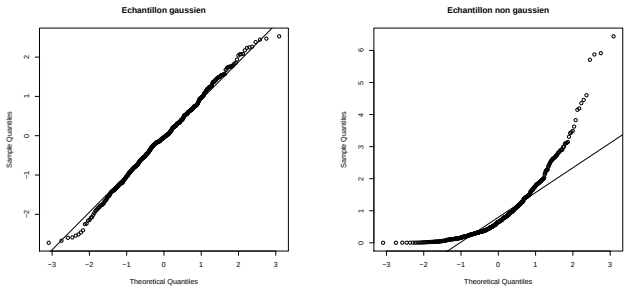
TP 2 : Régression non-paramétrique

Préambule. Dans ce TP, nous allons implémenter des procédures de régression non-paramétriques par noyaux et par polynômes locaux. Vous aurez besoin de charger les packages `locfit` et `np`.

On rappelle les fonctions suivantes :

- `par(mfrow=c(k,l))` permet d'afficher $k \times l$ graphiques sur une même figure.
- `lines()` (resp. `points()` permet d'ajouter une courbe (resp. des points) sur un graphe existant.
- `qqnorm(X)` permet de comparer graphiquement la distribution d'un échantillon `X` avec une distribution normale (si les points sont approximativement alignés, `X` est gaussien). `qqline(X)` permet d'ajouter la droite passant par le premier et troisième quartile :
  `qqnorm(X)`
  `qqline(X)`



  `qqplot(X,Y)` permet de comparer graphiquement les distributions de deux échantillons `X` et `Y`. Si les distributions sont identiques, les points sont approximativement alignés.
- `seq(a,b,l=m)` fournit un vecteur de `m` points équidistants entre `a` et `b`.

1 Première partie sur données simulées

1. Simuler un modèle de régression

$$Y_k = f(k/n) + \epsilon_k$$

où
— ϵ_k sont des variables i.i.d. gaussiennes centrées de variance σ^2
— la fonction de régression est

$$f(x) = 1 - x + 2x^2 - 0.8x^3 + 0.6x^4 - x^5$$

— sur une grille régulière $x = k/n$ avec $1 \leq k \leq n$

2. Tracer la vraie fonction de régression ainsi que le nuage de points pour plusieurs valeurs de n.

3. En utilisant des modèles de régression linéaires pour des polynômes de degré 1 à 10, construire des estimateurs de la vraie fonction de régression.
4. Les tracer sur le nuage de points.
5. Quels sont les degrés de polynômes qui fournissent les meilleurs résultats (visuels).
6. Calculer, pour chacun des modèles, l'erreur quadratique de prédiction ainsi que sa version empirique. Commenter.
7. Tracer les deux erreurs en fonction du degré.
8. On souhaite maintenant tester la stabilité de ces résultats en répétant l'expérience 100 fois.
9. Tracer les boites à moustaches pour ces deux types d'erreurs.
10. Commenter. Quel degré de polynôme fournit l'erreur la plus faible ?
11. En décomposant ces erreurs en un terme de biais plus un terme de variance, tracer sur un même graphique, l'erreur, le biais au carré et la variance. Commenter.
12. On estime maintenant la fonction de régression par un estimateur à noyau (Nadaraya Watson) en utilisant la commande `npreg` de R.

    L'estimateur de Nadaraya-Watson est implémenté dans la fonction `npreg` du package `np`. L'argument principal est la formule `Y~X` où `Y` est la variable réponse et `X` la variable explicative.

    ```
    mod.ker <- npreg(Y~X)
    ```

    La commande `names(mod.ker)` permet d'afficher le nom de l'ensemble des variables fournies par `npreg`.

    La valeur de l'estimateur $\hat{r}(x)$ pour x dans un vecteur $X0$ choisi par l'utilisateur est obtenue en spécifiant l'argument `exdat` dans la fonction `npreg`. Les valeurs $(\hat{r}(x), x \in X_0)$ sont alors contenues dans la variable `mean`.

