Compte-rendu pour le TP Statistique

Problème 1 :

Question 1 : Calculez des statistiques descriptives des distributions de l'IBE et de l'IP : moyenne, médiane, écart-type, IQR.

Code source :

```
means <- apply(data, 2, mean)
medians <- apply(data, 2, median)
sds <- apply(data, 2, sd)
iqrs <- apply(data, 2, IQR)

result <- data.frame(
  Mean = means,
  Median = medians,
  SD = sds,
  IQR = iqrs
)
print("********** Question 1 **********")
print(result)
print("****************************************")
```

Résultat obtenu affichée sur la console :

```
 [1] "********** Question 1 **********"
        Mean  Median       SD      IQR
IBE 48.293333 50.0000 29.286263 50.25000
IP   4.475667  4.6505  1.201006  1.16975
[1] "****************************************"
```

Pour calculer les statistiques descriptives des distributions de l'IBE et de l'IP, on a utilisé plusieurs fonctions en R. La fonction mean() permet de calculer la moyenne, median() pour la médiane, sd() pour l'écart-type, et IQR() pour l'intervalle interquartile (IQR). Ces fonctions ont été appliquées à chaque colonne du jeu de données en utilisant apply(data, 2, fonction), où 2 indique que l'opération est effectuée sur les colonnes.

Les résultats obtenus montrent les caractéristiques de chaque distribution :

Pour l'IBE, la moyenne est de 48,29, la médiane est 50, l'écart-type est de 29,29, et l'IQR est de 50,25.

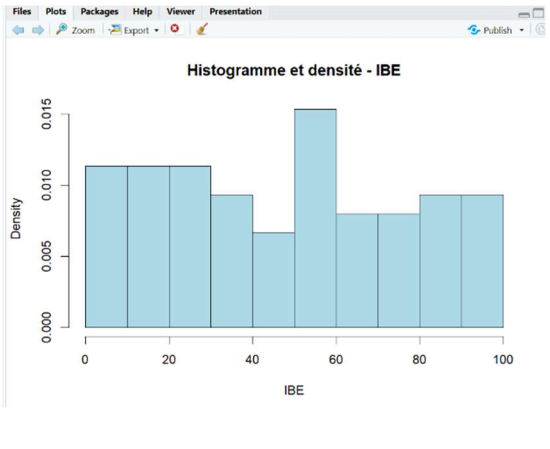
Pour l'IP, la moyenne est de 4,48, la médiane est 4,65, l'écart-type est de 1,20, et l'IQR est de 1,17.

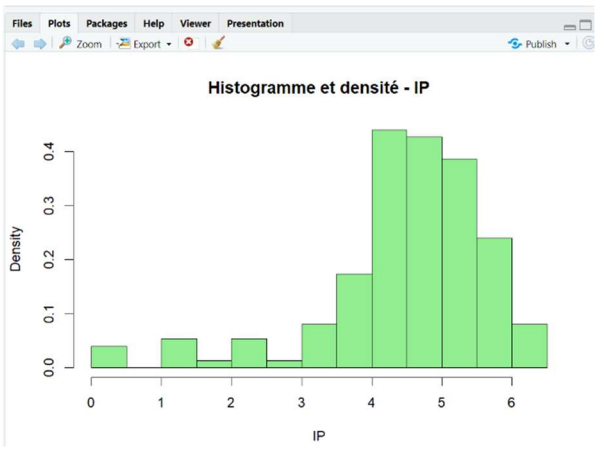
Question 2 : Affichez une estimation de densité par histogramme pour chacune des deux variables : IBE et IP

Code source :

```
hist_IBE = hist(data$IBE, freq = FALSE, main = "Histogramme et densité - IBE",xlab = "IBE", col = "lightblue", border = "black")

hist_IP = hist(data$IP, freq = FALSE, main = "Histogramme et densité - IP",
    xlab = "IP", col = "lightgreen", border = "black")
```

Résultats affichés :





Pour la question 2, on a utilisé la fonction hist() pour afficher des histogrammes avec une estimation de densité pour les variables IBE et IP. L'argument freq = FALSE permet de dessiner l'histogramme avec une densité normalisée plutôt que des fréquences brutes.

Pour chaque variable :

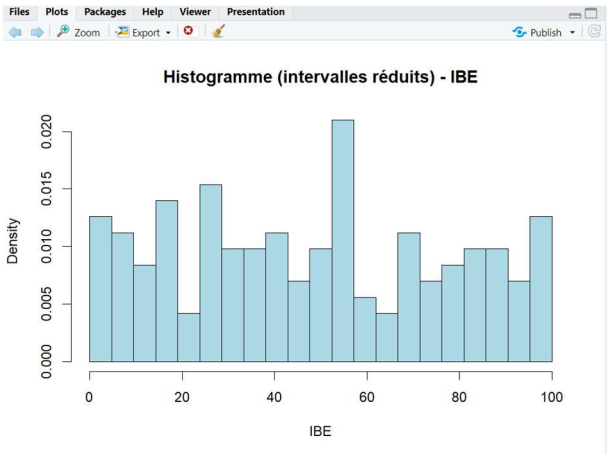
data$IBE et data$IP : spécifient les colonnes des données à utiliser.
main : définit le titre du graphique
xlab : spécifie le label de l'axe des x.
col : choisit la couleur de remplissage des barres,
border : détermine la couleur des bordures des barres.

