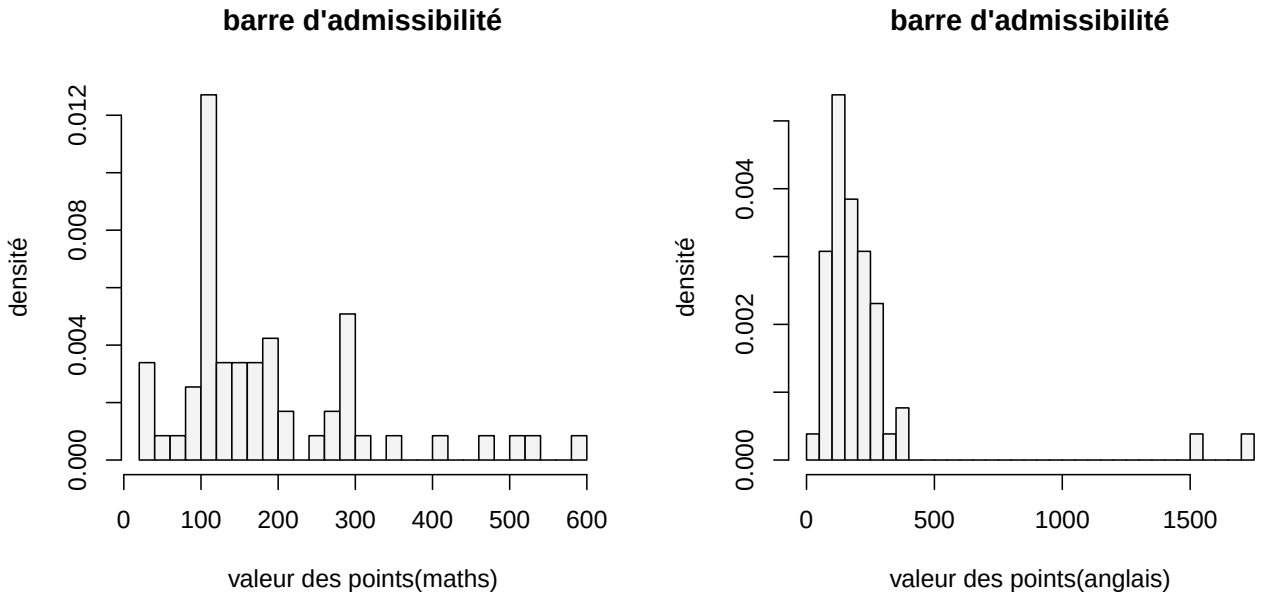
```
## [1] 0.4502063
```

Le maximum de vraissemblance vaut $\hat{\alpha}_{m,v} \approx 0.4502$. Il aurait donc pu être intéressant de choisir des paramètres a, b de la loi à priori en connaissance de ce que l'on devait obtenir au sens fréquentiste pour les paramètres de la loi à postériori, mais cela "corrompt un peu" l'esprit de l'exercice.

Par ailleurs, je constate que malgré un échantillon de taille relativement peu élevé ("516"), le peu d'information récolté à travers l'étude graphique pour déterminer un eloi à priori raisonnable a permis une approximation assez fine de α.

8. “alpha_maths = alpha_anglais” ?

```
par(mfrow=c(1,2))
hist(mut_mat$Barre,freq = FALSE, breaks=30,xlab="valeur des points(maths)", ylab="densité",
     main = "barre d'admissibilité" ,col=("#f3f3f3"))
hist(mut_ang$Barre,freq = FALSE, breaks=30,xlab="valeur des points(anglais)", ylab="densité",
     main = "barre d'admissibilité",col=("#f3f3f3") )
```



