# Data Science

## Compte rendu de TP : *Plans remplissant l'espace et krigeage*

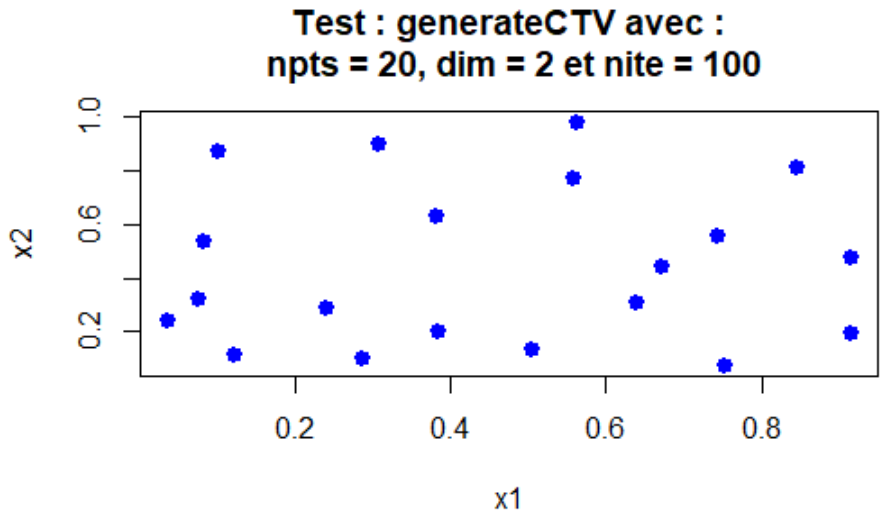
**DEGNI Fidèle**

**RODRIGUES Leticia**

### 1. *Construction de plans*

#### 1.1 Tessellations centroïdales de Voronoï

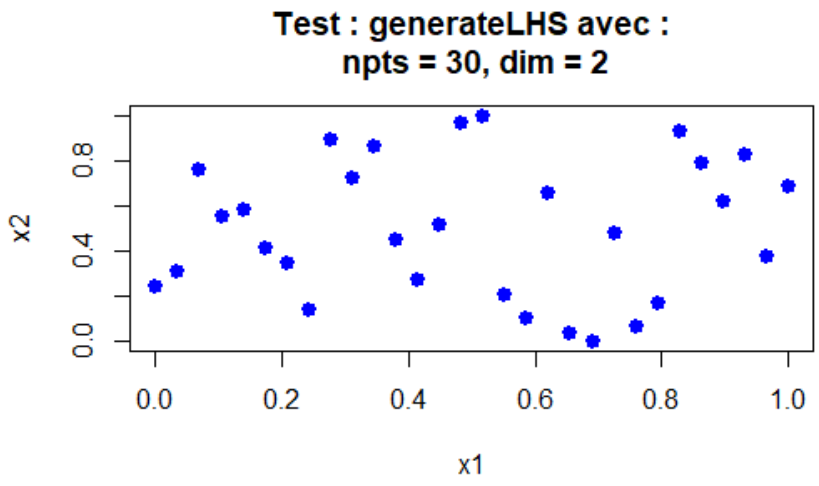
Voir code dans le ficher TP1.R




On obtient bien des points assez bien répartis dans l'espace

#### 1.2 Hypercubes latins

Voir code dans le ficher TP1.R



On vérifie bien qu'on a un point par ligne et par colonne

### 1.3 Critères

Voir code dans le ficher TP1.R

En testant le script sur le plan précédant (hypercubes latins avec $\mathrm{npts} = 30$ et $\dim = 2$), on obtient :

```
> evalMinDist(X)
$minDist
[1] 0.03448276

$allDist
             [,1]       [,2]       [,3]       [,4]      [,5]       [,6]
 [1,] 0.00000000 0.72413793 0.65517241 0.82758621 0.1034483 0.62068966
 [2,] 0.72413793 0.00000000 0.75862069 0.58620690 0.6206897 0.72413793
 [3,] 0.65517241 0.75862069 0.00000000 0.34482759 0.6896552 0.03448276
 [4,] 0.82758621 0.58620690 0.34482759 0.00000000 0.7241379 0.31034483
 [5,] 0.10344828 0.62068966 0.68965517 0.72413793 0.0000000 0.65517241
```

**Optimisation des calculs** : comme la matrice des interdistances est symétrique, le calcul des interdistances est effectuée seulement la partie triangulaires supérieure stricte et est copié dans la partie triangulaire inférieure stricte (voir détail dans le script) ; la diagonale étant nulle.

### 1.4 Hypercubes latins optimisés

Voir code dans le ficher TP1.R

On a codé deux optimisations : **1) recherche aléatoire** où on génère un certain nombre de plans et on garde le meilleur ; **2) un algorithme d'échange avec recuit simulé**.