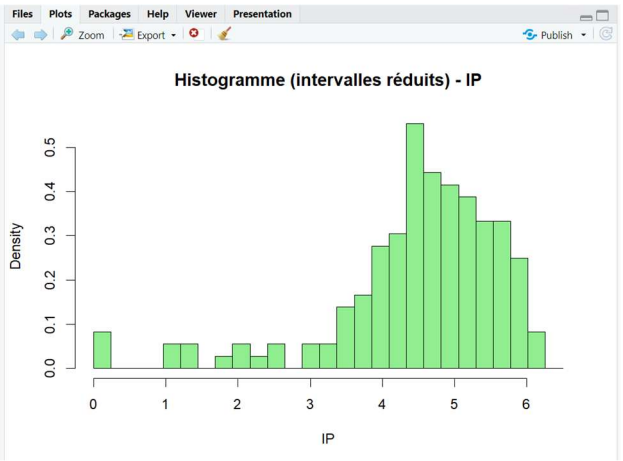
Question 3 : Affichez les mêmes histogrammes que ceux de la question précédente, mais avec des largeurs d'intervalles divisées par deux

Code source :

```
new_breaks_IBE <- seq(min(hist_IBE$breaks), max(hist_IBE$breaks), length.out = length(hist_IBE$breaks) * 2)
hist_IBE_breaks = hist(data$IBE, breaks =new_breaks_IBE, freq = FALSE, main = "Histogramme (intervalles réduits) - IBE",xlab = "IBE", col = "lightblue", border = "black")

new_breaks_IP <- seq(min(hist_IP$breaks), max(hist_IP$breaks), length.out = length(hist_IP$breaks) * 2)
hist_IP_breaks = hist(data$IP,breaks = new_breaks_IP, freq = FALSE, main = "Histogramme (intervalles réduits) - IP",
                      xlab = "IP", col = "lightgreen", border = "black")
```

Résultats affichés :





Pour la question 3, on a ajusté la largeur des intervalles des histogrammes en les divisant par deux. Cela a été réalisé en modifiant l'argument breaks de la fonction hist().
Voici les étapes détaillées :

- Nous avons utilisé seq() pour créer de nouveaux intervalles avec une densité plus fine. min(hist_IBE$breaks) et max(hist_IBE$breaks) définissent les limites des intervalles, et length.out = length(hist_IBE$breaks) * 2 double le nombre d'intervalles, réduisant ainsi leur largeur.
- Ensuite, hist() est appelé avec ces nouveaux intervalles (breaks = new_breaks_IBE et breaks = new_breaks_IP) pour les deux variables IBE et IP.
- Les autres arguments (freq = FALSE, main, xlab, col, border) restent les mêmes que dans la question précédente, pour maintenir une présentation cohérente.

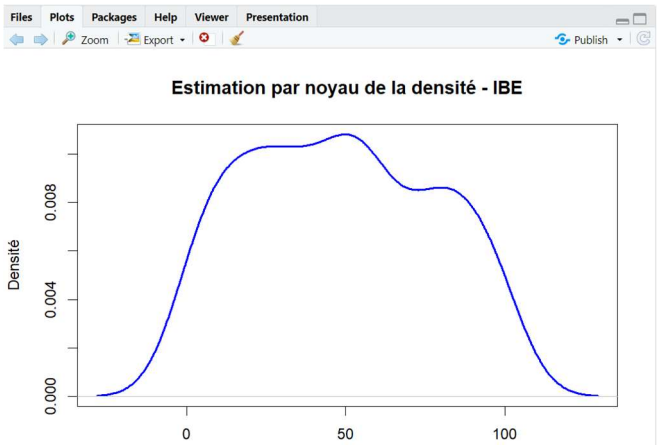
Question 4 : Affichez une estimation par noyau de la densité (en anglais : kernel density estimation) pour chacune des deux variables : IBE et IP