    ```
    X0 <- seq(30,50,l=100)
    mod.kern <- npreg(Y~X, exdat=X0)
    rhatX0 <- mod.kern$mean
    plot(X0,rhatX0)
    ```

    Si on ne spécifie pas la valeur de `exdat`, `npreg` calcule l'estimateur aux points du vecteur `X`.
13. Tracer sur un même graphique l'estimateur obtenu, la vraie fonction de régression et le meilleur estimateur obtenu par polynôme dans les questions précédentes.
14. Comparer les erreurs de chacunes des méthodes.

2 Deuxième partie sur données réelles

Nous allons travailler sur 3 jeux de données :

- **Data A.** Les données `cps71` du package `np` fournissent le logarithme du salaire et l'âge pour 205 canadiens.

- **Data B.** Les données du data.frame `Alcool` fournissent la consommation d'alcool en g/jour pour 312 individus (variable `Alcohol.consumption`) et leur durée de vie divisée par la durée de vie moyenne (variable `Life`).
- **Data C.** Les données du data frame `Hormone` fournissent la dose d'hormone administrée au patient (variable `Hormone`) et le taux d'endocrine résultant dans le sang (variable `Endocrine`) lors d'une étude clinique.

Rq : Les données Data B et C ont été simulées, mais les distributions ont été choisies de manière réaliste sur la base de travaux scientifiques.

(I) Méthode de Nadaraya-Watson et Polynômes locaux

Dans cette section, vous allez vous familiariser avec des estimateurs de régresssion NP, en vous appuyant sur le jeu de données Data A.

I-1-a) Charger le jeu de données cps71 (commande `data(cps71)`). Afficher le début de la matrice de données `cps71` à l'aide de la fonction `head`.

I-1-b) On veut étudier la fonction de régression de la variable X_i égale à l'âge (colonne `age`) sur la variable Y_i égale au logarithme du salaire (colonne `logwage`). Définir les vecteurs `X` et `Y` correspondant de longueur $n = 205$, et tracer les points d'observations $(X_i, Y_i)_{i=1,\dots,n}$.

I-1-c) Définir une grille `x0` de 1000 points régulièrement espacés sur l'intervalle $[\min(X_1, \dots, X_n), \max(X_1, \dots, X_n)]$.

I-2) **Régression par noyaux (estimateur de Nadarya Watson)**. L'estimateur de Nadaraya-Watson est implémenté dans la fonction `npreg` du package `np`. L'argument principal est la formule `Y~X` où `Y` est la variable réponse et `X` la variable explicative.

```
mod.ker <- npreg(Y~X)
```

La commande `names(mod.ker)` permet d'afficher le nom de l'ensemble des variables fournies par `npreg`.

La valeur de l'estimateur $\hat{r}(x)$ pour x dans un vecteur $X0$ choisi par l'utilisateur est obtenue en spécifiant l'argument `exdat` dans la fonction `npreg`. Les valeurs $(\hat{r}(x), x \in X_0)$ sont alors contenues dans la variable `mean`.

```
X0 <- seq(30,50,l=100)
mod.kern <- npreg(Y~X, exdat=X0)
rhatX0 <- mod.kern$mean
plot(X0,rhatX0)
```

Si on ne spécifie pas la valeur de `exdat`, `npreg` calcule l'estimateur aux points du vecteur `X`.

La fenêtre optimale est calculée par une validation croisée (interne) mais peut également être choisie par l'utilisateur. Dans ce TP, nous travaillerons avec la valeur déterminée par `npreg`..

Question. Sur un même dessin, tracer les points d'observations $(X_i, Y_i)_{i=1,\dots,n}$ ainsi que l'estimateur $\hat{r}$ par noyau avec la fenêtre par défaut, calculé aux points de `x0` . On notera `rkern` le vecteur contenant les valeurs $(\hat{r}(x), x \in$ `x0`$)$.

I-3) **Estimateurs par polynômes locaux.** La fonction `locfit` du package `locfit` réalise la régression par polynôme locaux. L'argument principal est la formule `Y∼lp(X)` où `Y` est la variable réponse et `X` la variable explicative.

```
mod.lp <- locfit(Y∼lp(X))
```

Cet estimateur comporte deux paramètres de régularisation (le degré maximum du polynôme, et une quantité qui contrôle la fonction de poids). Ces paramètres comportent des valeurs par défaut, et peuvent également être contrôlés par l'utilisateur. Dans ce TP, nous travaillerons avec les valeurs par défaut.

