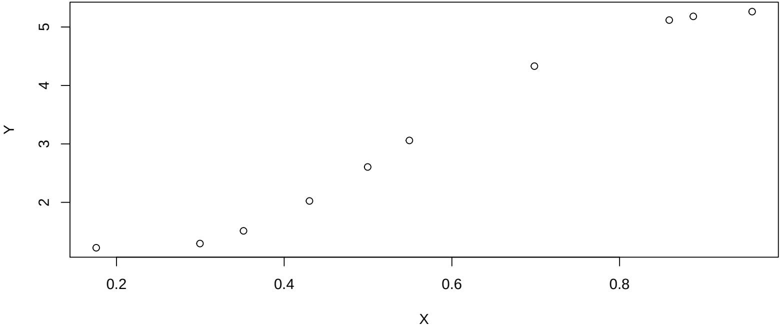
TP du cours métamodélisation - corrigé

Nicolas Durrande, Victor Picheny,

Q1

```
plot(X, Y)
```



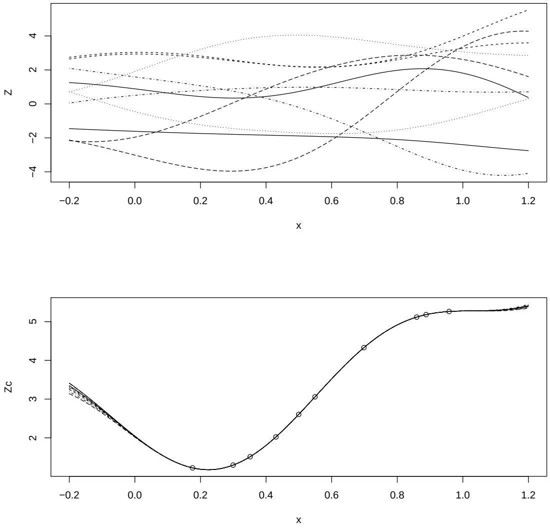
La fonction semble très régulière, probablement croissante.

Q2 et Q3

```
m <- km(y~1, design=data.frame(x=X), response=data.frame(y=Y),
        coef.trend = 0, nugget=1e-8,
        covtype="gauss", coef.var=4, coef.cov=.5)

x <- matrix(seq(-0.2, 1.2, 0.01))
Z <- t(simulate(m, 10, newdata=data.frame(x=x), cond=FALSE))
matplot(x, Z, type='l', col=1)

Zc <- t(simulate(m, 10, newdata=data.frame(x=x), cond=TRUE))
matplot(x, Zc, type='l', col=1)
points(X, Y)
```

`km` crée un modèle de krigeage, et `simulate` génère des trajectoires correspondant aux points `x`. En changeant `cond=TRUE`, on conditionne aux observations (les trajectoires passent par les observations).

Trajectoires interpolantes ou non, mais de même régularité



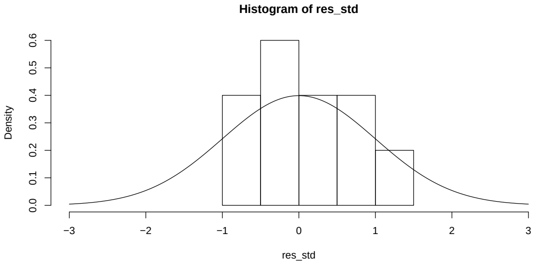
Q4 : étude d'un modèle km

```
m <- km(y~1, design=data.frame(x=X), response=data.frame(y=Y),
        covtype="gauss", coef.trend = 0, nugget=1e-8)
print(m)

Trend  coeff.:

 (Intercept)     0.0000

Covar. type  : gauss
Covar. coeff.:
               Estimate
    theta(x)     0.5349

Variance estimate: 35.01083

Nugget effect : 1e-08
```

- `Trend coeff` : moyenne, ici imposée égale à zéro
- `Covar. coeff` : paramètre de portée (homogène à une distance), estimé par max de vraisemblance
- `Variance estimate` : paramètre de variance du processus, estimé par max de vraisemblance