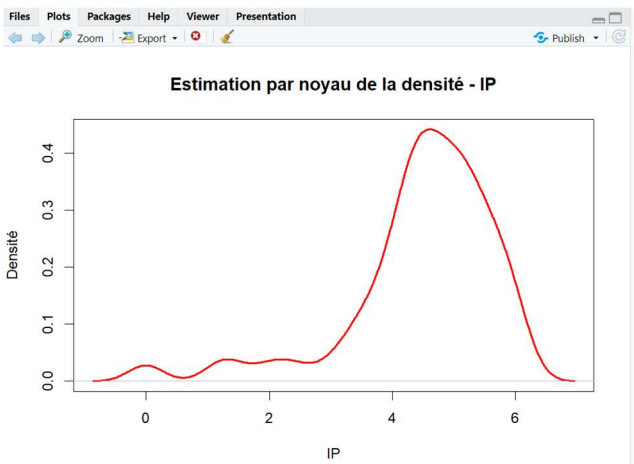
Code source :

```
kde_IBE <- density(data$IBE)
```

```
kde_IP <- density(data$IP)
plot(kde_IBE, main = "Estimation par noyau de la densité - IBE", xlab = "IBE", ylab = "Densité", col = "blue", lwd = 2)
plot(kde_IP,main = "Estimation par noyau de la densité - IP", xlab = "IP", ylab = "Densité", col = "red", lwd = 2)
```

Résultats affichés :





Pour la question 4, on a utilisé la méthode d'estimation par noyau (Kernel Density Estimation, KDE) pour estimer la densité des variables IBE et IP. Voici une explication des fonctions et des paramètres utilisés :

- **density()** : Cette fonction calcule l'estimation par noyau de la densité de probabilité pour une variable donnée. Nous avons appliqué density(data$IBE) et density(data$IP) pour calculer les densités respectivement pour IBE et IP.
- **plot()** : Ensuite, nous avons utilisé la fonction plot() pour afficher ces estimations de densité.
    - main permet de définir le titre du graphique.
    - xlab et ylab définissent les labels des axes des abscisses et des ordonnées.
    - col spécifie la couleur de la courbe de densité (bleu pour IBE, rouge pour IP).
    - lwd définit l'épaisseur de la courbe (2 ici pour rendre la ligne plus visible).

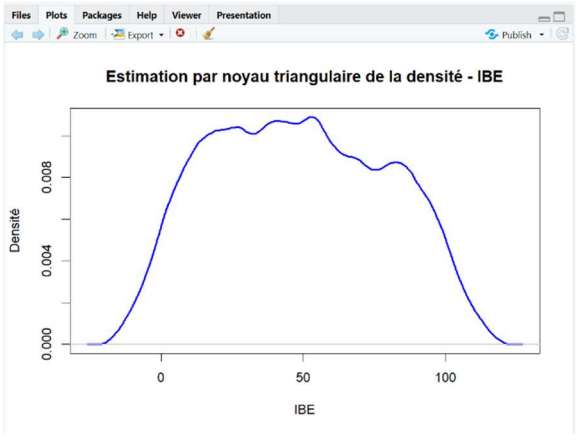
Question 5 : Refaites la Q4 - en utilisant un noyau triangulaire et en changeant la méthode du choix de paramètre de lissage par un validation croisée biaisée

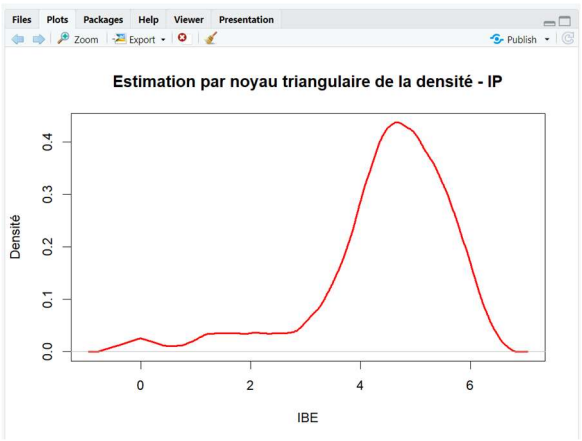
Code source :

```
kde_IBE_BIS <- density(data$IBE, kernel = "triangular", bw = "SJ")
kde_IP_BIS <- density(data$IP, kernel = "triangular", bw = "SJ")


plot(kde_IBE_BIS, main = "Estimation par noyau triangulaire de la densité - IBE",
    xlab = "IBE", ylab = "Densité", col = "blue", lwd = 2)
```

```
plot(kde_IP_BIS, main = "Estimation par noyau triangulaire de la densité - IP",
     xlab = "IBE", ylab = "Densité", col = "red", lwd = 2)
```

Résultats affichés :





On a utilisé la même fonction density() que la question 4, mais cette fois-ci avec des options supplémentaires :