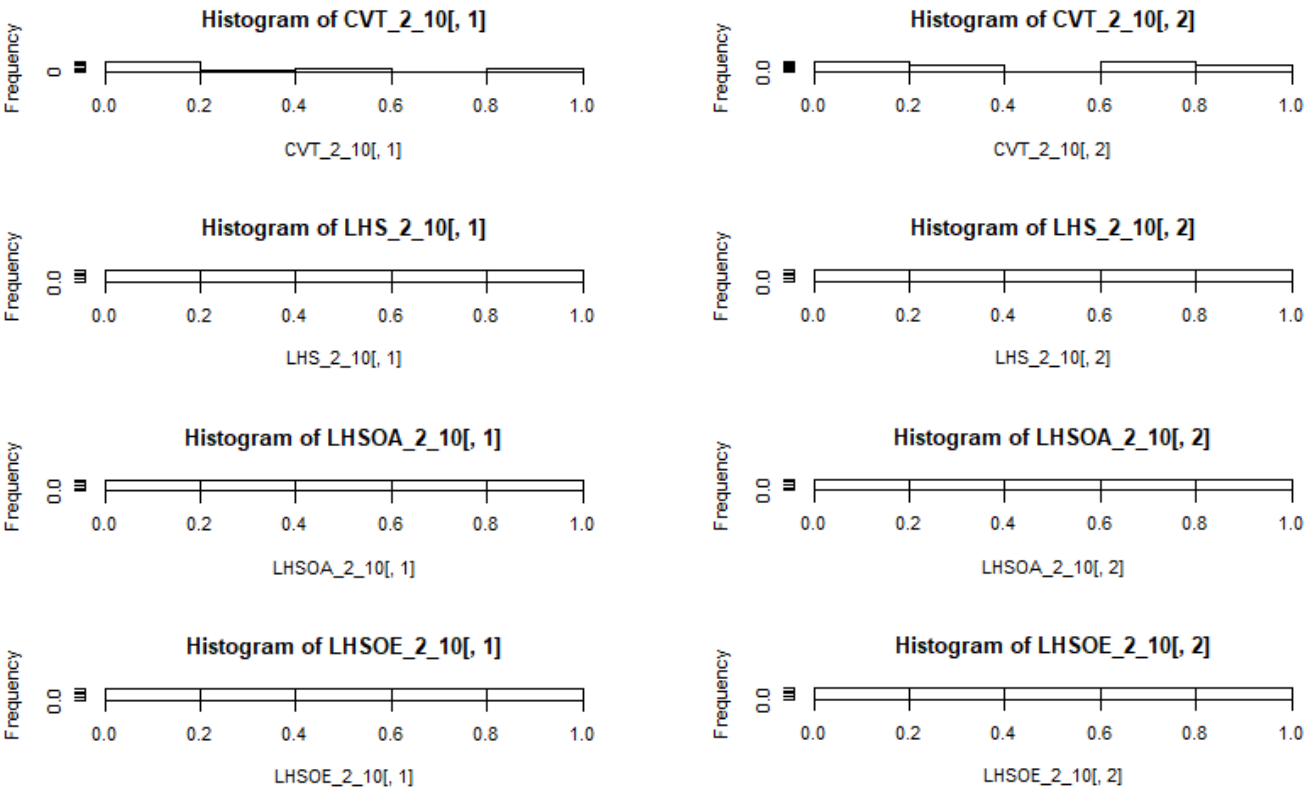
## *2. Analyse*

Voir code dans le ficher TP1.R

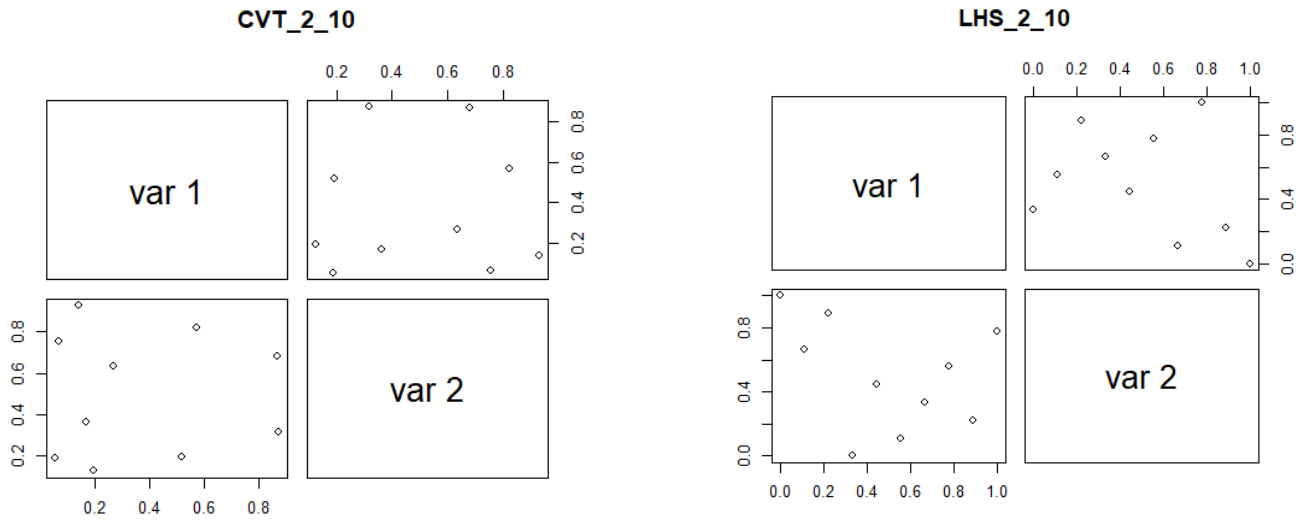
On va comparer différents plans générés avec les différentes méthodes écrites : CTV (*CVT_dim_npts*), LHS (*LHS_dim_npts*) quelconque, LHS optimisé par recherche aléatoire pure (*LHSOA_dim_npts*) et LHS optimisé par recuit simulé (*LHSOE_dim_npts*) avec (dim, npts) prenant les valeurs (2, 10), (5, 70), et (10, 150).

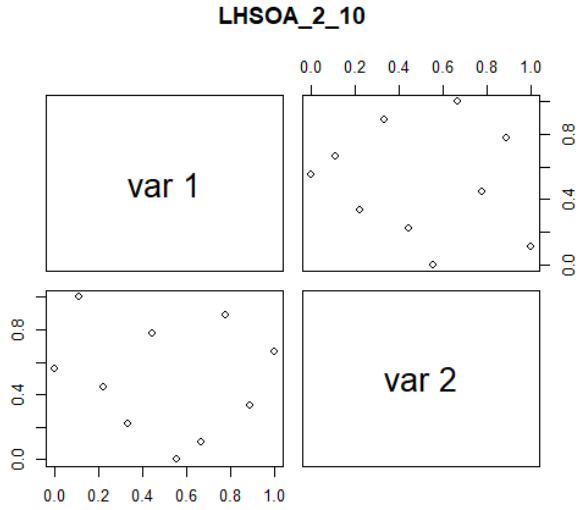
**Pour dim =2 et npts = 10, on observe une répartition uniforme des marginales de dimension 1 pour tous les plans, sauf pour CTV (image suivante).**

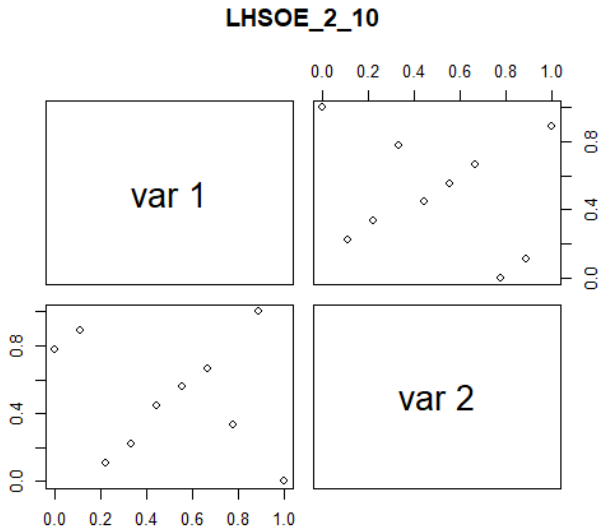
**Répartition sur les marginales de dimension 1 (histogrammes) pour dim = 2 et npts = 10 :**




**Répartition sur les marginales de dimension 2 pour dim = 2 et npts = 10 :**






**Pour dim =2 et npts = 10, on observe une répartition uniforme des marginales de dimension 2 pour tous les plans, sauf pour LHS optimisé avec recuit simulé.**

**Valeurs du critère maximin et de la discrépance ) pour dim = 2 et npts = 10 :**

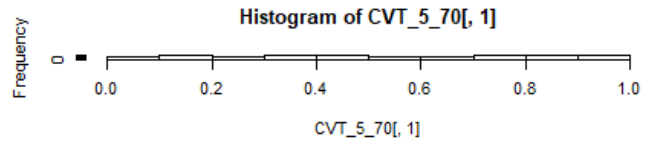
```
> # Valeurs du critere maximin
> evalMinDist(CVT_2_10)$minDist
[1] 0.1390853
> evalMinDist(LHS_2_10)$minDist
[1] 0.2222222
> evalMinDist(LHSOA_2_10)$minDist
[1] 0.1111111
> evalMinDist(LHSOE_2_10)$minDist
[1] 0.1111111
```

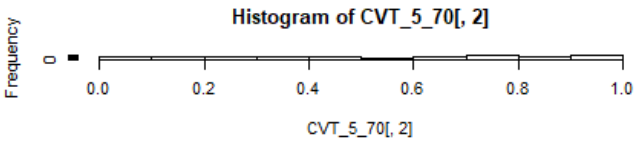
```
> # Valeurs de la discrepance
> discrepancy(CVT_2_10)
[1] 0.1057016
> discrepancy(LHS_2_10)
[1] 0.07303802
> discrepancy(LHSOA_2_10)
[1] 0.05887219
> discrepancy(LHSOE_2_10)
[1] 0.0544051
```

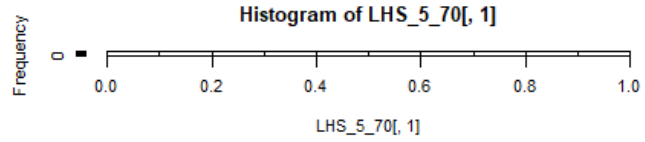
Ici, le critère maximin semble être meilleur pour les plans générés avec LHS optimisé par recherche aléatoire pure et LHS optimisé par recuit simulé. Mais en recommençant les tests, on se rend compte que les autres méthodes peuvent aussi donner de meilleures valeurs pour ce critères-là. On ne peut pas vraiment choisir la meilleure méthode juste avec ce critère. On a la même analyse pour le critère de discrépance.

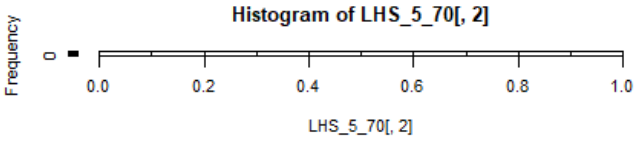
**Pour dim =5 et npts = 70, on observe une répartition uniforme des marginales de dimension 1 pour tous les plans, sauf pour CTV (image suivante).**

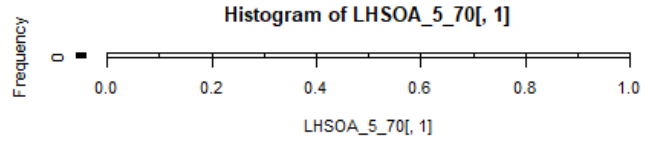
**Répartition sur les marginales de dimension 1 (histogrammes) pour dim = 5 et npts = 70 :**