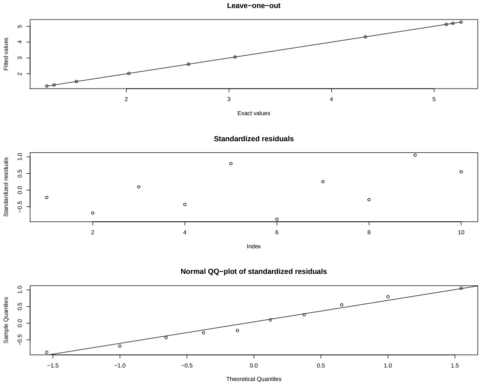
Q5

```
res <-leaveOneOut.km(m, type='SK')
res_std <- (Y-res$mean) / res$sd

hist(res_std, freq=FALSE, xlim=c(min(X, -3), max(X, 3)))
x <- seq(-3, 3, 0.03)
lines(x, dnorm(x))
```



Ici : les erreurs par validation croisée, une fois normalisées, sont raisonnablement gaussiennes centrées réduites.

Q5 : plot(m)

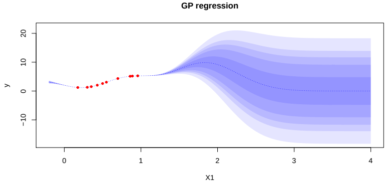


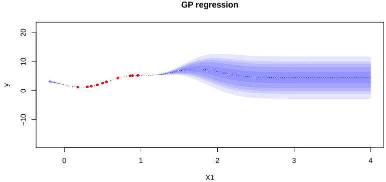
- Premier graphe : les erreurs sont toutes très faibles (modèle très précis)
- Deuxième graphe : on ne voit pas de structure pour les erreurs (OK)
- Troisième graphe : les résidus sont bien gaussiens (points sur la droite des quantiles)

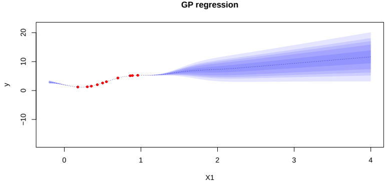
Q6

```
sectionview(m, xlim=c(-0.2, 4), ylim=c(-18, 22))

# modèle de krigeage ordinaire (estimation tendance constante)
m <- km(y~1, design=data.frame(x=X), response=data.frame(y=Y),
        covtype="gauss", nugget=1e-8)
sectionview(m, xlim=c(-0.2, 4), ylim=c(-18, 22))

# modèle de krigeage universel (estimation tendance lineaire)
m <- km(y~x, design=data.frame(x=X), response=data.frame(y=Y),
        covtype="gauss", nugget=1e-8)
sectionview(m, xlim=c(-0.2, 4), ylim=c(-18, 22))
```

Q6







En l'absence d'observations proches : le krigeage effectue un "retour à la moyenne".

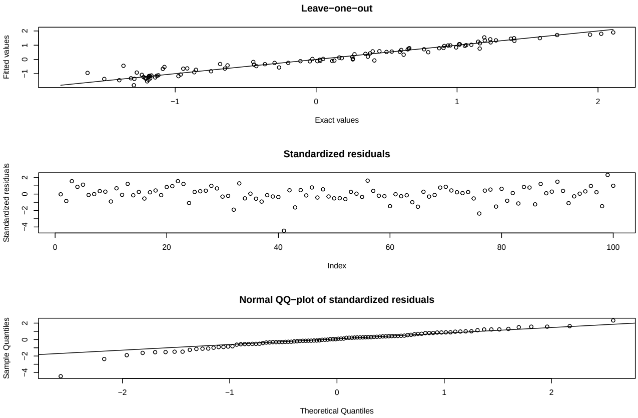
Choisir la tendance permet d'extrapoler... mais peut aussi amplifier l'erreur !