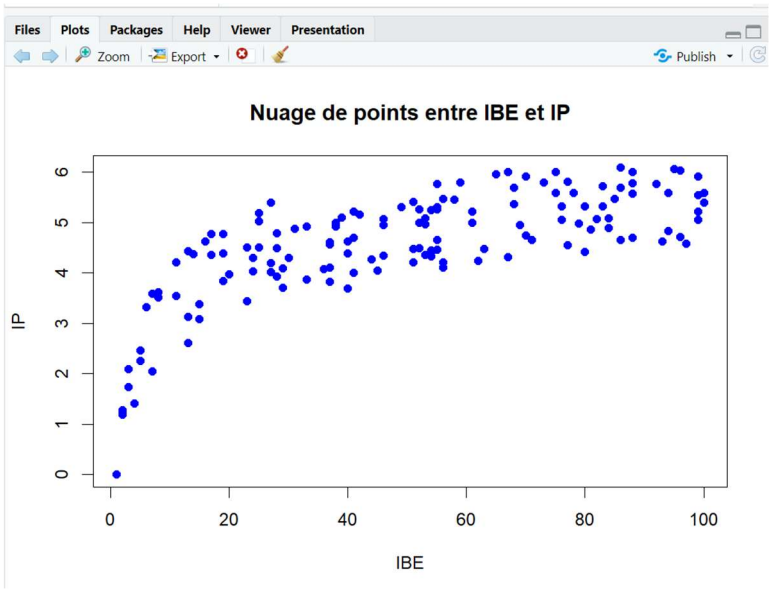
- kernel = "triangular" spécifie l'utilisation d'un noyau triangulaire pour l'estimation de la densité. Ce noyau donne une courbe plus douce par rapport à d'autres noyaux comme le noyau gaussien.
- bw = "SJ" définit la méthode de sélection du paramètre de lissage à l'aide de la validation croisée biaisée, qui permet de choisir automatiquement une valeur de lissage appropriée basée sur les données

Question 6 : Afficher le nuage de points entre les deux variables IBE et IP. Au regard de ce graphique, comment décririez-vous la relation entre ces deux variables ?

Code source :

```
plot(data$IBE, data$IP,
     main = "Nuage de points entre IBE et IP",
     xlab = "IBE", ylab = "IP",
     col = "blue", pch = 16)
```

Résultats affichés :



La fonction plot() est utilisée pour afficher la relation entre ces deux variables, où data$IBE est représenté sur l'axe des X et data$IP sur l'axe des Y. Le titre du graphique est défini avec main, et les étiquettes des axes sont spécifiées avec xlab et ylab. Les points sont colorés en bleu (col = "blue") et sont dessinés sous forme de cercles pleins (pch = 16). Ce graphique permet de visualiser la corrélation ou la dispersion entre les deux variables.

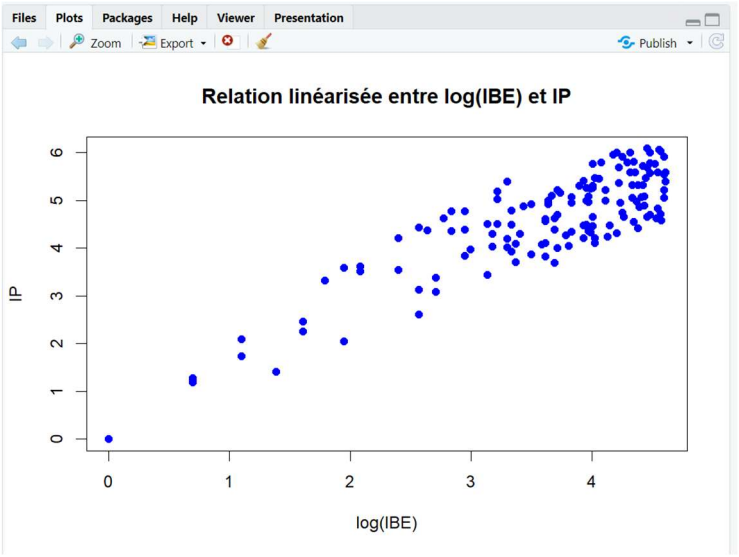
Sur la base de ce diagramme de dispersion, il semble y avoir une relation logarithmique entre l'IBE et l'IP.

Question 7 : Trouvez une transformation des données permettant de linéariser la relation entre les deux variables et affichez le nuage de points

Code source :

```
plot(log(data$IBE), data$IP,
    main = "Relation linéarisée entre log(IBE) et IP",
    xlab = "log(IBE)", ylab = "IP",
    col = "blue", pch = 16)
```

Résultat affiché :



La même fonction plot() a utilisée pour cette question, mais cette fois le log(data$IBE) est présenté sur l'axe X et data$IP sur l'axe Y, car on a trouvée qu'il semble y avoir une relation logarithmique entre l'IBE et l'IP. Donc on met log(IBE) sur X pour vérifier notre supposition.

Sur la base du diagramme de dispersion, on trouve qu'il y a une relation linéaire entre le log(IBE) et l'IP. Donc on a vérifié qu'il y a une relation logarithmique entre l'IBE et l'IP

Question 8 : À partir de ces données transformées, calculez le coefficient de corrélation de Pearson entre les deux variables. Calculez également les coefficients de corrélation de Spearman et Kendall.