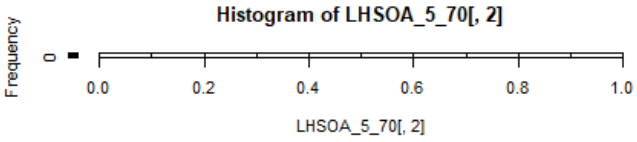


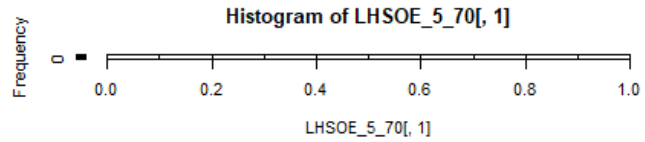



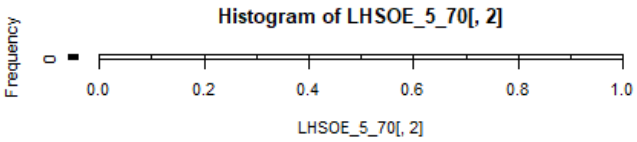



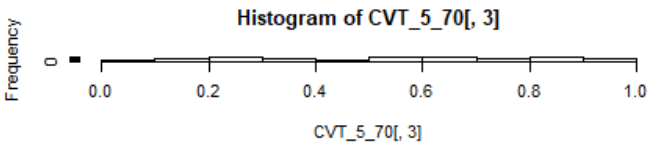



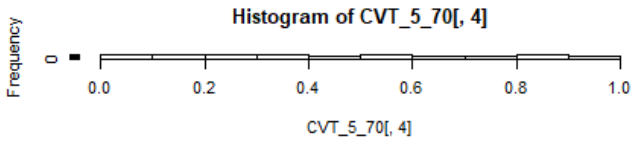



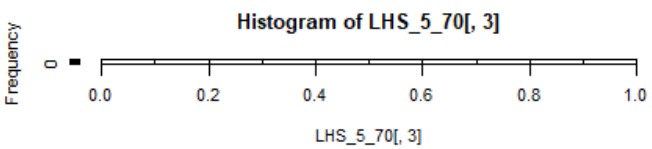



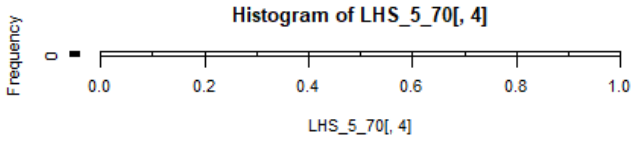



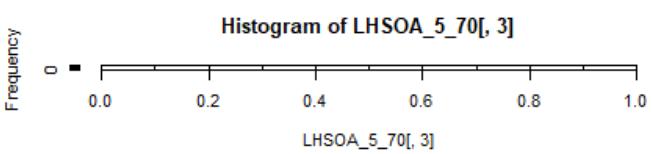



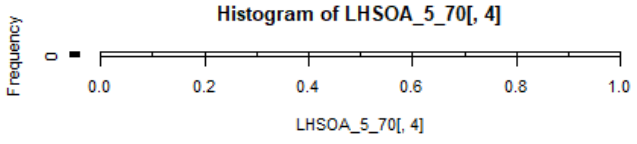



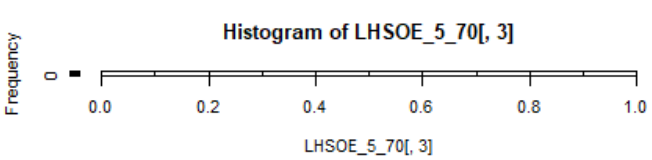



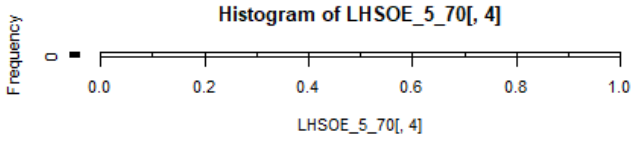



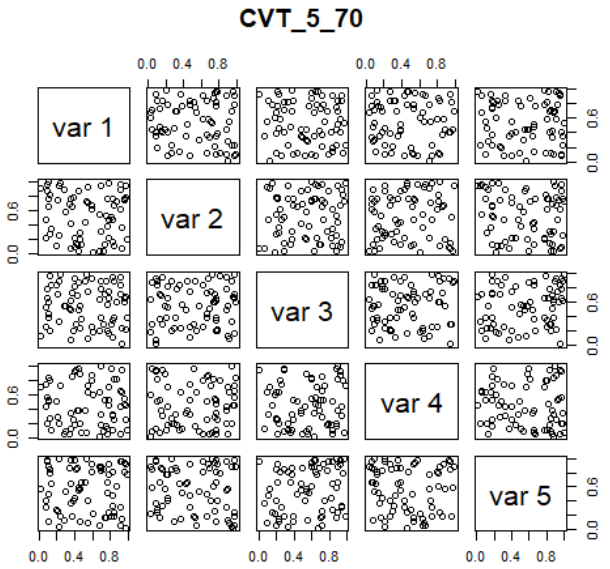



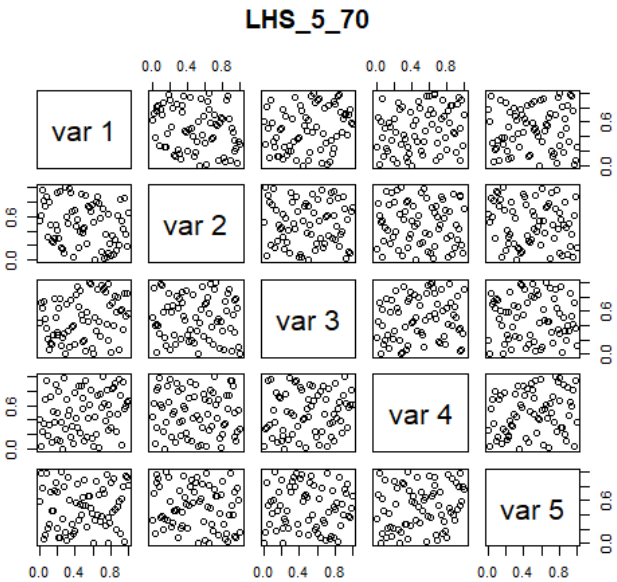



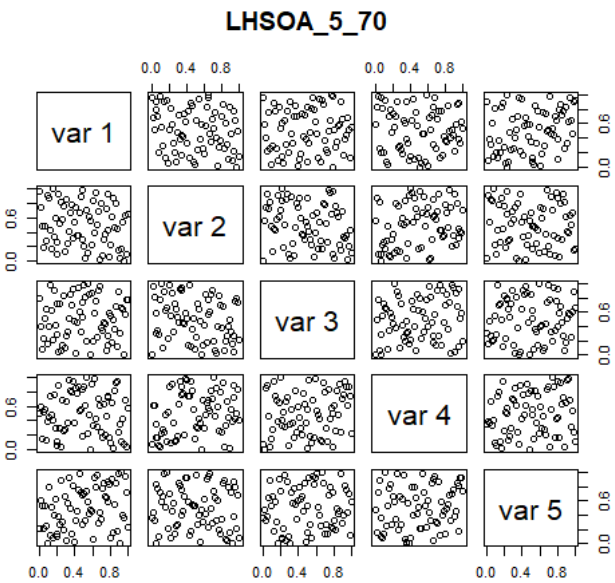



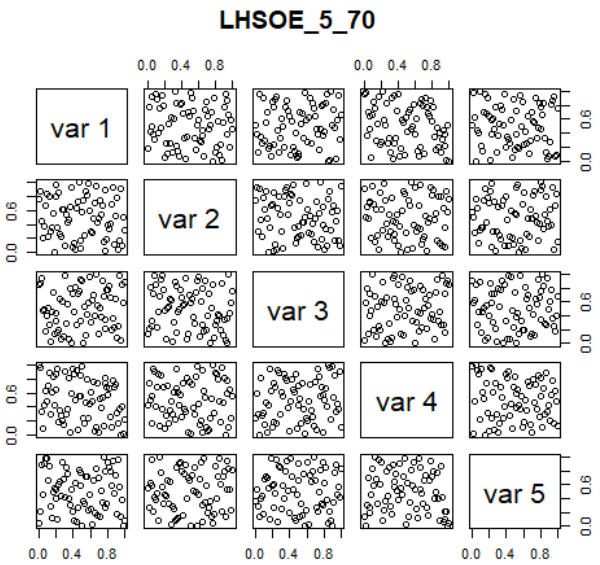



**Répartition sur les marginales de dimension 2 pour dim = 5 et npts = 70 :**



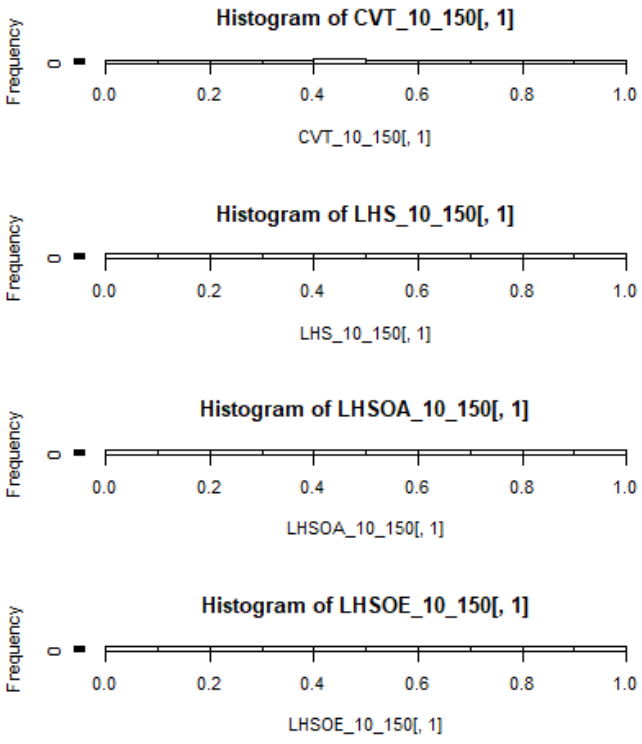

**Pour dim =5 et npts = 70, on observe une répartition uniforme des marginales de dimension 2 pour tous les plans. Néanmoins cela ne nous donne pas la répartition des points dans l'espace de dimension 5.**

