TP1: Probabilités - Densité de probabilité et fonction de répartition d'une variable aléatoire quantitative continue

Les deux TP de Biostat-1 ont pour objectif de vous initier à l'utilisation du logiciel R (voir https://www.r-project.org/ ou https://en.wikipedia.org/wiki/R_(programming_language)). Ce logiciel est massivement utilisé dans les laboratoires de recherche de Biologie pour l'analyse statistique des données expérimentales. Mais, en tant que langage de programmation, R peut également être utilisé pour de nombreux autres usages comme le calcul numérique, la simulation informatique de processus aléatoires, le traitement (simple) de bases de données, le traitement de données géographiques, etc.

Dans cette première séance de TP, vous allez utiliser R pour ses nombreuses fonctionnalités dans le domaine des probabilités. En particulier, vous allez tracer les fonctions de densité de probabilité et de répartition de lois normales et de mélanges de lois normales. Vous allez également calculer des probabilités et des quantiles pour des variables suivant des lois normales. Et tout ça, sans utiliser les tables statistiques !

Exercice 1: Prise en main du logiciel R/RStudio

R est un langage de programmation et un environnement intégré de manipulation de données, de calcul et de préparation de graphiques. R est gratuit, libre d'accès et téléchargeable sur le site https://www.r-project.org/. Au cours des TP, vous allez utiliser R comme un outil pour réaliser des études statistiques.

Pour faciliter l'apprentissage de R, nous utiliserons une surcouche de R, appelée RStudio (voir https://rstudio.com/ ou https://en.wikipedia.org/wiki/RStudio). Il s'agit d'un environnement de développement multiplateforme pour R. RStudio permet notamment d'avoir accès sur une même interface à la console R, au fichier script, aux pages d'aide, aux graphiques et à la vue de l'environnement.

1. Lancez le logiciel RStudio.

2. Ouvrez un nouveau fichier script. Pour cela, allez dans File > New File > R Script.

3. Enregistrez ce fichier sous le nom "BS1_TP1.R".

N.B.: Pour sauvergarder un script R, il est important de vous assurer que le nom de fichier a bien une extension en ".R". Dans le cas contraire, RStudio ne fonctionnera pas correctement.

Le fichier "BS1_TP1.R" sera le fichier dans lequel vous allez écrire vos commandes R. N'oubliez pas de le sauvegarder au fur et à mesure !

D'un point de vue pratique, nous vous conseillons de conserver ouverts le script R dans logiciel RStudio et la fenêtre affichant l'énoncé tout au long du TP. Vous fonctionnerez pas à pas, en alternant entre énoncé et script.

Dans ce TP, vous devrez principalement effectuer des copier-coller de commandes R (déjà fournies) entre énoncé et script pour pouvoir les exécuter. Il vous sera progressivement demandé un peu plus de travail (voir les cases `## ToDo`), comme la modification de commandes voire la création de nouvelles commandes.

La première partie de ce TP a pour but de vous familiariser avec le logiciel R. Pour cela, vous allez exécuter quelques commandes simples qui vous permettront de découvrir la philosophie de R et les différents objets que vous utiliserez par la suite.

En R, différents types de variables peuvent-être définies: des nombres entiers, des nombres décimaux, des chaînes de caractères, etc. Ces variables sont stockées dans des 'objets'. En R, un objet possède deux attributs

principaux:

- Son *nom*. Pour nommer un objet R, vous pouvez choisir un nom quelconque pourvu qu'il ne contienne pas d'espace (utiliser le caractère `_` si besoin) ou de caractères spéciaux. Généralement on choisit un nom d'objet constitué de lettres (minuscules ou majuscules), de chiffres et du caractère `_`. Notez que R est sensible à la "casse", c'est-à-dire que l'objet nommé `x` (en minuscule) est différent de l'objet `X` (en majuscule).
- Son *contenu*. Le contenu d'un objet peut prendre plusieurs formes: un vecteur ('vector'), une matrice ('matrix'), un tableau de données ('data.frame'), une liste ('list'), etc.

Dans ce TP, vous utiliserez principalement des vecteurs. Un vecteur est une succession (ordonnée) de valeurs de même type (que des nombres, ou que des chaînes de caractères, ..., bref que des "trucs" de même "nature"). Grosso-modo, un "vecteur R" est un tableau unidimensionnel possédant autant de cases qu'il faut pour accueillir chacune des valeurs. Par comparaison, une "matrice R" est un tableau bidimensionnel de `nrow` lignes et `ncol` colonnes, possédant donc `nrow*ncol` cases.

1- Créez votre premier objet dans R

Création de l'objet nommé `x1` contenant la valeur 12

Dans la fenêtre de script (en haut à gauche), saisissez la commande suivante:

```
x1 <- 12
```

- Dans RStudio, lorsque vous saisissez une commande dans la **fenêtre de script**, il ne se passe strictement rien, même après avoir appuyé sur 'ENTER'. Pas de panique, c'est tout à fait normal. Pour exécuter la commande, il faut explicitement le faire en plaçant le curseur sur la ligne de la commande (par exemple, en début, milieu ou fin de ligne, voire en sélectionnant entièrement la ligne), puis en appuyant sur 'CTRL + ENTER' (ou en cliquant sur le bouton 'Run' en haut à droite de la fenêtre de script).
- Vous devriez constater que la ligne de commande a été copiée, validée et exécutée au niveau de la fenêtre 'Console' (en bas à gauche). La fenêtre 'Console' devrait en effet afficher: > x1 <- 12 C'est-à-dire la ligne figurant dans la fenêtre de script précédée du caractère '>' qui correspond à ce qu'on appelle "**invite de commande**" (ou "prompt" en anglais).
- Vous devriez également constater que la fenêtre 'Environment' (en haut à droite) affiche maintenant une rubrique 'Values' contenant le nom 'x1' et la valeur '12'.
- Le signe `<-` (ou plus précisemment la succession, sans espace, du caractère `<` et du caractère `-`) permet d'attribuer une valeur à un objet. Dans cet exemple, on attribue la valeur `12` à un objet qu'on a choisit de nommer `x1` (N.B.: il s'agit bien d'un nom d'objet R vailde constitué d'une lettre et d'un chiffre). Cette commande est appelée une '**affectation**' (en anglais, 'assignment'). S'il existe des variantes de syntaxe pour l'affectation dans R, nous vous conseillons fortement d'utiliser systématiquement la syntaxe: nom de la variable (à gauche), signe `<-` (au milieu), valeur (à droite).