Code source :

```
coef_corr_Pearson <- cor(log(data$IBE), data$IP, method = "pearson")
coef_corr_Spearman <- cor(log(data$IBE), data$IP, method = "spearman")
coef_corr_Kendall <- cor(log(data$IBE), data$IP, method = "kendall")

print("************** Question 8 **************")
print("le cofficient de corrélation de Pearson")
print(coef_corr_Pearson)
print("le cofficient de corrélation de Spearman")
print(coef_corr_Spearman)
print("le cofficient de corrélation de Kendall.")
print(coef_corr_Kendall)
print("****************************************")
```

Résultat affiché :

```
[1] "************** Question 8 **************"
[1] "le cofficient de corrélation de Pearson"
[1] 0.7235905
[1] "le cofficient de corrélation de Spearman"
[1] 0.7563248
[1] "le cofficient de corrélation de Kendall."
[1] 0.5727973
[1] "****************************************"
>
```

Dans cette question, la fonction cor() est utilisée pour calculer la corrélation entre deux variables. L'argument method permet de spécifier la méthode à utiliser :

- **method = "pearson"** : Calcule la corrélation de Pearson, qui mesure une relation linéaire entre les deux variables.
- **method = "spearman"** : Calcule la corrélation de Spearman, qui mesure la relation monotone entre les deux variables à l'aide des rangs.
- **method = "kendall"** : Calcule la corrélation de Kendall, qui évalue la relation monotone basée sur les paires concordantes et discordantes.

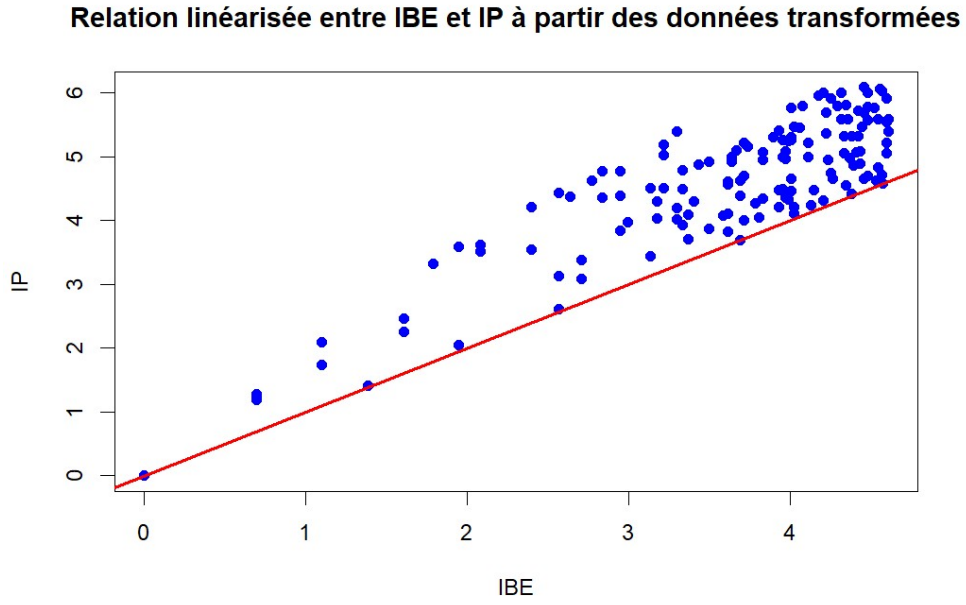
Question 9 : En reprenant le graphique de la question précédente, ajoutez

- une ligne diagonale (pente = 1), de couleur rouge
- les noms des abscisse et ordonnée (c'est-à-dire, IBE et IP).

Code source :

```
plot(log(data$IBE), data$IP,
    main = "Relation linéarisée entre IBE et IP à partir des données transformées",
    xlab = "IBE", ylab = "IP",
    col = "blue", pch = 16)
abline(a = 0, b = 1, col = "red", lwd = 2)
```

Résultat affiché :



La fonction abline() a été utilisé pour tracer une droite avec une ordonnée à l'origine égale à 0 et une pente égale à 1, col = "red" pour colorer la ligne en rouge, et lwd = 2 augmente son épaisseur.

Question 10 : Par défaut, les graphiques sont enregistrés dans des fichiers au format PDF. Reprenez la réponse de la question précédente, en faisant en sorte de produire en sortie un fichier au format PNG

Exporter les graphiques en forme PNG, en utilisant le bouton « export » dans le RStudio, puis choisir le format d'export PNG.