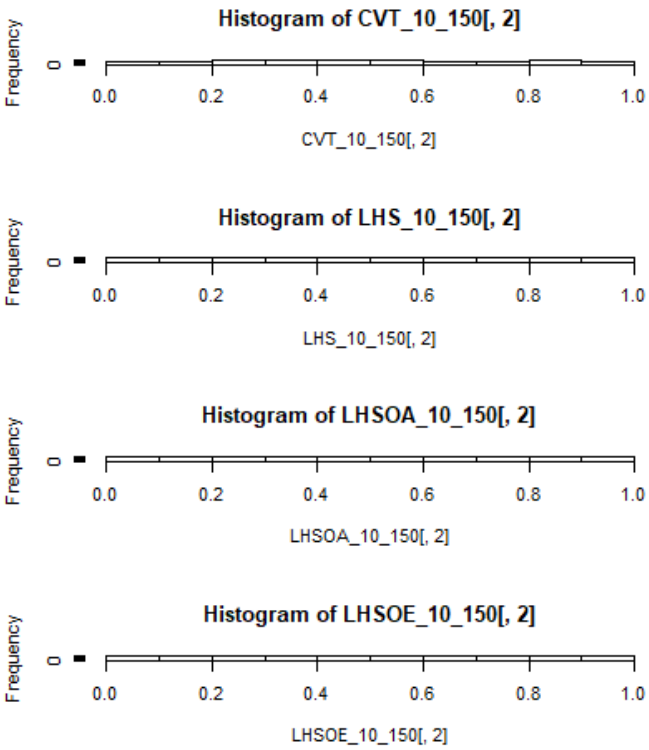
**Valeurs du critère maximin et de la discrépance ) pour dim = 5 et npts = 70 :**

```
> # Valeurs du critere maximin
> evalMinDist(CVT_5_70)$minDist
[1] 0.131693
> evalMinDist(LHS_5_70)$minDist
[1] 0.08695652
> evalMinDist(LHSOA_5_70)$minDist
[1] 0.05797101
> evalMinDist(LHSOE_5_70)$minDist
[1] 0.08695652
```

```
> # Valeurs de la discrepance
> discrepancy(CVT_5_70)
[1] 0.01614755
> discrepancy(LHS_5_70)
[1] 0.01808812
> discrepancy(LHSOA_5_70)
[1] 0.01479417
> discrepancy(LHSOE_5_70)
[1] 0.01357115
```

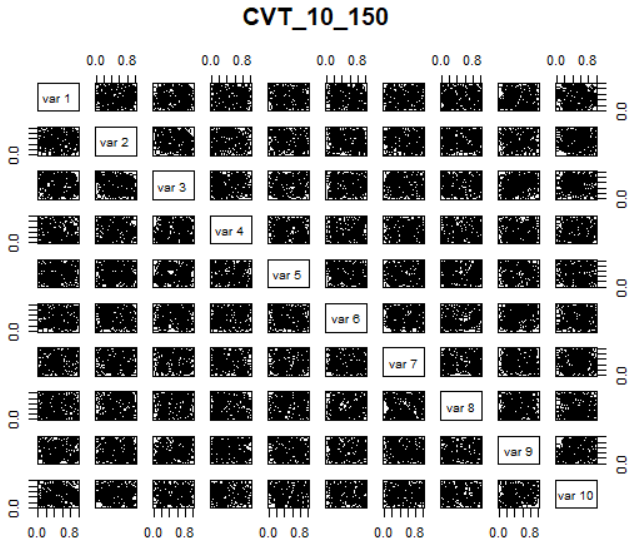
**Répartition sur les marginales de dimension 1 (histogrammes) pour dim = 10 et npts = 150 :**

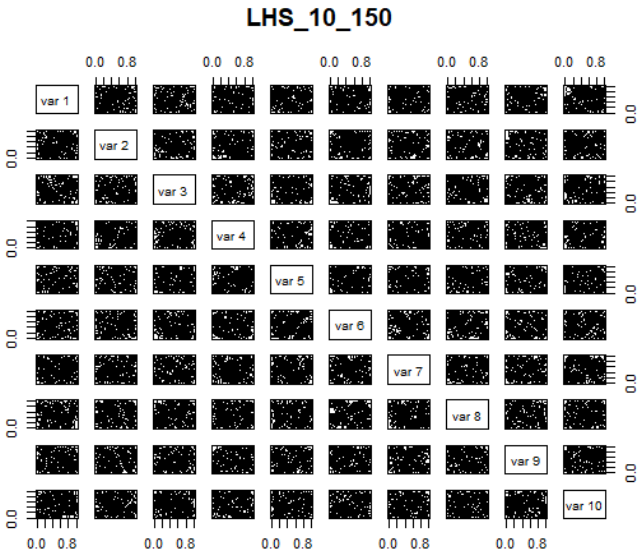


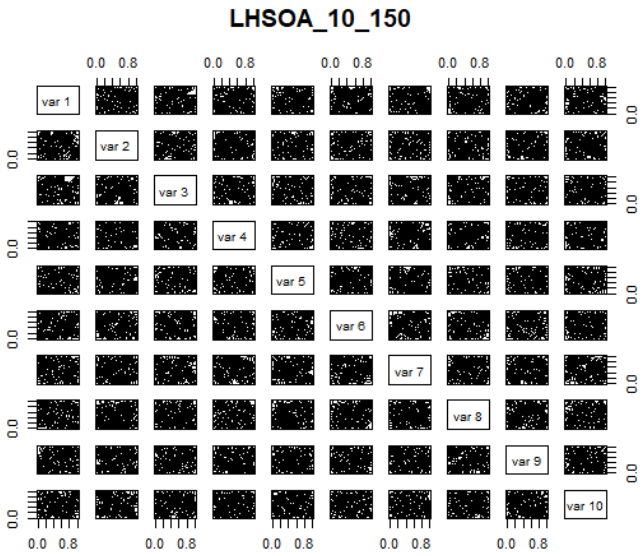



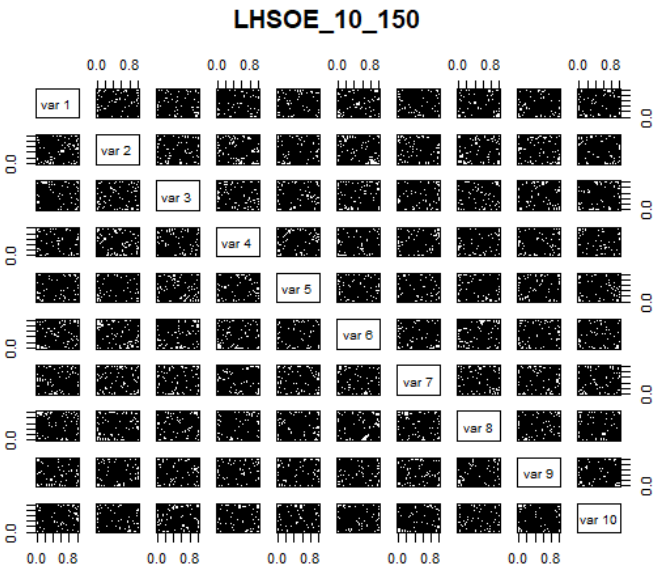

**Pour dim = 10 et npts = 150, on observe une répartition uniforme des marginales de dimension 1 pour tous les plans, sauf pour CTV**

**Répartition sur les marginales de dimension 2 pour dim = 10 et npts = 150 :**






**Pour dim = 10 et npts = 150, on observe une répartition uniforme des marginales de dimension 2 pour tous les plans. Néanmoins cela ne nous donne pas la répartition des points dans l'espace de dimension 10.**

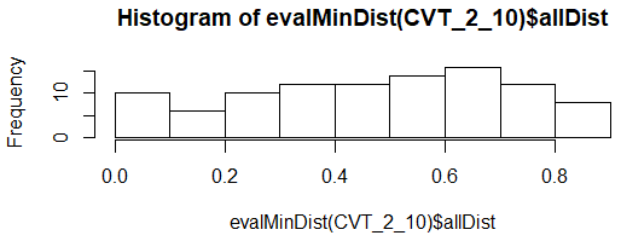
**Valeurs du critère maximin et de la discrépance ) pour dim = 10 et npts = 150 :**

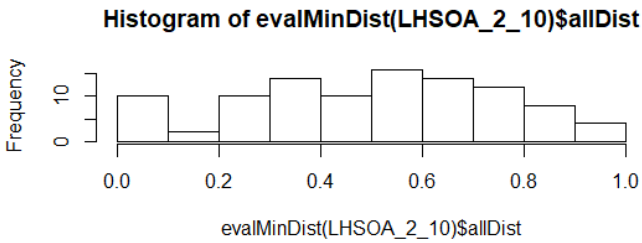
```
> # Valeurs du critere maximin
> evalMinDist(CVT_10_150)$minDist
[1] 0.223862
> evalMinDist(LHS_10_150)$minDist
[1] 0.2550336
> evalMinDist(LHSOA_10_150)$minDist
[1] 0.1610738
> evalMinDist(LHSOE_10_150)$minDist
[1] 0.1946309
```

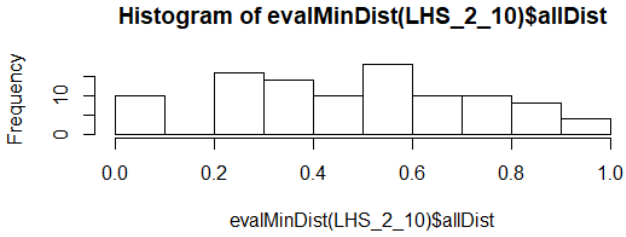
```
> # Valeurs de la discrepance
> discrepancy(CVT_10_150)
[1] 0.002068993
> discrepancy(LHS_10_150)
[1] 0.002283764
> discrepancy(LHSOA_10_150)
[1] 0.002525051
> discrepancy(LHSOE_10_150)
[1] 0.001983957
```

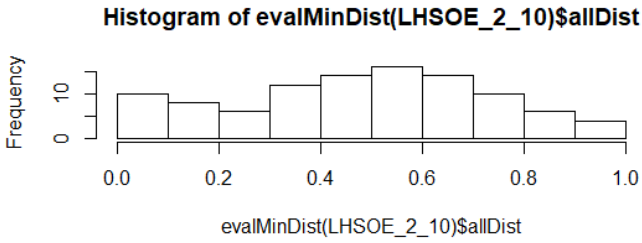
Pour les dimensions 5 et 10, le critère maximin semble être meilleur pour les plans générés avec LHS optimisé par recherche aléatoire pure et LHS optimisé par recuit simulé. En recommençant les tests, on se rend compte que les autres méthodes peuvent aussi donner de meilleures valeurs pour ce critères-là, mais moins souvent que ce qui est observé en dimension 2. On pourrait choisir la meilleure méthode avec ce critère mais avec un peu de prudence (en observant d'autre critères). On a la même analyse pour le critère de discrépance.