Félicitations ! Vous venez de créer votre premier objet dans R. En d'autres termes, vous avez réussi à stocker dans la mémoire de l'ordinateur une valeur (en l'occurence 12) que vous pourrez réutiliser ou modifier à loisir à l'aide de l'objet que vous avez choisi de nommer 'x1'.

Vérification du contenu de l'objet `x1`

Pour afficher le contenu d'un objet sur la **console** (fenêtre en bas à gauche), il suffit d'écrire son nom et de valider:

```
x1
```

Vous devriez voir `[1] 12` s'afficher sur la console. A ce stade, ne vous préoccupez pas de l'affichage `[1]`. Ce qui est pertinent à ce stade est de constater que l'objet `x1` contient une et une seule valeur: `12`.

Dans la suite du TP vous allez créer de nombreuses variables. Pensez à régulièrement vérifier leur contenu par affichage sur la console.

Au fait, avez-vous pensé à sauvegarder le script contenant votre glorieuse première commande R ? Si ce n'est pas le cas, faites-le via le menu File > Save, le raccourci clavier 'CTRL + S', ou le bouton avec l'icône de "disquette".

2- Créez votre premier vecteur

Créez l'objet nommé `vect` contenant les valeurs 12.7, 45 et -2.1 (dans cet ordre):

```
vect <- c(12.7, 45, -2.1)
```

Il s'agit toujours d'une commande d'affectation `<-`, mais cette fois, la valeur que l'on stocke dans l'objet `vect` n'est plus une valeur unique, mais un vecteur de trois valeurs.

La fonction `c(...)` permet de créer un vecteur en "concaténant" (d'où le "c") plusieurs valeurs, c'est-à-dire en les enchaînant les unes après les autres.

L'**appel à une fonction** sous R se reconnait facilement par la présence d'un couple de parenthèses (ouvrante `(` et fermante `)`) suivant immédiatement le nom de la fonction. N.B.: en R, les parenthèses peuvent aussi apparaitre dans des calculs arithmétiques pour bien définir l'ordre de priorité des opérations. Mais dans ce cas, la parenthèse ouvrante `(` n'est pas précédée d'un nom. Dans l'exemple, on a appelé la fonction `c(...)` (c'est le nom de la fonction qui permet de concaténer des valeurs) avec trois "**arguments**": `12.7`, `45`et `-2.1`. Remarquez que le séparateur décimal est un point `.` (convention anglophone) et que la virgule `,` permet de séparer les arguments.

Vérifiez dans la fenêtre 'Environment' que vous avez bien créé un nouvel objet nommé `vect`. Vous devriez voir l'affichage `num[1:3] 12.7 45 -2.1` ce qui signifie que `vect` est un vecteur de valeurs numériques (`num`) ordonnées de la position `1` à la position `3`.

3- Obtenez de l'aide

Le logiciel R contient un grand nombre de fonctions pré-programmées. Le détail de la syntaxe d'appel de ces fonctions est accessible via la fonction **`help(...)`**. Pour accéder à la page d'aide d'une fonction, par exemple la fonction `seq(...)`, il faut taper la commande suivante:

```
help(seq)
```

La page d'aide apparaît dans l'onglet "Help" de la fenêtre en bas à droite. Cette page vous renseigne sur:

- la manière d'appeler la fonction (on appelle ceci 'usage'),
- les différents arguments de la fonction,
- la sortie de la fonction,
- des exemples d'utilisation de la fonction.

4- Créez une séquence de nombres

La fonction `seq(...)` permet de générer un vecteur de nombres compris entre une valeur initiale et une valeur finale en utilisant un pas d'incrémentation que l'on peut choisir (1 par défaut). Elle a trois arguments principaux:

- `from`: argument qui définit la première valeur du vecteur,
- `to`: argument qui définit la dernière valeur du vecteur,
- `by`: le pas d'incrémentation.

Créez le vecteur `vect1` qui contient toutes les valeurs allant de -3 à 3 par pas de 1.2.

```
vect1 <- seq(from=-3, to=3, by=1.2)
```

- Bien qu'il existe des variantes de syntaxe, nous vous conseillons d'utiliser le plus souvent possible le nom explicite des arguments des fonctions. En d'autres termes, si la commande `seq(-3, 3, 1.2)` produit exactement le même résultat que la commande `seq(from=-3, to=3, by=1.2)`, la deuxième

solution est beaucoup plus compréhensible et mémorisable. Notez qu'un script a vocation à être relu par un humain plutôt que par une machine. Si ce n'était pas le cas, on programmerait dans un "langage machine" directement compréhensible par l'ordinateur.

- Remarquez que, contrairement au signe `<-` utilisé pour affecter une valeur à une variable, il faut utiliser le signe `=` pour donner une valeur à un paramètre de fonction.

Combien de valeurs dans un vecteur ?

Pour déterminer le nombre d'éléments contenus dans l'objet `vect1` (c'est-à-dire la taille du vecteur `vect1`), exécutez la commande:

```
length(vect1)
```

Remarquez que le résultat de la fonction (normalement vous devez obtenir la valeur 6) s'affiche dans la fenêtre 'Console' (en bas à gauche) et qu'il n'y a pas de modification de votre 'Environment' (en haut à droite), en particulier pas de création ou de modification des objets R existants. En effet, vous avez demandé le calcul du nombre de cases de `vect1`, mais vous n'avez pas utilisé l'opérateur l'affectation `<-`. La valeur 6 a été calculée, puis affichée à l'écran mais pas sauvegardée dans la mémoire de l'ordinateur.