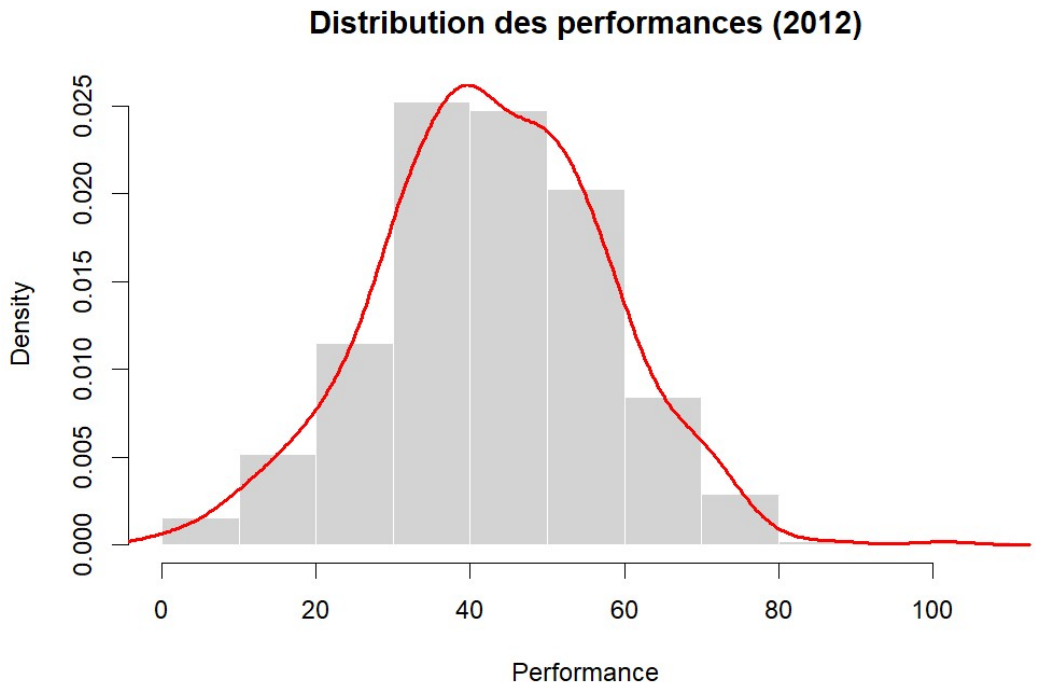
Problème 2 :

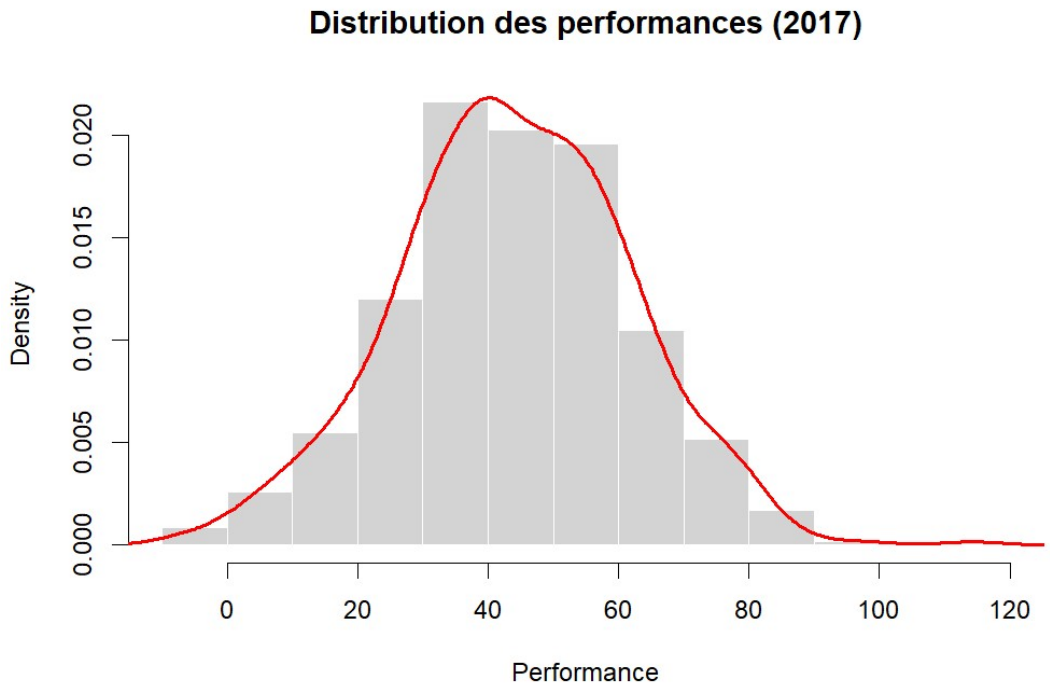
Question 1 : Affichez la distribution des performances pour les 3 années.