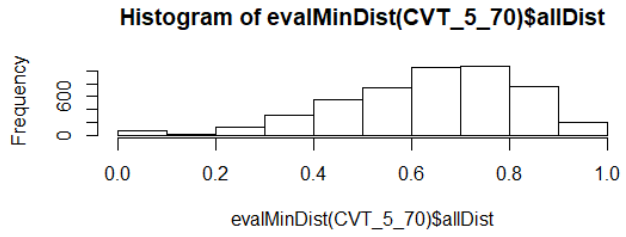
**Histogrammes des interdistances :**

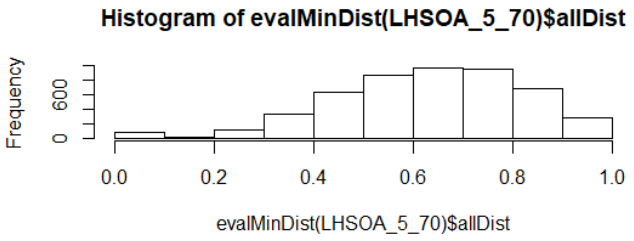


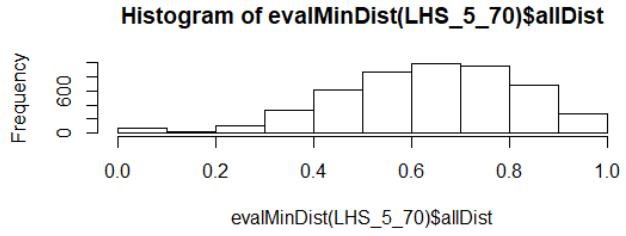



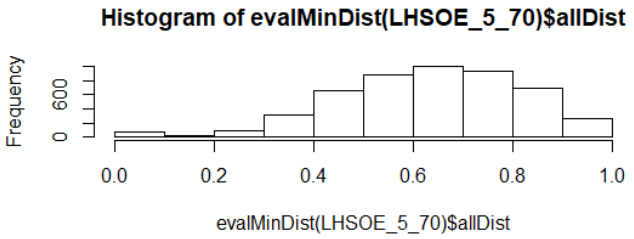



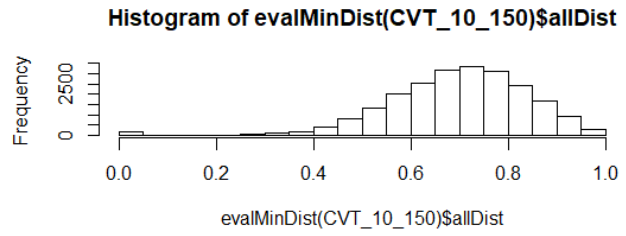



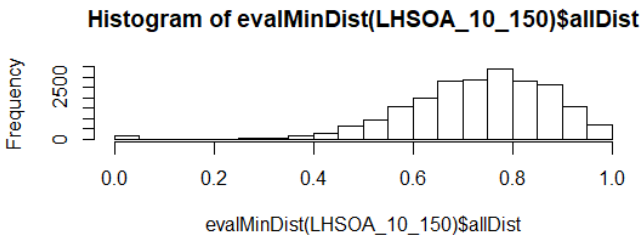



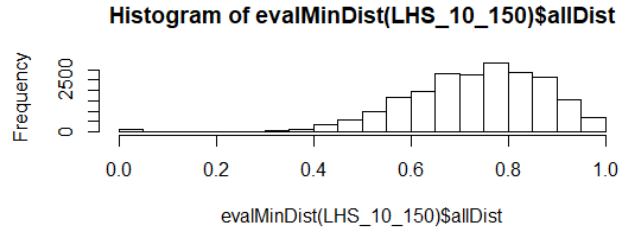



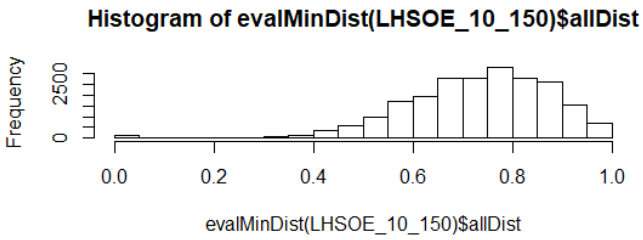



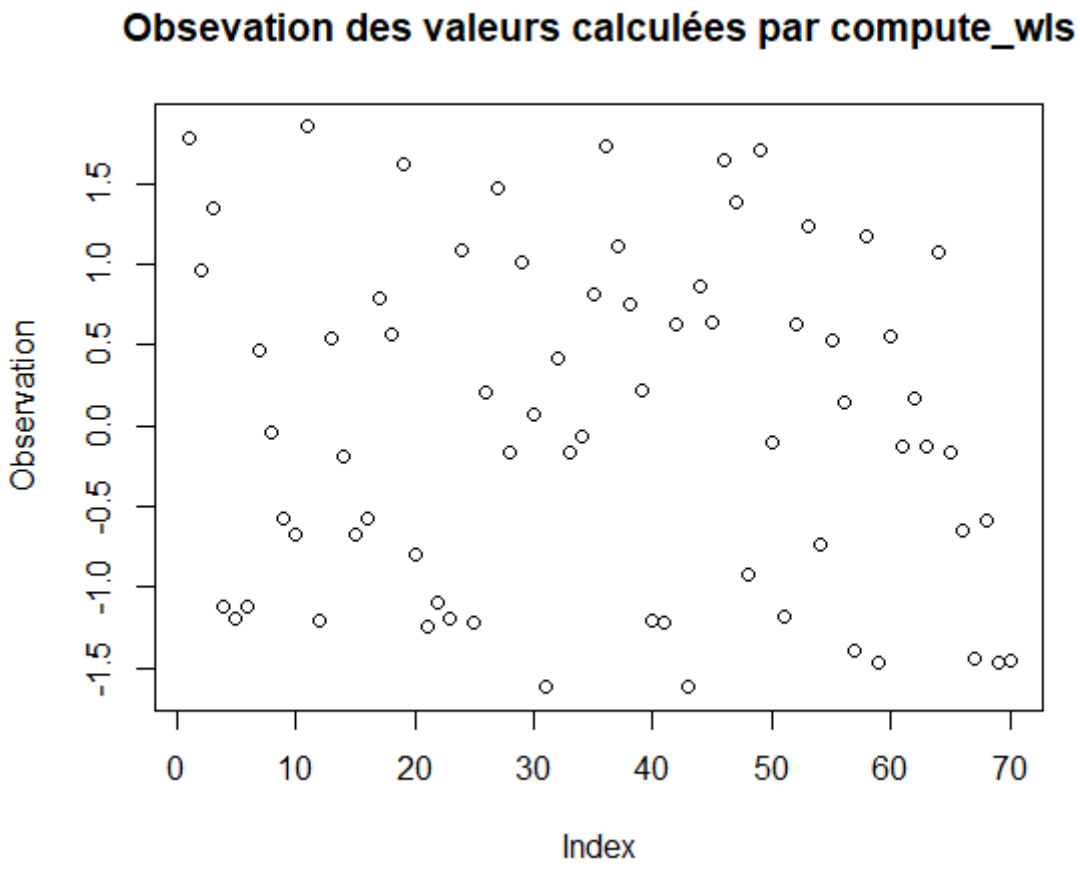



A dimension fixée, la répartition des interdistances est similaires pour les différentes méthodes. On aura donc du mal à choisir une méthode au détriment d'une autre avec ce critère. On peut tout de même observer que les grandes interdistances sont les plus représentées lorsque la dimension augmente. Cela est logique car les points sont de plus en plus éloignés les uns des autres en grandes dimensions.