En conservant comme prior une loi gamma(1,1) et une loi de Pareto pour les données de chacunes des matières, on obtient une loi à postériori du type gamma de paramètres suivants :

```
n_mat=length(mut_mat2$Barre)
c(1+n_mat,1+sum(log(mut_mat2$Barre))-n_mat*log(21))
```

```
## [1]  60.000 117.933
```

```
# Un échantillon
ech_mat=rgamma(1000,1+n_mat,1+sum(log(mut_mat$Barre))-n_mat*log(21))

# un intervalle de crédibilité associé à cet échantillon
cred.echan_mat=c(sort(ech_mat)[2.5/100*1000],sort(ech_mat)[97.5/100*1000])
cred.echan_mat
```

```
## [1] 0.3848983 0.6493194
```

```
mean(cred.echan_mat)
```

```
## [1] 0.5171089
```

```
sd(cred.echan_mat)
```

```
## [1] 0.1869739
```

```
# un intervalle de crédibilité associé à ma loi à postériori
cred_mat=qgamma(c(.025, .975),1+n_mat,1+sum(log(mut_mat2$Barre))-n_mat*log(21))
cred_mat
```

```
## [1] 0.3882402 0.6453302
```

```
# paramètres
mean(cred_mat)
```

```
## [1] 0.5167852
```

```
sd(cred_mat)
```

```
## [1] 0.1817901
```

```
ech_mat2=rgamma(10000,1+n_mat,1+sum(log(mut_mat$Barre))-n_mat*log(21))
cred.echan_mat2=c(sort(ech_mat2)[2.5/100*10000],sort(ech_mat2)[97.5/100*10000])
cred.echan_mat2
```

```
## [1] 0.3887952 0.6484990
```

```
mean(cred.echan_mat2)
```

```
## [1] 0.5186471
```

Bref, on obtient une valeur de α_{maths} autour de 0.51.

Et pour l'anglais

```
n_ang=length(mut_ang2$Barre)
c(1+n_ang,1+sum(log(mut_ang2$Barre))-n_ang*log(21))
```

```
## [1]  53.0000 108.1779
```

```
# Un échantillon
ech_ang=rgamma(10000,1+n_ang,1+sum(log(mut_ang$Barre))-n_ang*log(21))

# un intervalle de crédibilité associé à cet échantillon
cred.echan_ang=c(sort(ech_ang)[2.5/100*10000],sort(ech_ang)[97.5/100*10000])
cred.echan_ang
```

```
## [1] 0.3659396 0.6308877
```

```
mean(cred.echan_ang)
```

```
## [1] 0.4984136
```

```
sd(cred.echan_ang)
```

```
## [1] 0.1873466
```

```
# un intervalle de crédibilité associé à ma loi à postériori
cred_ang=qgamma(c(.025, .975),1+n_ang,1+sum(log(mut_ang2$Barre))-n_ang*log(21))
cred_ang
```

```
## [1] 0.3669941 0.6303608
```

```
mean(cred_ang)
```

```
## [1] 0.4986774
```

```
sd(cred_ang)
```

```
## [1] 0.1862284
```

```
ech_ang2=rgamma(10000,1+n_ang,1+sum(log(mut_ang$Barre))-n_ang*log(21))
cred.echan_ang2=c(sort(ech_ang2)[2.5/100*10000],sort(ech_ang2)[97.5/100*10000])
cred.echan_ang2
```

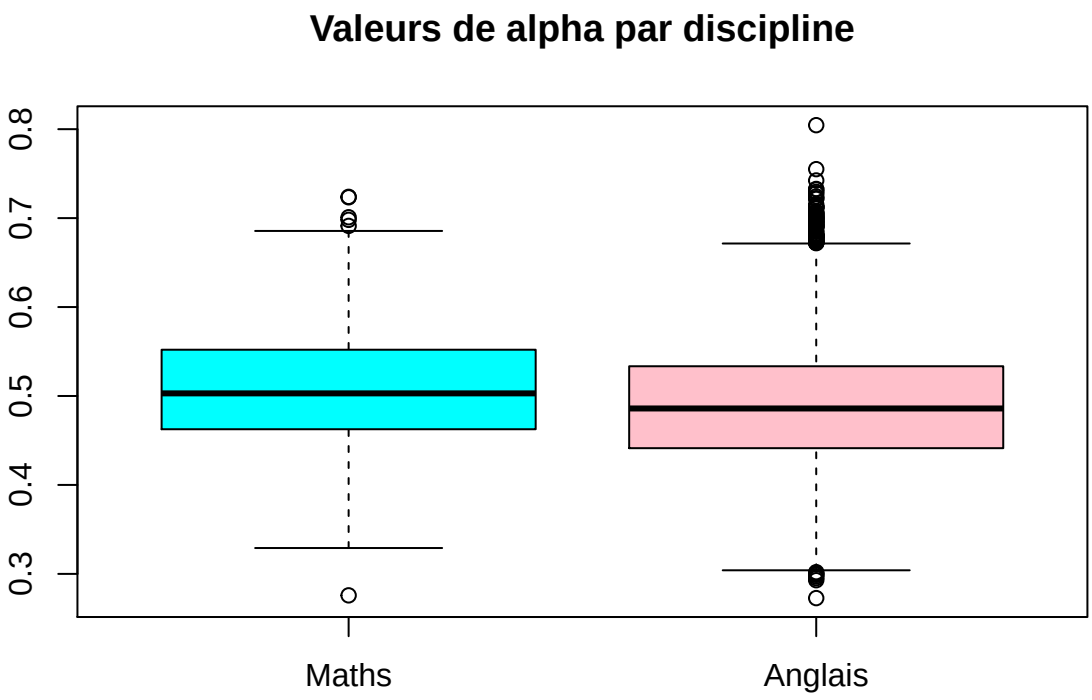
```
## [1] 0.3676379 0.6280480
```

```
mean(cred.echan_ang2)
```

```
## [1] 0.497843
```