Une méthode alternative pour afficher le contenu d'un objet R

Dans la 1re section, nous avons vu que pour afficher le contenu d'un objet sur la console (fenêtre en bas à gauche) il suffit d'écrire son nom puis de valider. Une manière alternative est d'utiliser explicitement la fonction `print(...)`:

```
print(vect1)
```

Vous devriez voir s'afficher `[1] -3.0 -1.8 -0.6  0.6  1.8  3.0`, c'est-à-dire les 6 valeurs contenues dans l'objet-vecteur`vect1`. Cette commande, qui provoque le même résultat que la commande `vect1` seule, a néanmoins l'avantage de spécifier explicitement qu'on souhaite un affichage (en anglais, 'print'). De plus, dans certaines situations, comme lors de l'exécution automatique d'un script, l'utilisation de la fonction `print(...)` permet de s'assurer que l'affichage aura lieu.

5- Générez des nombres aléatoires

La fonction `rnorm(...)` permet de tirer aléatoirement et indépendamment des valeurs dans une loi normale. Cette fonction admet trois paramètres principaux:

- `n`: nombre de valeurs à tirer dans la loi normale,
- `mean`: espérance de la loi normale (c'est-à-dire la quantité notée μ dans le cours de Probabilités de L2),
- `sd` (*standard deviation*): écart-type de la loi normale (c'est-à-dire la quantité notée σ dans le cours de Probabilités de L2).

Remarquez que R utilise une convention différente de celle utilisée dans le cours de Probabilités de L2 pour le deuxième paramètre de la loi normale, celui qui détermine sa variabilité. Dans R, il faut spécifier l'écart-type σ, alors que dans le cours, on utilise le plus souvent la variance σ^2.

Créez un vecteur nommé `vect2` contenant 1000 valeurs tirées aléatoirement dans une distribution normale $\mathcal{N}(\mu = 5, \sigma^2 = 4)$

```
vect2 <- rnorm(n=1000, mean=5, sd=sqrt(4))
```

La fonction `sqrt(...)` permet de calculer la racine carrée d'un nombre (en anglais 'square root', d'où l'abréviation 'sqrt').

Copiez et modifiez la commande précédente pour créer un vecteur nommé `vect3` contenant 1000 valeurs tirées aléatoirement dans une distribution normale $\mathcal{N}(\mu = 2, \sigma^2 = 6)$.

```
## ToDo
```

6- Faites votre premier graphique avec R

Tracez le nuage de points correspondant aux valeurs du vecteur `vect3` en fonction des valeurs du vecteur `vect2`. Pour cela, utilisez la fonction `plot(...)` en spécifiant ses deux arguments principaux:

- `x`: vecteur qui contient les coordonnées de l'abscisse pour tous les points,
- `y`: vecteur qui contient les coordonnées de l'ordonnée pour tous les points.

```
plot(vect2, vect3)
```

D'autres arguments peuvent être utilisés pour paramétrer le graphique:

- `main`: pour ajouter un titre,
- `xlab` et `ylab`: pour définir le nom des axes,
- `col`: pour changer la couleur des points.

```
plot(vect2, vect3, col="red", main="Nuage de points", xlab="valeurs tirées dans une distribution N(5,4)
     ylab="valeurs tirées dans une distribution N(2,6)")
```

- Modifiez la commande précédente pour obtenir un graphique où les points sont colorés en cyan.
- En utilisant les arguments `xlim` et `ylim`, modifiez l'étendue de chacun des axes.
- En utilisant l'argument `pch` modifiez la forme des points.
- En utilisant l'argument `cex` modifiez la taille des points.

```
## ToDo
```

Pour savoir quelles valeurs utiliser vous pouvez consulter des ressources disponibles sur internet. En voici quelques unes:

- R pour les débutants, Emmanuel Paradis (page 43 pour l'étendue et page 51 pour la forme des points): https://cran.r-project.org/doc/contrib/Paradis-rdebuts_fr.pdf
- R color cheatsheet, Melanie Frazier (page 4 pour le nom des couleurs): https://www.nceas.ucsb.edu/sites/default/files/2020-04/colorPaletteCheatsheet.pdf

Si ce n'est pas encore fait, ne pensez-vous pas qu'il serait temps de sauvegarder votre script ?

Exercice 2: Etude de la loi normale centrée-réduite

Dans cette partie, vous allez utiliser une variable aléatoire Z suivant une distribution normale centrée-réduite.

Avant de vous précipiter sur du code R, remémorez-vous quelques caractéristiques des distributions normales:

- Retrouvez dans le cours la formule mathématique de la fonction de densité de probabilité d'une loi normale quelconque, puis celle d'une loi normale centrée-réduite.
- Quelle est la forme de la fonction de densité de toute loi normale ? Quelles sont ses caractéristiques ?
- Quelle est l'espérance de Z ?
- Quelle est la variance de Z ?

1- Manipulation de la fonction `dnorm(...)` pour calculer la fonction de densité

Utilisée avec un seul argument `x`, la fonction `dnorm(...)` renvoie la fonction de densité de probabilité d'une loi normale centrée-réduite évaluée en `x`:

- `x`: un vecteur de valeurs pour lesquelles on veut calculer la valeur de la densité de probabilité

La commande suivante permet de calculer la valeur de densité de probabilité d'une loi normale centrée-réduite en 2:

```
dnorm(x=2)
```

Vous devriez obtenir la valeur 0.05399097, résultat de:

$$\frac{1}{\sqrt{2\pi}} \exp(-2)$$

R possède donc une fonction préprogrammée pour calculer la fonction de densité. Il est inutile de le faire en utilisant la constante π, la fonction exponentielle et la fonction racine carrée.