La valeur de l'estimateur $\hat{r}(x)$ pour x dans un vecteur $X0$ choisi par l'utilisateur est obtenue dans un second temps à l'aide de la fonction `predict` qui requiert (au moins) deux arguments : un "modèle" de régression, et un ensemble de valeur où la fonction de regression doit être appliquée :

```
mod.lp <- locfit(Y∼lp(X))
X0 <- seq(30,50,l=100)
rhatX0 <- predict(mod.lp, X0)
```

Question. Sur un même dessin, tracer les points d'observations $(X_i, Y_i)_{i=1,\dots,n}$ ainsi que l'estimateur $\hat{r}$ par polynômes locaux, calculé aux points de `x0` . On notera `rlp` le vecteur contenant les valeurs $(\hat{r}(x), x \in$ `x0`$)$.

I-4) La fonction `lowess` implémente un estimateur de la fonction de régression qui mélange les k plus proches voisins et la régression par polynomes locaux de degré 1. Cette fonction est très souvent utilisée pour un examen visuel. La régularisation est contrôlée par le paramètre `f` (plus `f` est grand, plus l'estimateur est lissé) ; par défaut `f`$=2/3$.

```
plot(X,Y)
lines(lowess(X,Y), col='red')
lines( lowess(X,Y, f=0.1), col='blue')
```

(II) Validation croisée sur le jeu de données Data B

II-1-a) Charger le jeu de données Data B. On veut étudier la fonction de régression de la variable X_i égale à la consommation d'alcool sur la variable Y_i égale à l'âge de decès normalisé. Définir les vecteurs `X` et `Y` correspondant de longueur `n` $= 312$, et tracer les points d'observations $(X_i, Y_i)_{i=1,\dots,n}$.

II-1-b) Définir une grille `x0` de 1000 points régulièrement espacés sur l'intervalle $[\min(X_1, \dots, X_n), \max(X_1, \dots, X_n)]$

II-2-a) Calculer le vecteur `rkern` contenant les valeurs de l'estimateur par noyau aux points de `x0`, et le vecteur `rlp` contenant les valeurs de l'estimateur par polynômes locaux aux points de `x0`.

II-2-b) Tracer sur un même graphique :

- Les points $(X_i, Y_i)_{i=1,\dots,n}$
- L'estimateur par noyau, en rouge (argument `col='red'`)

- L'estimateur par polynômes locaux, en bleu (argument `col='blue'`)

II- 3) Dans cette question, nous allons voir pas-à-pas comment calculer l'erreur de validation croisée 10-fold, et tracer les valeurs $(\hat{Y}_i)_{i=1,\dots,n}$ estimées par VC en fonction des valeurs observées $(Y_i)_{i=1,\dots,n}$.

II-3-a) **Découpage de l'échantillon :**$\{1,\dots \mathtt{n}\}$: $\mathtt{n} = 312$ n'est pas divisible par 10, nous allons donc diviser $\{1,\dots \mathtt{n}\}$ en 10 sous-ensembles disjoints de taille 31 ou 32. Soit

```
J1 <- sample(rep(1:10, l=n))
```

La commande `table(J1)` donne les effectifs de chaque valeur dans le vecteur `J1`. Le vecteur `J1` contient 32 fois les valeurs 1 et 2, et 31 fois les valeurs 3,4,...,10, dans un ordre aléatoire. Pour tout $\ell = 1,\dots,10$, I_ℓ est défini par les positions des valeurs ℓ dans le vecteur `J1`.

Exple :

```
I2 <- which(J1==2)
```

II-3-b) Soit `l=1`. L'ensemble d'apprentissage 1 correspond aux observations telles que `J1` $\neq 1$, et l'ensemble de validation aux observations telles que `J1` $= 1$

- Calculer l'estimateur de régression par polynômes locaux à partir de l'ensemble d'apprentissage 1 :
  - variable explicative : `X[which(J1!=1) ]`
  - variable réponse : `Y[which(J1!=1)]`
- Calculer le vecteur `Yhat` des valeurs prédites par VC pour l'ensemble de validation1 (variables explicatives `X[which(J1==1) ]`
- Tracer les valeurs prédites `Yhat` en fonctions des valeurs observées `Y[which(J1==1)]` pour les observations de l'ensemble de validation 1.