**Les analyses ci-dessus nous amènent à conclure qu'il n'est chose aisée de choisir les meilleurs plans. En effet, les « meilleurs » plans dépendent grandement des critères de performances considérés. Dans certains cas, les critères ne permettent même pas de discriminer les différentes méthodes.**

## 3. *Prise en main du cas test*

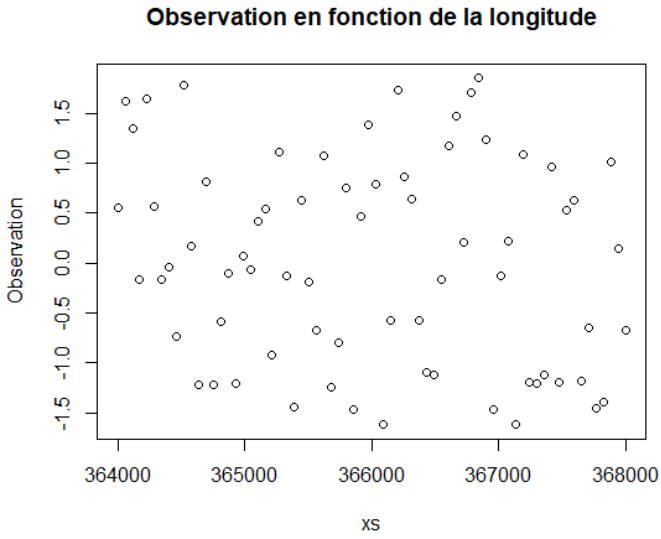
On génère un plan d'expériences simple avec les hypercubes latins et on génère les observations avec *compute_wls*. Voici un *plot* pour 70 observations (à partir d'un plan d'expériences à 70 points) :

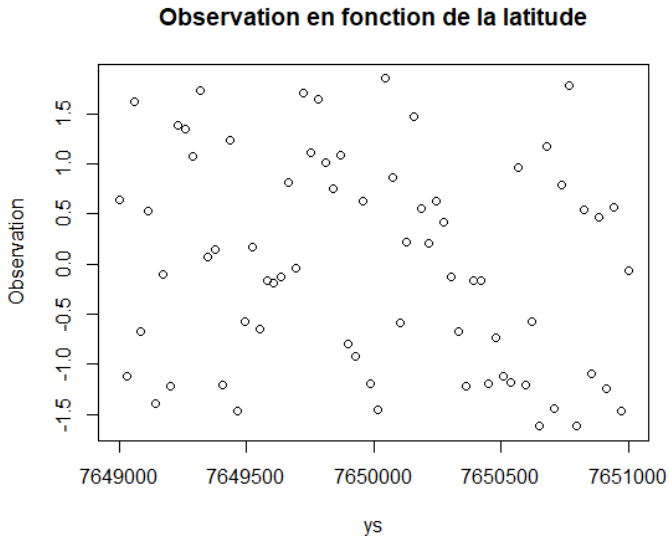


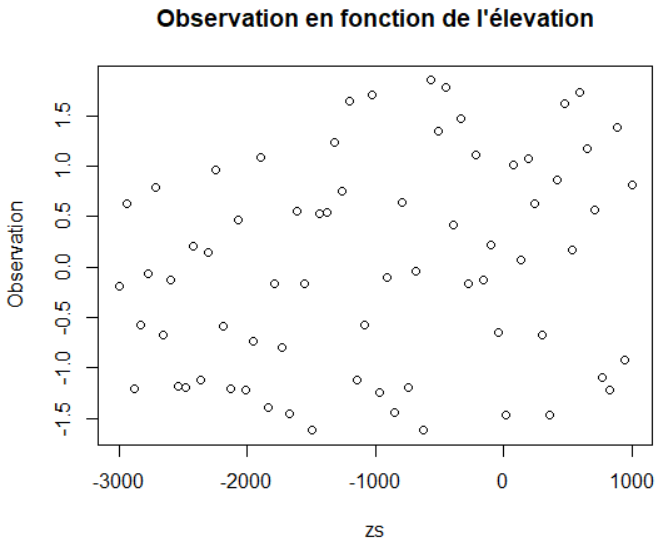

## 4. *Modèle de krigeage*

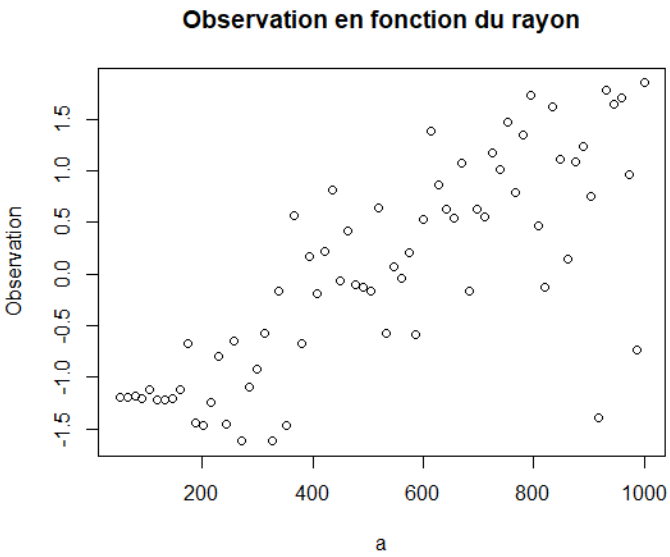
### 4.1 Réflexions préliminaires

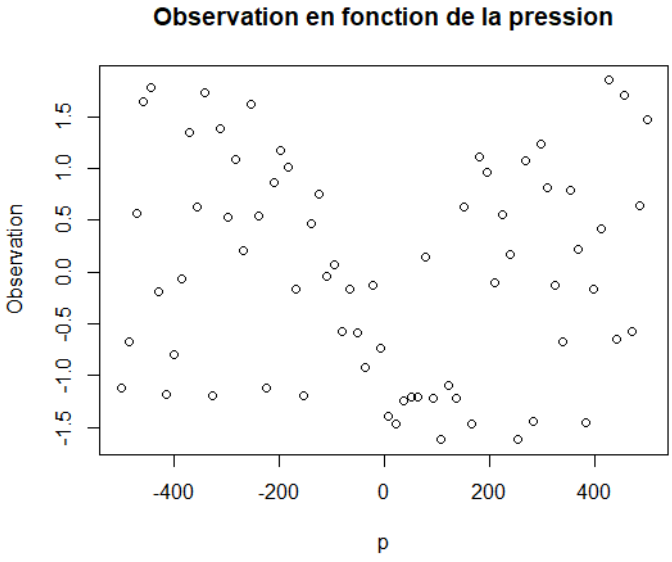
On a des données en dimension 5 ; il est impossible de les visualiser directement. Mais on peut regarder les observations en fonction des différentes variables, ce qui ferait ressortir par exemple des tendances. Tout d'abord, nous allons dénormaliser les colonnes de notre plan d'expériences générés puis faire la visualisation :


Observation en fonction de la longitude
Observation
xs
364000
365000
366000
367000
368000
-1.5
-1.0
-0.5
0.0
0.5
1.0
1.5


Observation en fonction de la latitude
Observation
ys
7649000
7649500
7650000
7650500
7651000
-1.5
-1.0
-0.5
0.0
0.5
1.0
1.5


Observation en fonction de l'élevation
Observation
zs
-3000
-2000
-1000
0
1000
-1.5
-1.0
-0.5
0.0
0.5
1.0
1.5


Observation en fonction du rayon
Observation
a
200
400
600
800
1000
-1.5
-1.0
-0.5
0.0
0.5
1.0
1.5

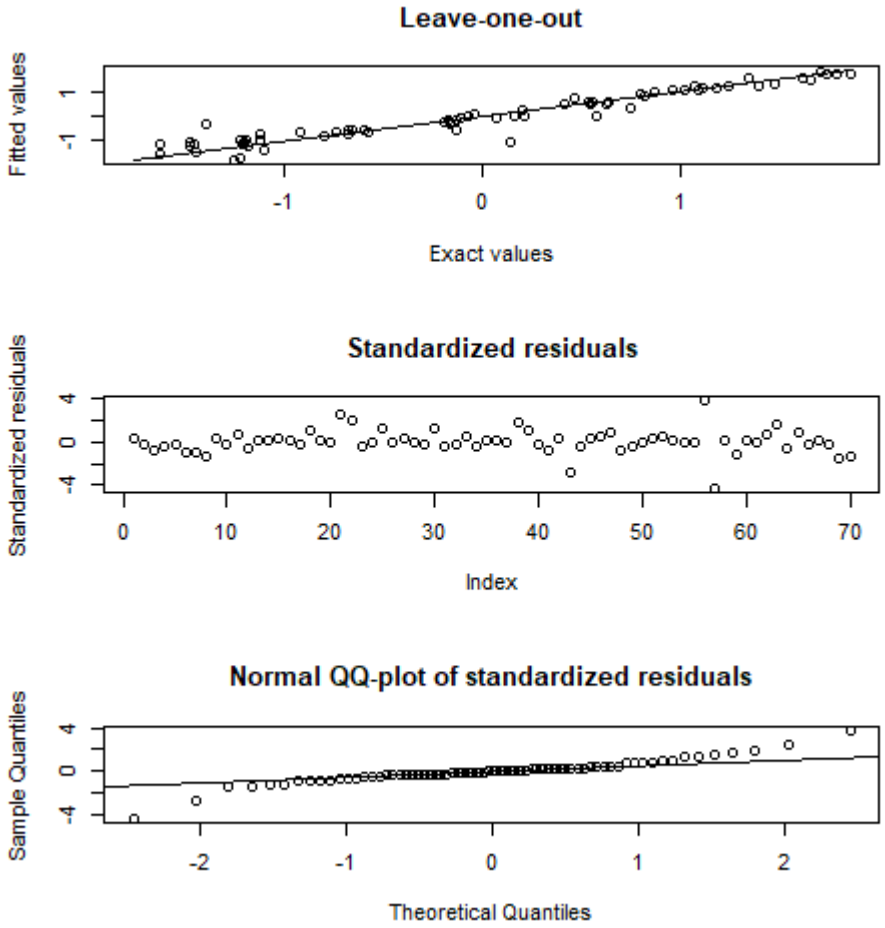

Observation en fonction de la pression
Observation
p
-400
-200
0
200
400
-1.5
-1.0
-0.5
0.0
0.5
1.0
1.5

On observe une tendance claire par rapport au rayon : la réponse augmente lorsque le rayon de la source augmente (tendance linéaire). On peut aussi remarquer une diminution de la réponse avec la pression jusqu'à un minimum puis une augmentation (tendance quadratique).