Utilisez la fonction `dnorm(...)` pour calculer la densité de probabilité de Z pour chacune des valeurs suivantes: -1.645, -1, 0, 1 et 1.96.

```
dnorm(x=c(-1.645, -1, 0, 1, 1.96))
```

Retrouver l'espérance et la variance à partir de la fonction de densité (optionnel)

On souhaite retrouver les valeurs connues pour l'espérance et la variance de la variable Z en utilisant les capacités d'intégration numérique de R. En effet, on peut se rappeler que pour toute variable aléatoire X de fonction de densité de probabilité f, l'espérance $\mathbb{E}(X)$ et la variance $\mathbb{V}(X)$ sont définies par:

$$\mathbb{E}(X) = \int_{-\infty}^{+\infty} u\, f(u)\, du$$

$$\mathbb{V}(X) = \int_{-\infty}^{+\infty} (u - \mathbb{E}(X))^2 \times f(u)\, du$$

Pour une loi normale centrée-réduite, on connait la fonction R calculant la fonction de densité: `dnorm(...)`

Reste "juste" à savoir comment faire une intégration numérique avec R. Pour cela, on peut utiliser la fonction `integrate(...)` en spécifiant une borne inférieure `lower` et une borne supérieure `upper`. Par ailleurs, les valeurs `-Inf` et `+Inf` représentent respectivement $+\infty$ et $+\infty$.

Exécutez la commande suivante:

```
EspZ <- integrate(function(u) { return( u*dnorm(u) ) }, lower=-Inf, upper=+Inf)$value
```

La syntaxe est un peu plus compliquée que ce qu'on a vu jusqu'à présent, mais à part le `$value` et l'utilisation des accolades qu'on ne détaillera pas dans cette séance, on comprend sans trop de difficultés qu'on intègre (`integrate`) entre $+\infty$ (`lower=-Inf`) et $+\infty$ (`upper=+Inf`) une fonction (`function`) qui renvoie (`return`) $u\, f(u)$ où f est la fonction de densité de la loi normale centrée-réduite (`dnorm`).

La valeur de `EspZ` est-elle celle que vous attendiez ?

Copiez et modifiez la commande précédente pour calculer la variance de Z:

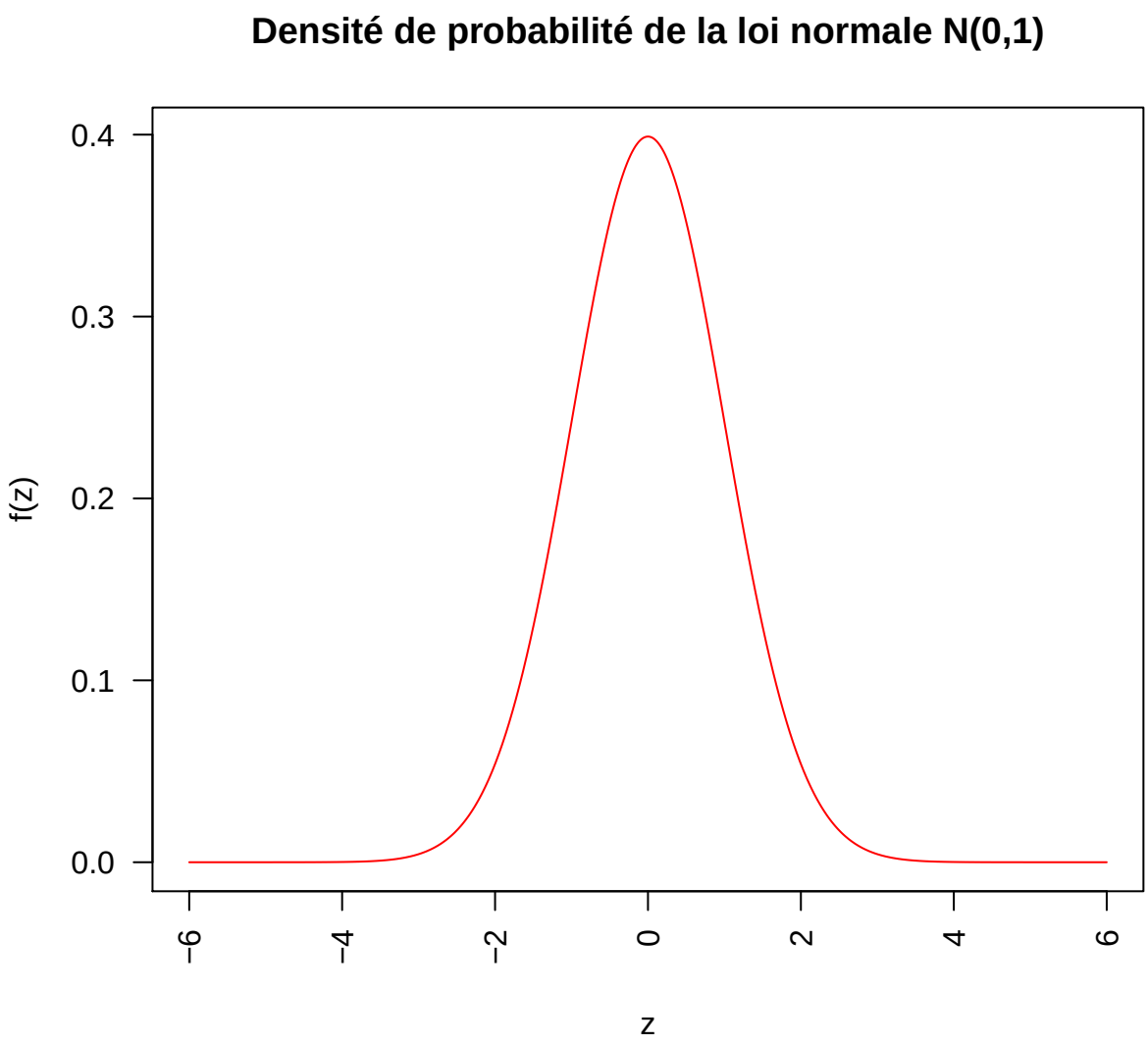
```
## ToDo
```

2- Tracé de la fonction de densité de probabilité d'une distribution normale centré-réduite $\mathcal{N}(0,1)$