II- 3-c) Nous allons appliquer la même procédure pour tous les $\ell = 1,\dots,10$:

- Définir un vecteur `Yhat.lp` de longueur `n`. Ce vecteur servira à stocker les valeurs prédites pour $\ell = 1,\dots,10$.
- Pour tout $\ell \in \{1,\dots,10\}$: calculer le vecteur `Yhat` des valeurs prédites par VC pour les observations de l'ensemble de validation ℓ $\{\hat{Y}_i, i \in I_\ell\}$, et les stocker dans le vecteur `Yhat.lp` à l'aide de la commande :
  `Yhat.lp[which(J1==l)] <- Yhat`

II-3-d) Tracer les valeurs prédites par VC en fonction des valeurs observées $(Y_i, \hat{Y}_i)_{i=1,\dots,\mathtt{n}}$, ainsi que la droite $y = x$. Calculer la corrélation de Pearson entre $(Y_i)_{i=1,\dots,\mathtt{n}}$ et $(\hat{Y}_i)_{i=1,\dots,\mathtt{n}}$ Que pensez-vous de la qualité de prédiction ? Basé sur ces données, pensez-vous qu'il y a un lien entre consommation d'alcool et mortalité ?

II-3-e) Calculer l'erreur de VC `Ecv.lp`

II-4) La véritable fonction de régression (utilisée pour simuler les données) est :

$$r(x) = 10^{-4}(x+40)^3 \exp\left(-\frac{x+40}{20}\right)$$

Sur un même graphe, tracer les observations $(X_i, Y_i)_{i=1,\dots,n}$, la vraie fonction r et l'estimateur par polynômes locaux. Que pensez vous de la qualité d'estimation ?

II-5) Comment expliquez-vous qu'on puisse avoir une bonne qualité d'estimation mais une mauvaise prédiction ?

(III) Validation croisée sur le jeu de données Data C

Dans cette section, nous allons analyser le jeu de données Data C. Les questions suivantes sont très similaires à celles de la section précédente. Il peut-être judicieux de réutiliser les scripts...

III-1) Charger le jeu de données Data C. On veut étudier la fonction de régression de la variable X_i égale à la dose d'hormone prescrite sur la variable Y_i égale au taux d'endocrine mesuré. Définir les vecteurs `X` et `Y` correspondant de longueur $n = 100$, et tracer les points d'observations $(X_i, Y_i)_{i=1,\dots,n}$.

Définir une grille `x0` de 1000 points régulièrement espacés sur l'intervalle $[\min(X_1, \dots, X_n), \max(X_1, \dots, X_n)]$

III-2) Tracer sur un mï£¡me graphique :
- Les points $(X_i, Y_i)_{i=1,\dots,n}$
- L'estimateur par noyau, en rouge (argument `col='red'`)
- L'estimateur par polynômes locaux, en bleu (argument `col='blue'`)

III-3) Calculer les valeurs prédites par VC (10-fold) pour l'estimateur par polynômes locaux, et tracer les valeurs prédites en fonctions des valeurs observées. Que pensez-vous de la qualité de prédiction ?

III-4) Un simple coup d'oeil aux données montre qu'un modèle linéaire n'est pas adapté. Néanmoins, à titre d'exercice, on va comparer les résultats ci-dessus avec les prédictions obtenues par un modèle linéaire.

Prédiction dans un modèle linéaire aux points d'un vecteur `X0`*, pour un modèle estimé à partir d'un vecteur de variables explicatives* `X` *et d'un vecteur de variables réponse* `Y` *:*

```
mod <- lm(Y~X)
co <- coef(mod)
rhatX0 <- co[1] + co[2]*X0
```

Calculer les valeurs `Yhat.lm` prédites par VC pour un modèle linéaire. Tracer ces valeurs prédites en fonction des valeurs observées et conclure sur les capacités prédictives du modèle. Comparer les erreurs de VC avec l'estimateur par polynômes locaux et l'estimateur linéaire.