On obtient une valeur de $\alpha_{anglais}$ autour de 0.497. L'hypothèse d'égalité des paramètres ne semble pas déraisonnable.

```
boxplot(ech_mat,ech_ang,names=c("Maths","Anglais"),col=c("cyan","pink"),main="Valeurs de alpha par disci
```

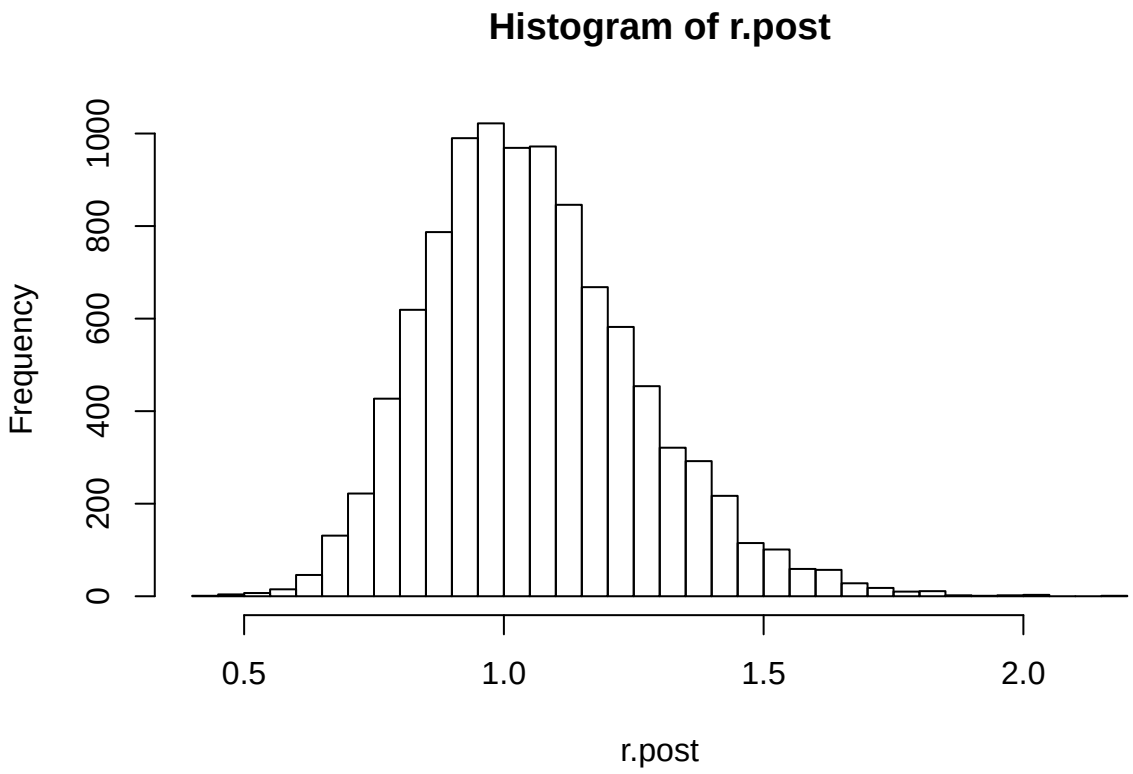


Les échantillons tirés sont très proches. Testons si l'égalité $\alpha_{Maths} = \alpha_{Anglais}$ est plausible en étudiant le rapport $r = \frac{\alpha_{Maths}}{\alpha_{Anglais}}$.

```
par(mfrow=c(1, 1))
niter = 10000
r.post = ech_mat/ ech_ang
summary(r.post)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.4030  0.9129  1.0371  1.0581  1.1812  2.1783
```