Utilisez la procédure suivante pour tracer la courbe de la fonction de densité de probabilité d'une loi normale centré-réduite $\mathcal{N}(0,1)$:

1. Créez le vecteur `ValX` contenant toutes les valeurs de x allant de -6 à 6 par pas de 0.01.
2. Pour toutes les valeurs de x_i contenues dans le vecteur `ValX`, calculez les valeurs de densité de probabilité $f(x_i)$ correspondantes, f désignant la fonction de densité de probabilité d'une loi normale centré-réduite $\mathcal{N}(0,1)$. Stockez les valeurs $f(x_i)$ dans le vecteur `VectDens`.
3. A partir des valeurs de x_i contenues dans le vecteur `VectX` et des valeurs de densité $f(x_i)$ contenues dans le vecteur `VectDens`, tracez la fonction de densité de probabilité de la distribution normale centrée-réduite $\mathcal{N}(0,1)$. N'oubliez pas de mettre des noms à vos axes et un titre à votre graphique. Dans la fonction `plot(...)`, vous devez utiliser l'argument `type="l"`, pour tracer une courbe.

```r
ValX <- seq(from=-6, to=6, by=0.01)
VectDens <- dnorm(x=ValX)
plot(ValX, VectDens, type="l", xlab="z", ylab="f(z)", col="red",
     main="Densité de probabilité de la loi normale N(0,1)", las=2)
```



L'argument `las=2` permet de placer les labels des axes perpendiculairement aux axes.

3- Calculs de probabilités pour une distribution normale centré-réduite $\mathcal{N}(0,1)$

3-1 En utilisant la fonction `integrate(...)`

Rappelez l'expression de la probabilité $\mathbb{P}(a < X < b)$ pour une variable aléatoire continue X quelconque en fonction de sa densité de probabilité f:

$$\mathbb{P}(a < X < b) = ...$$

- En utilisant la fonction `integrate(...)`, on peut calculer la probabilité:

$$p_1 = \mathbb{P}(Z \leq -1)$$

```
p1 <- integrate(function(x) { return( dnorm(x) ) }, lower=-Inf, upper=-1)$value
```

- Sur le même principe, calculez les probabilités suivantes:

$$p_2 = \mathbb{P}(Z \leq 0)$$

```
## ToDo
```

$$p_3 = \mathbb{P}(Z \geq 2)$$

```
## ToDo
```

$$p_4 = \mathbb{P}(-1.645 \leq Z \leq 1.96)$$

```
## ToDo
```

3-2 En utilisant la fonction `pnorm(...)`

Rappelez l'expression de la probabilité $\mathbb{P}(a < X < b)$ pour une variable aléatoire continue X quelconque en fonction de sa fonction de répartition F:

$$\mathbb{P}(a < X < b) = ...$$

Utilisée avec un seul argument `q`, la fonction `pnorm(...)` renvoie la fonction de répartition Φ d'une loi normale centrée-réduite évaluée en `q`:

- `q`: un vecteur de valeurs pour lesquelles on veut calculer la valeur de la fonction de répartition

La commande suivante permet de calculer la probabilité $\mathbb{P}(Z \leq -1)$ avec $Z \sim \mathcal{N}(\mu = 0, \sigma^2 = 1)$.

```
pnorm(q=-1)
```

Vous devriez obtenir la valeur 0.1586553, résultat de:

$$\int_{-\infty}^{-1} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{u^2}{2}\right) \, du$$

- Sur le même principe, c'est-à-dire en utilisant la fonction `pnorm(...)`, recalculez les probabilités que vous avez calculées à la section 3-1 avec la fonction `integrate(...)`:

$$\mathbb{P}(Z \leq 0)$$

```
## ToDo
```

$$\mathbb{P}(Z \geq 2)$$

```
## ToDo
```

$$\mathbb{P}(-1.645 \leq Z \leq 1.96)$$

```
## ToDo
```

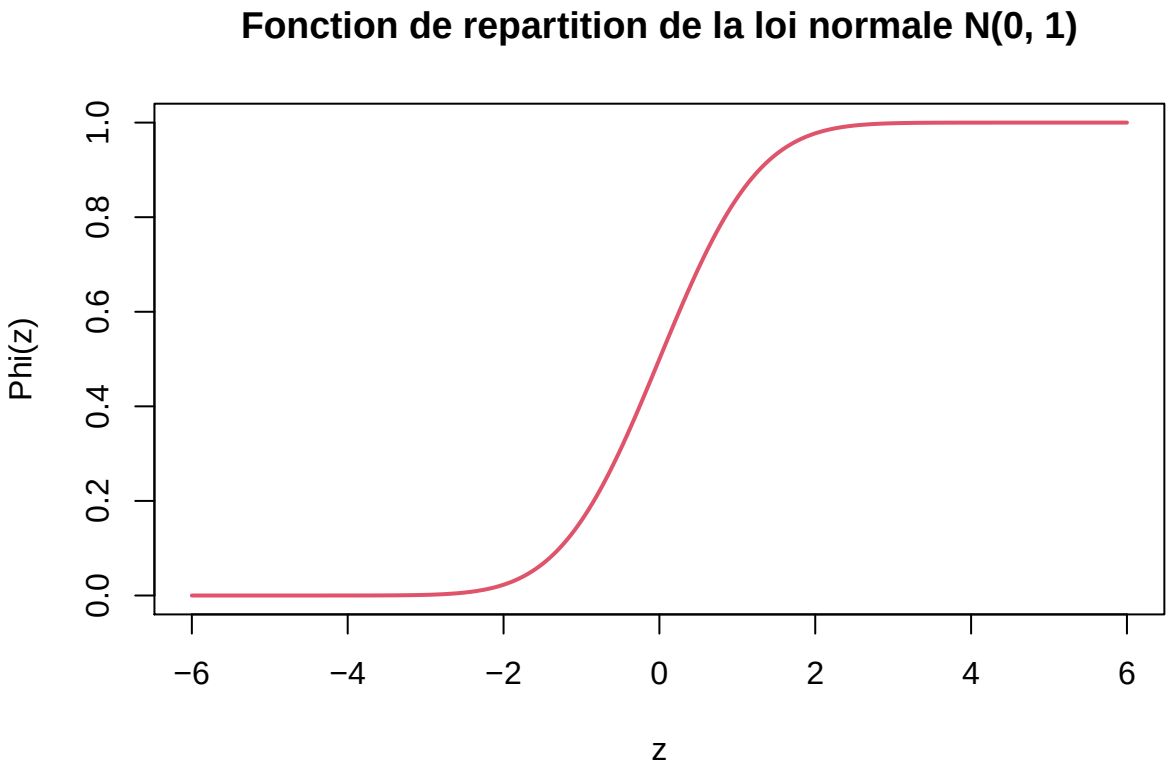
Vous devriez retrouver exactement les mêmes valeurs numériques que dans la section 3-1.