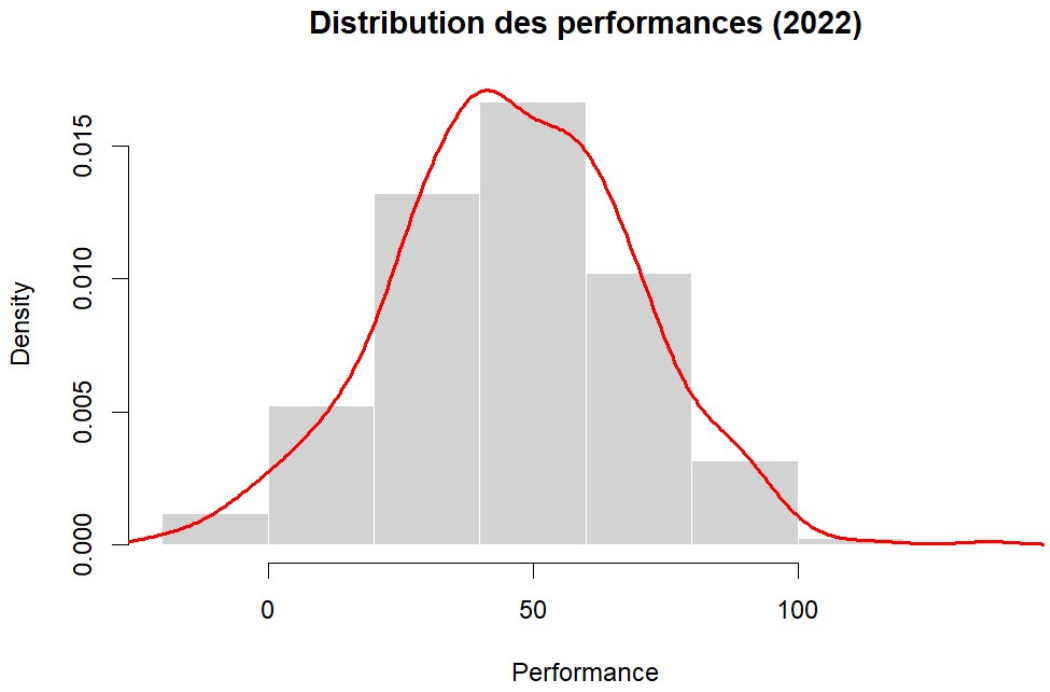
Code source :

```
hist(data2012,
     freq = FALSE,
     main = "Distribution des performances (2012)",
     xlab = "Performance",
     col = "lightgray",
     border = "white")
lines(density(data2012), col = "red", lwd = 2)

hist(data2017,
     freq = FALSE,
     main = "Distribution des performances (2017)",
     xlab = "Performance",
     col = "lightgray",
     border = "white")
lines(density(data2017), col = "red", lwd = 2)

hist(data2022,
     freq = FALSE,
     main = "Distribution des performances (2022)",
     xlab = "Performance",
     col = "lightgray",
     border = "white")
lines(density(data2022), col = "red", lwd = 2)
```

Résultats affichés :


Distribution des performances (2012)
Density
0.000
0.005
0.010
0.015
0.020
0.025
0
20
40
60
80
100
Performance


Distribution des performances (2017)
Density
0.000
0.005
0.010
0.015
0.020
0
20
40
60
80
100
120
Performance



Pour afficher les distributions des performances, j'ai utilisé la fonction hist() pour tracer un histogramme, et la fonction lines(density()) pour superposer une estimation de densité.

- **hist()** : Pour créer un histogramme.
  - L'argument freq = FALSE permet de normaliser les données pour afficher la densité au lieu des fréquences brutes.
  - main donne un titre au graphique, et xlab ajoute un label à l'axe des abscisses.
  - col = "lightgray" colore les barres en gris clair, et border = "white" définit la couleur des bordures.
- **lines(density())** : Pour ajouter une courbe de densité rouge (col = "red") à l'histogramme pour mieux visualiser la distribution des données. L'épaisseur de la ligne est augmentée avec lwd = 2 pour une meilleure lisibilité.

Question 2 : Visuellement, les performances semblent suivre une loi de probabilité normale. Effectuez un test de Shapiro-Wilk afin de vérifier cette hypothèse.

Code source :

```
shapiro_test_2012 <- shapiro.test(data2012)
shapiro_test_2017 <- shapiro.test(data2017)
shapiro_test_2022 <- shapiro.test(data2022)

print("***** Résultat du test de shapiro-wilk pour les données de l'année 2012  ******")
print(shapiro_test_2012)
```

```
print("***** Résultat du test de shapiro-wilk pour les données de l'année 2017  ******")
print(shapiro_test_2017)
print("***** Résultat du test de shapiro-wilk pour les données de l'année 2022  ******")
print(shapiro_test_2022)
```