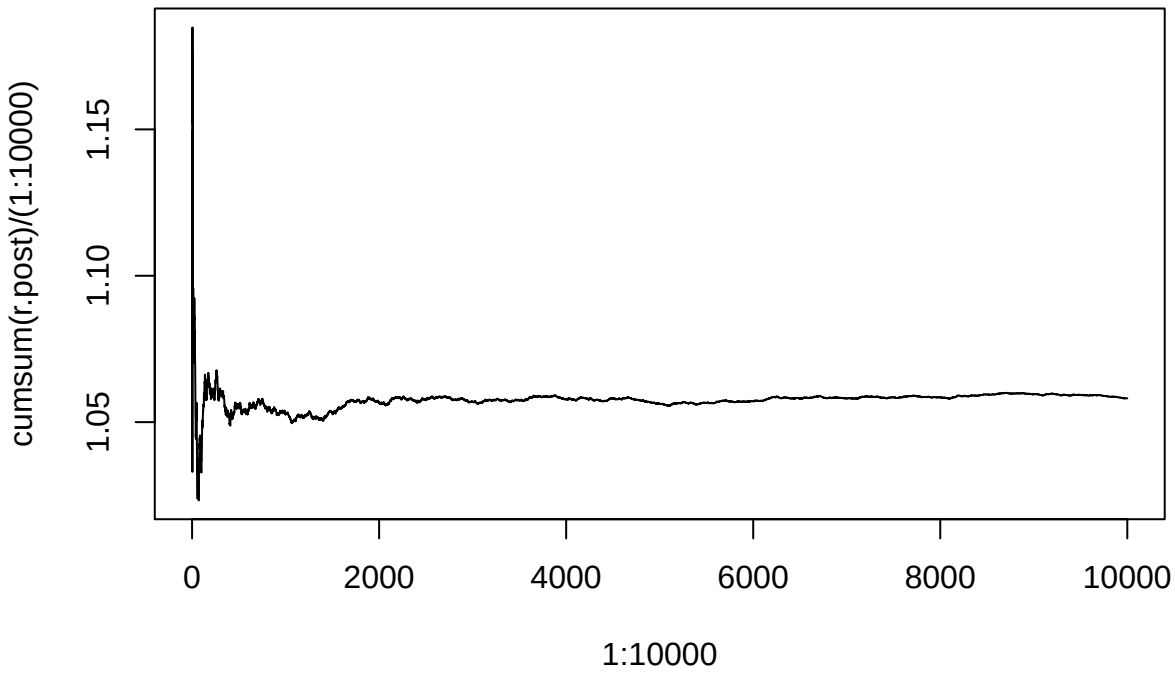
```
hist(r.post, breaks=50)
```




```
quantile(r.post, c(0.025, 0.975))
```

```
##      2.5%     97.5%
## 0.7147608 1.5211400
```

```
plot(1:10000,cumsum(r.post)/(1:10000),type="l")
abline(h=1,col="red")
```



L'hypothèse selon laquelle "les 2 valeurs de α sont les mêmes" est extrêmement plausible au vu de l'histogramme. Le graphique de r ne tend pas précisément vers 1 (il n'y a aucune raison pour ce soit exactement 1), mais l'hypothèse est toit à fait réaliste au vu des analyses.

Je la valide.

Et en vrai ? (maximum de vraisemblance)

La vraissemblance des X_i s'écrit $L(m, \alpha, z_1, ..., z_n) = \frac{(\alpha m^\alpha)^n}{\prod_{i=1}^{n} (z_i)^{\alpha+1}} \mathbf{1}_{\{\mathbf{z} \geq \mathbf{m}\}}$

Après passage par la log-vraissemblance, annulation de sa dérivée partielle et vérification de la nature de l'extrema, le maximum de vraissemblance est défini par $\hat{\alpha}_{m,v} = \frac{n}{\sum_{i=1}^{n} ln(x_i) - ln(m)}$.

```
alpha_mat=length(mut_mat2$Barre)/sum(log(mut_mat2$Barre)-log(21))
alpha_mat
```

```
## [1] 0.5045626
```

```
(cred_mat[1]+cred_mat[2])/2
```

```
## [1] 0.5167852
```

```
alpha_ang=length(mut_ang2$Barre)/sum(log(mut_ang2$Barre)-log(21))
alpha_ang
```

```
## [1] 0.4851748
```

```
(cred_ang[1]+cred_ang[2])/2
```

```
## [1] 0.4986774
```

Les résultats obtenus par maximum de vraissemblance (et donc par l'approche fréquentiste) tendent à confirmer les résultats précédemment obtenus. L'hypothèse $\alpha_{Maths} = \alpha_{Anglais}$ est plausible.