4- Tracé de la courbe de la fonction de répartition de la loi normale centrée-réduite

En utilisant la fonction `pnorm(...)` et une procédure similaire à celle utilisée précédemment pour la fonction de densité, tracez la courbe de la fonction de répartition Φ de la loi normale centrée-réduite.

```
## ToDo (create a ValX vector from -6 to 6 by step of 0.01)
## ToDo (compute the VectProb vector containig the value of phi for each value of ValX)
## ToDo (plot VectProb as a function of ValX)
```

Vous devez obtenir le graphique suivant:



5- Calcul de la table du Φ

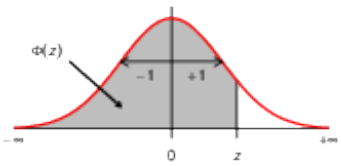
La table A de la fonction de répartition Φ de la distribution normale centrée-réduite qui vous a été fournie dans le polycopié de TD (voir une copie ci-dessous) donne, pour une variable normale centrée-réduite Z les valeurs de $\Phi(z) = \mathbb{P}(Z \leq z)$ pour z de 0 à 2.99 par pas de 0.01, puis pour quelques valeurs au delà de $z = 3$.

On se propose d'utiliser R pour calculer des valeurs intermédiaires non données par la table A.

Développez une procédure pour calculer les valeurs de $\Phi(z)$ pour z entre -0.02 et -0.01 par pas de 0.001 (11 valeurs à arrondir à 4 décimales). En d'autres termes, il faut trouver l'ensemble des commandes R (éventuellement une seule commande) permettant d'obtenir les données manquantes du tableau suivant:

z	-0.020	-0.019	-0.018	-0.017	-0.016	-0.015	-0.014	-0.013	-0.012	-0.011	-0.010
$\Phi(z)$	...	...	...	...	...	...	...	...	...	...	...

Table A
Fonction de répartition Φ de la distribution normale centrée-réduite



La table donne la fonction de répartition Φ d'une variable aléatoire Z distribuée suivant la loi normale centrée-réduite :

$$\Phi(z) = \mathrm{P}\left[Z \leq z\right]$$

z	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.00	0.5000	0.5040	0.5080	0.5120	0.5160	0.5199	0.5239	0.5279	0.5319	0.5359
0.10	0.5398	0.5438	0.5478	0.5517	0.5557	0.5596	0.5636	0.5675	0.5714	0.5753
0.20	0.5793	0.5832	0.5871	0.5910	0.5948	0.5987	0.6026	0.6064	0.6103	0.6141
0.30	0.6179	0.6217	0.6255	0.6293	0.6331	0.6368	0.6406	0.6443	0.6480	0.6517
0.40	0.6554	0.6591	0.6628	0.6664	0.6700	0.6736	0.6772	0.6808	0.6844	0.6879
0.50	0.6915	0.6950	0.6985	0.7019	0.7054	0.7088	0.7123	0.7157	0.7190	0.7224
0.60	0.7257	0.7291	0.7324	0.7357	0.7389	0.7422	0.7454	0.7486	0.7517	0.7549
0.70	0.7580	0.7611	0.7642	0.7673	0.7704	0.7734	0.7764	0.7794	0.7823	0.7852
0.80	0.7881	0.7910	0.7939	0.7967	0.7995	0.8023	0.8051	0.8078	0.8106	0.8133
0.90	0.8159	0.8186	0.8212	0.8238	0.8264	0.8289	0.8315	0.8340	0.8365	0.8389
1.00	0.8413	0.8438	0.8461	0.8485	0.8508	0.8531	0.8554	0.8577	0.8599	0.8621
1.10	0.8643	0.8665	0.8686	0.8708	0.8729	0.8749	0.8770	0.8790	0.8810	0.8830
1.20	0.8849	0.8869	0.8888	0.8907	0.8925	0.8944	0.8962	0.8980	0.8997	0.9015
1.30	0.9032	0.9049	0.9066	0.9082	0.9099	0.9115	0.9131	0.9147	0.9162	0.9177
1.40	0.9192	0.9207	0.9222	0.9236	0.9251	0.9265	0.9279	0.9292	0.9306	0.9319
1.50	0.9332	0.9345	0.9357	0.9370	0.9382	0.9394	0.9406	0.9418	0.9429	0.9441
1.60	0.9452	0.9463	0.9474	0.9484	0.9495	0.9505	0.9515	0.9525	0.9535	0.9545
1.70	0.9554	0.9564	0.9573	0.9582	0.9591	0.9599	0.9608	0.9616	0.9625	0.9633
1.80	0.9641	0.9649	0.9656	0.9664	0.9671	0.9678	0.9686	0.9693	0.9699	0.9706
1.90	0.9713	0.9719	0.9726	0.9732	0.9738	0.9744	0.9750	0.9756	0.9761	0.9767
2.00	0.9772	0.9778	0.9783	0.9788	0.9793	0.9798	0.9803	0.9808	0.9812	0.9817
2.10	0.9821	0.9826	0.9830	0.9834	0.9838	0.9842	0.9846	0.9850	0.9854	0.9857
2.20	0.9861	0.9864	0.9868	0.9871	0.9875	0.9878	0.9881	0.9884	0.9887	0.9890
2.30	0.9893	0.9896	0.9898	0.9901	0.9904	0.9906	0.9909	0.9911	0.9913	0.9916
2.40	0.9918	0.9920	0.9922	0.9925	0.9927	0.9929	0.9931	0.9932	0.9934	0.9936
2.50	0.9938	0.9940	0.9941	0.9943	0.9945	0.9946	0.9948	0.9949	0.9951	0.9952
2.60	0.9953	0.9955	0.9956	0.9957	0.9959	0.9960	0.9961	0.9962	0.9963	0.9964
2.70	0.9965	0.9966	0.9967	0.9968	0.9969	0.9970	0.9971	0.9972	0.9973	0.9974
2.80	0.9974	0.9975	0.9976	0.9977	0.9977	0.9978	0.9979	0.9979	0.9980	0.9981
2.90	0.9981	0.9982	0.9982	0.9983	0.9984	0.9984	0.9985	0.9985	0.9986	0.9986