Q7 et Q8

```
km(formula = y ~ ., design = data.frame(x = X), response = data.frame(y = Y),
    covtype = "matern5_2", nugget = 1e-08)

Trend  coeff.:
                Estimate
 (Intercept)     -1.5942
         x.1      0.0819
         x.2     -0.0102
         x.3      0.8546
         x.4      2.5819
         x.5     -0.1406

Covar. type  : matern5_2
Covar. coeff.:
                Estimate
  theta(x.1)      1.9621
  theta(x.2)      1.9661
  theta(x.3)      1.6171
  theta(x.4)      0.4056
  theta(x.5)      0.0561

Variance estimate: 0.2128299
```

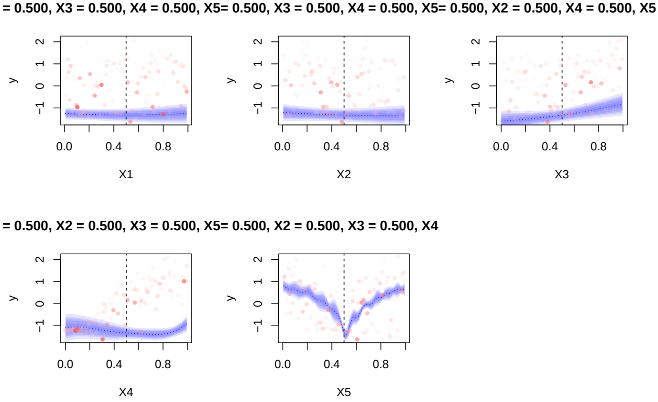
On observe bien : un paramètre de tendance et de portée par $x.i$. Sachant que chaque $x.i$ varie entre 0 et 1, une portée égale à 2 est très élevée (le krigeage est presque constant dans cette direction, $x.i$ est peu influent), et de 0.05 très petite (beaucoup de variation dans cette direction).

Q8



Validation : erreur par validation croisée plutôt bonne, quelques points imprécis pour les valeurs faibles de Y. QQ-plot : OK sauf pour les quantiles faibles.

Q8

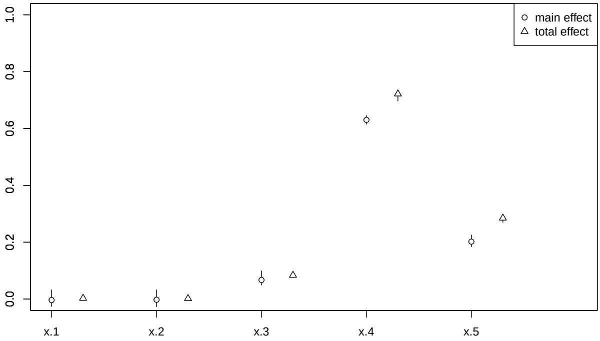


On retrouve bien : un modèle très "plat" pour $x.1$, $x.2$, $x.3$, très variable pour $x.5$.

Q10

```
getmean <- function(newdata, m) {
  pred <- predict(object=m, newdata=newdata, type="UK")
  return(pred$mean)
}
X1 <- data.frame(randomLHS(10000,5))
X2 <- data.frame(randomLHS(10000,5))
colnames(X1) <- colnames(X2) <- colnames(m@X)
res2 <- soboljansen(model = getmean, X1=X1, X2=X2, nboot = 50, conf = 0.95, m=m)
```

Q10



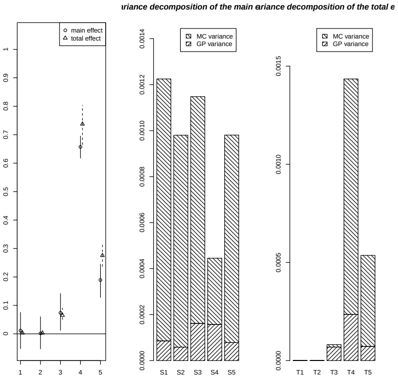
Très rapide... mais ne prend pas en compte l'erreur supplémentaire due au métamodèle.

Q10

```
X1 <- data.frame(randomLHS(1000,5))
X2 <- data.frame(randomLHS(1000,5))
candidate <- data.frame(randomLHS(100,5))
colnames(X1) <- colnames(X2) <- colnames(candidate) <- colnames(m@X)
res <- sobolGP(model = m, type="UK", MCmethod="soboljansen",
               X1=X1, X2=X2, nsim = 20, nboot=50, sequential = TRUE, candidate=candidate)
plot(res)
```

On est forcé de réduire la taille de X1 et X2.

Q10



Beaucoup plus long ! Mais on peut attribuer la part d'erreur due à Monte-Carlo (la taille de X1 et X2) et celle due au krigeage (essentiellement due à la taille du plan d'expériences). Ici : le modèle est très précis, donc l'erreur vient essentiellement du MC.