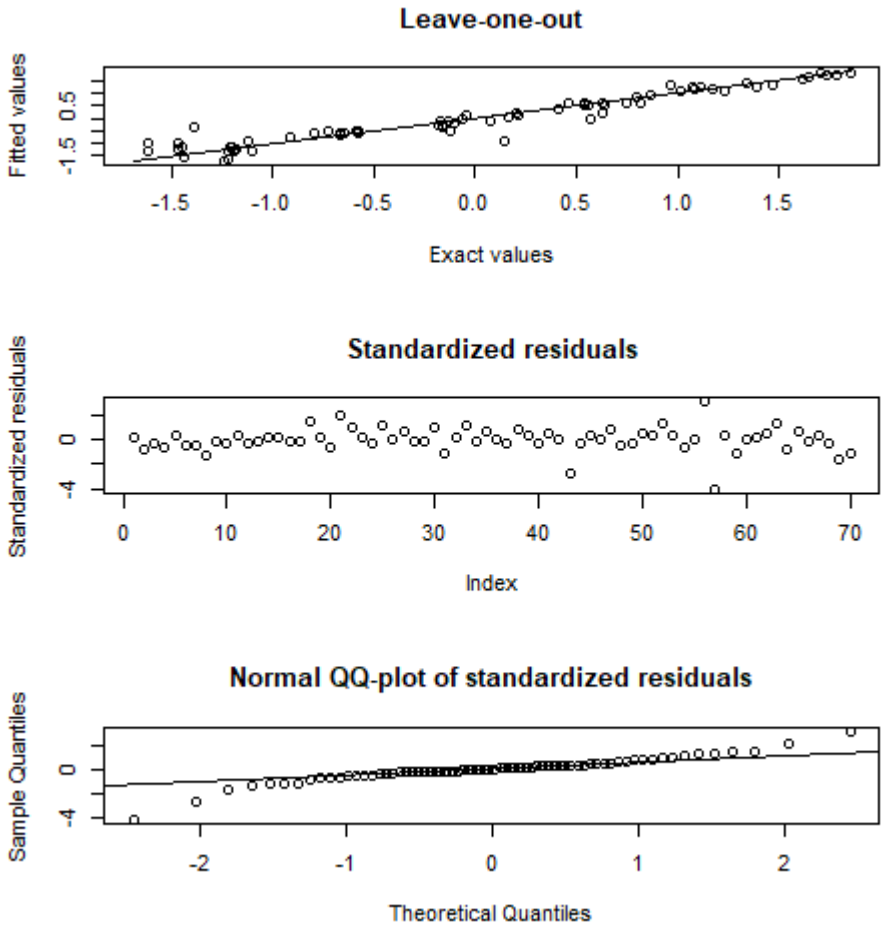
## 4.2 Construction des modèles

On construit différents modèles qu'on va comparer :

Modèle 1 : tendance constante



Modèle 2 : tendance linéaire en a (rayon)




Les quatre autres modèles peuvent être « plottés » dans le fichier mainScript_DatascienceClass.R.

Détail des six modèles construits : modèle 1 avec une tendance constante, noyau matern5_2 ; modèle 2 avec une tendance linéaire en a (le rayon), noyau matern5_2 ; modèle 3 avec une tendance linéaire en a et une tendance quadratique en p (la pression), noyau matern5_2 ; modèle 4 avec une tendance linéaire en a et une tendance quadratique en p et suppression de la constante, noyau matern5_2 ; modèle 5 avec une tendance linéaire en a et une tendance quadratique en p, noyau gaussien ; modèle 6 avec une tendance linéaire en a et une tendance quadratique en p, noyau exponentiel.

En faisant le plot de tous ces modèles et en observant les résidus (qui doivent avoir des valeurs entre -2 et 2 pour minimiser la variance de krigeage), les *leave-one-out* (qui doivent être le plus proches possible de la droite d'équation $y = x$) et les QQ-plot, c'est le modèle 3 qui semble le mieux adapté. **Mais les modèles 3 et 2 sont très proches en termes de qualité et on va préférer le modèle 2 au modèle 3 car il utilise moins de paramètres**.

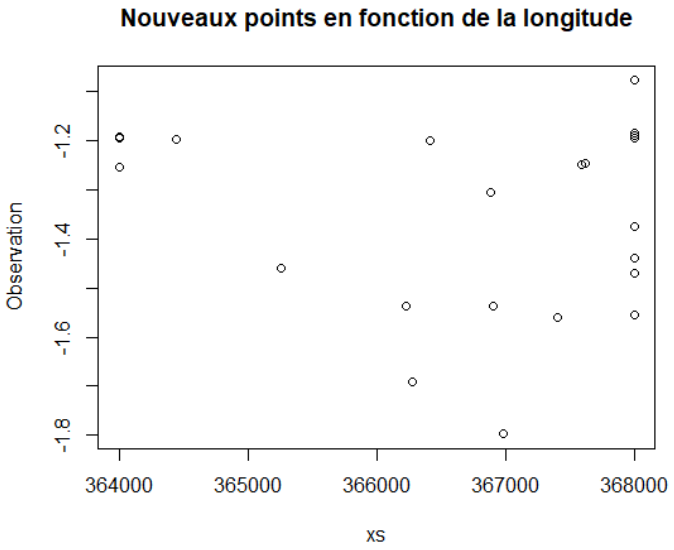
Le modèle retenu est donc le modèle 2 avec une tendance linéaire en a (le rayon), noyau matern5_2.

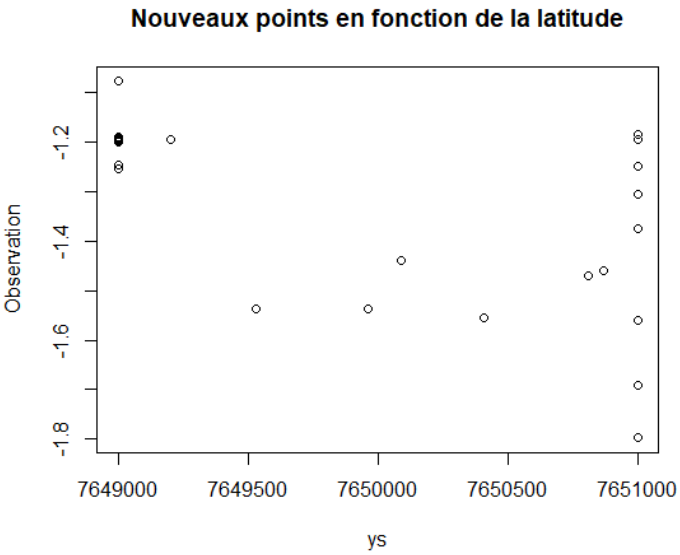
## 4.3 Identification de la source du volcan

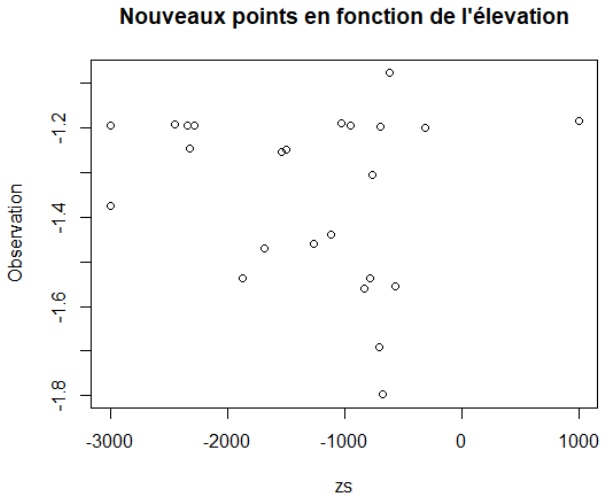
Ici c'est le modèle 1 qui va être utilisé car R nous renvoie une erreur lorsque nous donnons le modèle 2 à la fonction EGO.nsteps du package *rgenoud* ; erreur que nous n'avons pas pu résoudre.