Table calculée à l'aide de la fonction pnorm de R

Exemples de lecture :

$$\Phi(1.96) = \Phi(1.90 + 0.06) = 0.9750 \qquad \text{et} \qquad \Phi(1.64) = \Phi(1.60 + 0.04) = 0.9495$$

Table pour les grandes valeurs de z

z	3.0	3.2	3.4	3.6	3.8	4.0	4.5
$\Phi(z)$	0.998650	0.999313	0.999663	0.999841	0.999928	0.999968	0.999997

Table calculée à l'aide de la fonction pnorm de R

Remarque :

$$\Phi(z) = 1 - \Phi(-z) \qquad \text{et} \qquad \Phi(-z) = 1 - \Phi(z)$$

Par exemple :

$$\Phi(-1.96) = 1 - \Phi(1.96) = 1 - 0.9750 = 0.0250$$

Figure 1: Table A: fonction de répartition de la distribution normale centrée-réduite

```
## ToDo (create the appropriate vector of "z")
## ToDo (compute and round the values of "phi(z)")
```

Même question pour z variant entre 3 et 3.2 par pas de 0.02 (11 valeurs à arrondir à 6 décimales).

z	3.00	3.02	3.04	3.06	3.08	3.10	3.12	3.14	3.16	3.18	3.20
$\Phi(z)$	...	...	...	...	...	...	...	...	...	...	...

```
## ToDo
## ToDo
```

Fonctions à utiliser: `pnorm(...)` et `round(...)`

6- Calculs de quantiles pour une distribution normale centré-réduite $\mathcal{N}(0, 1)$

La fonction `qnorm(...)` est la fonction réciproque de la fonction `pnorm(...)`. Elle renvoie le quantile théorique pour une loi normale. En d'autres termes, elle prend en entrée une probabilité p et renvoie le nombre z dont la valeur de la fonction de répartition Φ est la probabilité proposée:

$$\Phi(z) = \mathbb{P}(Z \leq z) = p$$

Par exemple:

```
qnorm(p=0.5)
```

renvoie 0, la médiane d'une distribution normale centrée-réduite.

En utilisant la fonction `qnorm(...)`, calculez les valeurs de z_A, z_B et z_C telles que:

$$\mathbb{P}(Z < z_A) = 0.05$$

```
## ToDo
```

$$\mathbb{P}(Z < z_B) = 0.975$$

```
## ToDo
```

$$\mathbb{P}(Z > z_C) = 0.23$$

```
## ToDo
```

Calculez maintenant, à l'aide de la fonction `pnorm(...)`, les valeurs de $\Phi(z)$ pour $z = z_A$, $z = z_B$, $z = z_C$.

```
## ToDo
## ToDo
## ToDo
```

Les résultats étaient-ils attendus ? Pourquoi ?

7- Calcul de la table de l'écart-réduit

La table B des valeurs critiques de la distribution normale centrée-réduite qui vous a été fournie dans le polycopié de TD (voir une copie ci-dessous) donne, pour une variable normale centrée-réduite Z, les valeurs ϵ_α telle que $\mathbb{P}(|Z| \geq \epsilon_\alpha) = \alpha$, pour α de 0 à 0.99 par pas de 0.01, puis pour quelques valeurs plus petites que $\alpha = 10^{-3}$.

On pourra remarquer que si $\mathbb{P}(|Z| \geq \epsilon_\alpha) = \alpha$, alors $\mathbb{P}(Z \leq +\epsilon_\alpha) = \Phi(\epsilon_\alpha) = 1 - \alpha/2$

On se propose d'utiliser R pour calculer des valeurs intermédiaires non données par la table B.