Résultats affichés sur la console :

```
> print("***** Résultat du test de shapiro-wilk pour les données de l'année 2012  ******")
[1] "***** Résultat du test de shapiro-wilk pour les données de l'année 2012  ******"
> print(shapiro_test_2012)

        Shapiro-Wilk normality test

data:  data2012
W = 0.99775, p-value = 0.6317

>
> print("***** Résultat du test de shapiro-wilk pour les données de l'année 2017  ******")
[1] "***** Résultat du test de shapiro-wilk pour les données de l'année 2017  ******"
> print(shapiro_test_2017)

        Shapiro-Wilk normality test

data:  data2017
W = 0.99775, p-value = 0.6317

>
> print("***** Résultat du test de shapiro-wilk pour les données de l'année 2022  ******")
[1] "***** Résultat du test de shapiro-wilk pour les données de l'année 2022  ******"
> print(shapiro_test_2022)

        Shapiro-Wilk normality test

data:  data2022
W = 0.99775, p-value = 0.6317
```

Pour vérifier si les performances suivent une loi normale, j'ai utilisé le test de Shapiro-Wilk avec la fonction shapiro.test() pour les trois ensembles de données (2012, 2017 et 2022). Voici une explication :

- **shapiro.test()** : pour évaluer l'hypothèse nulle selon laquelle les données suivent une distribution normale.
    - Le résultat fournit deux éléments clés :
        - La statistique W, qui mesure la concordance entre les données observées et une distribution normale théorique.
        - Le p-value, qui indique si l'hypothèse de normalité peut être acceptée ou rejetée.

Exemple de résultat :

Pour data2012, le test donne W=0.99775W = 0.99775W=0.99775 et un **p-value** de 0.6317.

- Comme le **p-value** est supérieur à 0.05, nous ne pouvons pas rejeter l'hypothèse nulle, ce qui signifie que les données de 2012 suivent vraisemblablement une distribution normale.

Question 3 : Les actionnaires de l'entreprise se demandent si ces changements d'équipe dirigeante ont eu un effet significatif sur les performances des employés (en espérant que cet effet soit bon !). Répondez à leurs interrogations, en trouvant et calculant un

test statistique approprié. Pour choisir le bon test, vous vous poserez (entre autres) la question de « l'appariement » des données.

Code source :

```
data_indep <- data.frame(
  performance = c(data2012, data2017, data2022),
  periode = factor(rep(c("2012", "2017", "2022"), each = length(data2012)))
)

anova_indep <- aov(performance ~ periode, data = data_indep)
summary(anova_indep)
```

Résultat affiché :

```
> anova_indep <- aov(performance ~ periode, data = data_indep)
> summary(anova_indep)
              Df Sum Sq Mean Sq F value  Pr(>F)
periode        2   3583  1791.5   5.046 0.00653 **
Residuals   1746 619937   355.1
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
>
```

Dans cette question, on effectue une analyse de variance (ANOVA) pour évaluer si les performances des employés diffèrent significativement entre les trois périodes (2012, 2017, et 2022). Voici une explication des étapes et résultats obtenus :

- Le dataframe data_indep est créé pour regrouper toutes les données de performance dans une seule colonne (performance) et une colonne catégorielle indiquant la période correspondante (periode)
- La fonction aov() est utilisée pour effectuer le test ANOVA avec la formule performance ~ periode. Cela signifie que nous testons si la variable performance varie en fonction des niveaux de periode (les trois années).

Résumé des résultats :

Le tableau de résultats (summary(anova_indep)) montre un **F-statistic** de **5.046** et une **p-value** de **0.00653**.

Une p-value inférieure à 0.05 indique que nous rejetons l'hypothèse nulle : **les performances diffèrent significativement entre au moins deux des périodes**.

Question 4 : Trouvez et calculez un équivalent non-paramétrique du test de la question précédente.

Code source :

```
kruskal_test <- kruskal.test(
  x = c(data2012, data2017, data2022),
  g = factor(rep(c("2012", "2017", "2022"), each = length(data2012)))
```

)
print(kruskal_test)

Résultat affiché :

```
+ )
> print(kruskal_test)

        Kruskal-Wallis rank sum test

data:  c(data2012, data2017, data2022) and factor(rep(c("2012", "2017", "2022"), each = length(data2012)))
Kruskal-Wallis chi-squared = 9.5144, df = 2, p-value = 0.00859
```

Dans cette question, nous effectuons un test de Kruskal-Wallis, une alternative non-paramétrique à l'ANOVA. Ce test est utilisé lorsque les données ne respectent pas complètement les hypothèses de normalité ou d'homogénéité des variances.

- La fonction kruskal.test() est utilisée pour comparer les trois groupes de données.
  - x : Toutes les données de performance combinées.
  - g : Un facteur indiquant la période correspondante pour chaque donnée.

Résumé des résultats :
Le test retourne une statistique de Kruskal-Wallis (**chi-squared = 9.5144**) avec une **p-value** de **0.00859**.
Comme la p-value est inférieure à 0.05, nous rejetons l'hypothèse nulle : **il existe des différences significatives dans les performances entre les périodes**.