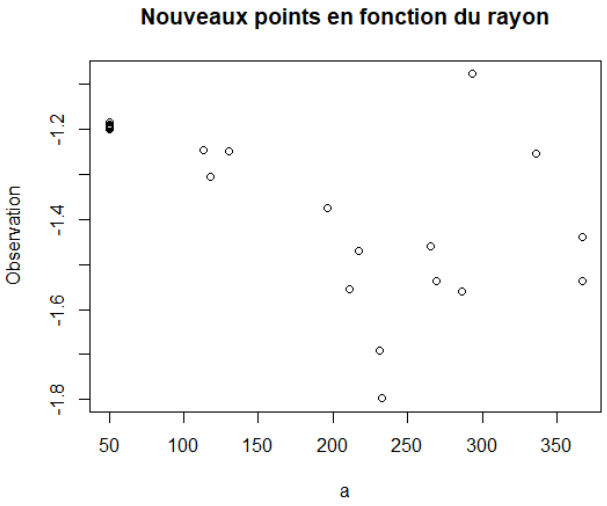
```
Maximization Problem.
Error in y.predict.trend + y.predict.complement : non-conformable arrays
>
```

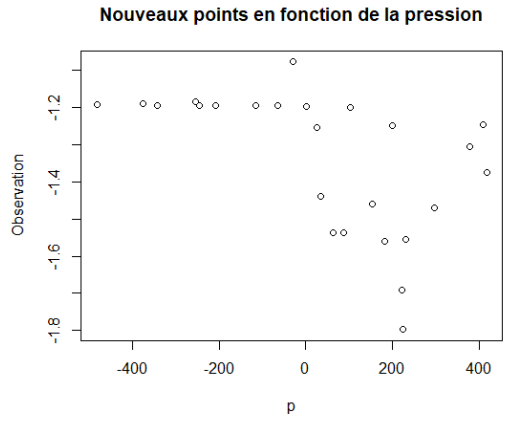
On calcule 25 nouveaux points pour l'optimisation du plan d'expérience avec EGO.nsteps



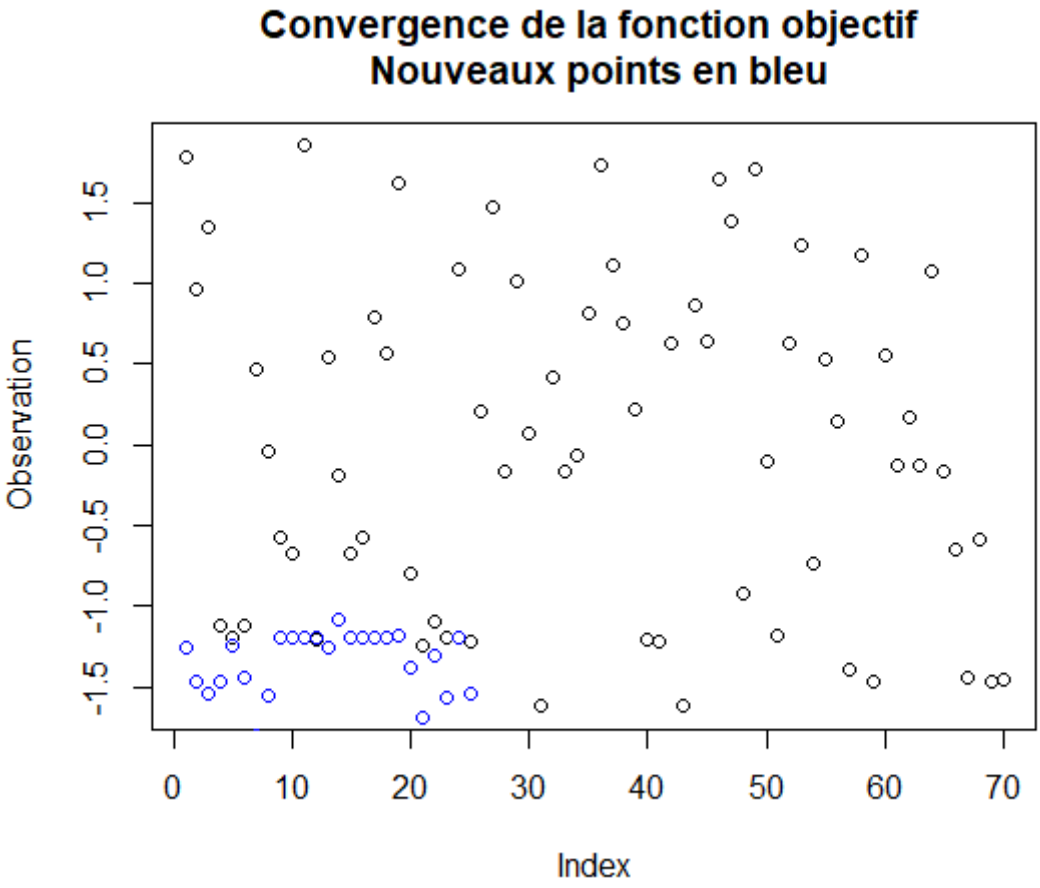



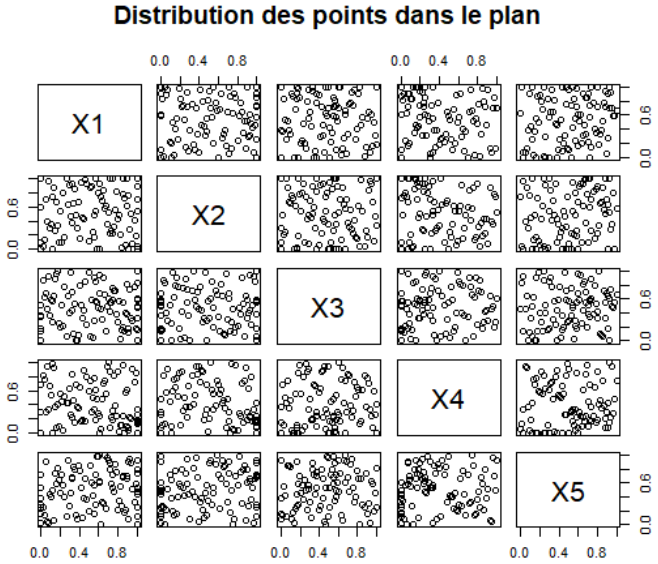



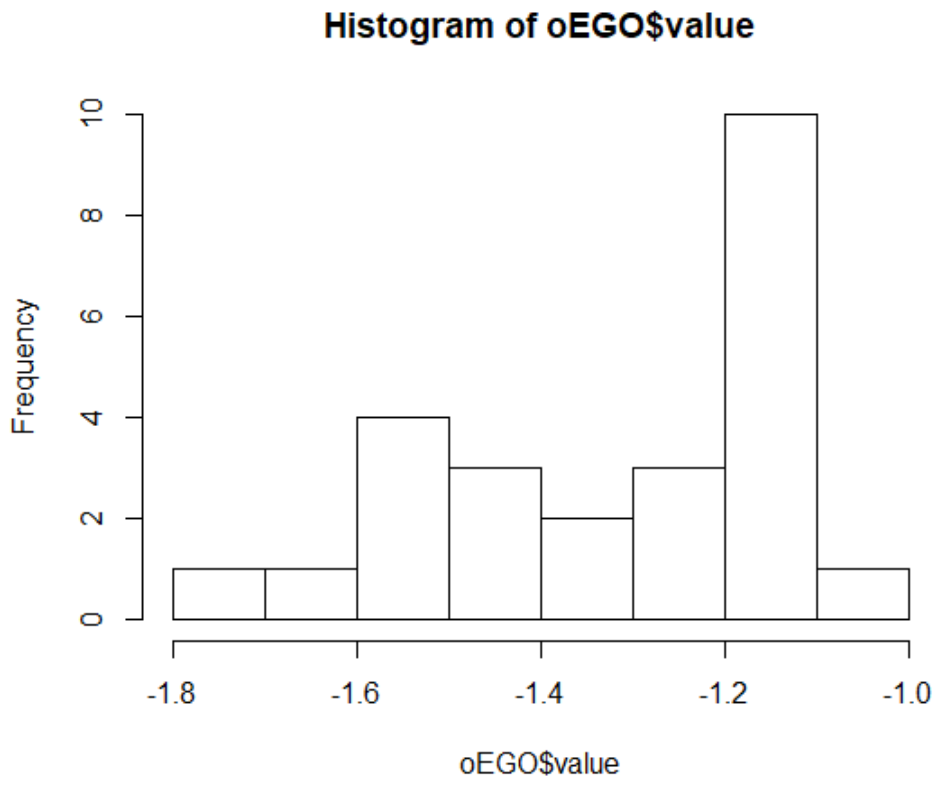
**Résultats de l'optimisation :**




Les nouveaux points améliorent bien la fonction objectif : on a un minimum plus petit que celui du plan d'expérience initial (voir les points en bleu).



De plus, on observe une bonne distribution des points dans le plan. Mais attention : ceci ne nous donne pas la réelle distribution des points dans l'espace de dimension 5 !




**Position de la source du volcan, soit le minimum de la fonction objectif :**

On a ci-dessous les valeurs de xs, ys, zs, a et p qui donnent la position de la source :

```
> min(oEGO$value)
[1] -1.79757
> ind <- which.min(oEGO$value)
> argMin <- unnormed_var[ind,]
> argMin
          xs           ys           zs           a           p
 366982.4736 7651000.0000    -678.2546     232.6128     223.3065
>
```