Développez une procédure pour calculer les valeurs de ϵ_α pour α entre 0.01 et 0.02 par pas de 0.001 (11 valeurs à arrondir à 4 décimales) En d'autres termes, il faut trouver l'ensemble des commandes R (éventuellement une seule commande) permettant d'obtenir les données manquantes du tableau suivant:

α	0.010	0.011	0.012	0.013	0.014	0.015	0.016	0.017	0.018	0.019	0.020
ϵ_α	...	...	...	...	...	...	...	...	...	...	...

```
## ToDo (create the appropriate vector of "alpha")
## ToDo (compute and round the values of "epsilon_alpha")
```

Même question pour α variant entre 0.001 et 0.01 par pas de 0.001 (10 valeurs à arrondir à 5 décimales)

α	0.001	0.002	0.003	0.004	0.005	0.006	0.007	0.008	0.009	0.010
ϵ_α	...	...	...	...	...	...	...	...	...	...

```
## ToDo
## ToDo
```

Fonctions à utiliser: `qnorm(...)` et `round(...)`

Hey buddy, did you save your script? Think about it now!

Exercice 3: Etude de lois normales quelconques

Cet exercice reprend une partie de l'énoncé de l'exercice 2A-2 du TD 2-A.

On s'intéresse à une population P constituée de $p_H = 40\%$ d'hommes et de $p_F = 1 - p_H = 60\%$ de femmes.

On suppose que la taille des femmes suit une loi normale d'espérance $\mu_F = 165$ cm et de variance $\sigma_F^2 = 47.8$ cm^2 et que la taille des hommes suit une loi normale d'espérance $\mu_H = 177$ cm et de variance $\sigma_H^2 = 56.6$ cm^2.

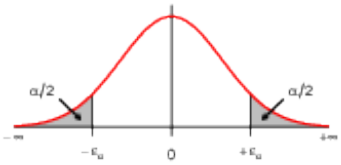
Stockez les différentes valeurs de cet énoncé sous forme de variables R:

```
# Population is composed of 40% males and 60% females
p_H <- 0.4
p_F <- 1-p_H

# Female stature is normally distributed with mean 165cm and variance 47.8
mu_F <- 165.0
sig2_F <- 47.8

# Male stature is normally distributed with mean 177cm and variance 56.6
mu_H <- 177.0
sig2_H <- 56.6
```

Table B
Valeurs critiques de la distribution normale centrée-réduite



Si Z est une variable aléatoire distribuée suivant la loi normale centrée-réduite, la table donne la valeur critique ε_α telle que :

$$\mathrm{P}\left[|Z| \geq \varepsilon_\alpha\right] = \alpha$$

α	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.00	∞	2.5758	2.3263	2.1701	2.0537	1.9600	1.8808	1.8119	1.7507	1.6954
0.10	1.6449	1.5982	1.5548	1.5141	1.4758	1.4395	1.4051	1.3722	1.3408	1.3106
0.20	1.2816	1.2536	1.2265	1.2004	1.1750	1.1503	1.1264	1.1031	1.0803	1.0581
0.30	1.0364	1.0152	0.9945	0.9741	0.9542	0.9346	0.9154	0.8965	0.8779	0.8596
0.40	0.8416	0.8239	0.8064	0.7892	0.7722	0.7554	0.7388	0.7225	0.7063	0.6903
0.50	0.6745	0.6588	0.6433	0.6280	0.6128	0.5978	0.5828	0.5681	0.5534	0.5388
0.60	0.5244	0.5101	0.4959	0.4817	0.4677	0.4538	0.4399	0.4261	0.4125	0.3989
0.70	0.3853	0.3719	0.3585	0.3451	0.3319	0.3186	0.3055	0.2924	0.2793	0.2663
0.80	0.2533	0.2404	0.2275	0.2147	0.2019	0.1891	0.1764	0.1637	0.1510	0.1383
0.90	0.1257	0.1130	0.1004	0.08784	0.07527	0.06271	0.05015	0.03761	0.02507	0.01253

Table calculée à l'aide de la fonction qnorm de R

Exemples de lecture :

$$\varepsilon_{5\%} = \varepsilon_{0.00+0.05} = 1.960 \qquad \text{et} \qquad \varepsilon_{10\%} = \varepsilon_{0.10+0.00} = 1.645$$

Table pour les petites valeurs de α

α	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}
ε_α	3.29053	3.89059	4.41717	4.89164	5.32672	5.73073	6.10941

Table calculée à l'aide de la fonction qnorm de R

Remarques :

$$\mathrm{P}\left[Z < -\varepsilon_\alpha\right] = \frac{\alpha}{2} \qquad \mathrm{P}\left[-\varepsilon_\alpha < Z < +\varepsilon_\alpha\right] = 1 - \alpha \qquad \mathrm{P}\left[Z > +\varepsilon_\alpha\right] = \frac{\alpha}{2} \qquad \mathrm{P}\left[Z < +\varepsilon_\alpha\right] = 1 - \frac{\alpha}{2}$$

Exemples :

- Pour $\alpha = 5\%$:

$$\mathrm{P}\left[Z < -1.96\right] = 2.5\% \qquad \mathrm{P}\left[-1.96 < Z < +1.96\right] = 95\% \qquad \mathrm{P}\left[Z > +1.96\right] = 2.5\%$$

- Pour $\alpha = 10\%$:

$$\mathrm{P}\left[Z < -1.645\right] = 5\% \qquad \mathrm{P}\left[-1.645 < Z < +1.645\right] = 90\% \qquad \mathrm{P}\left[Z > +1.645\right] = 5\%$$

Figure 2: Table B: valeurs critiques de la distribution normale centrée-réduite

Avez-vous remarqué? On peut mettre des commentaires dans un script R. Très utile pour améliorer la lisibilité et la compréhension des commandes. Il suffit d'utiliser le caractère dièse `#` (certains disent 'hashtag'...) : tout ce qui suit sur la même ligne sera considéré comme un commentaire par R, c'est-à-dire ignoré.

Nous vous conseillons de prendre l'habitude de commenter vos scripts au fur et à mesure que vous les écrivez. N'hésitez pas non plus à laisser des lignes blanches pour bien séparer les blocs logiques de vos calculs. N'écrivez pas de lignes trop longues dans un script. Allez à la ligne tous les 80-100 caractères.

Dernier conseil: prenez l'habitude de commenter vos scripts en anglais. Vous éviterez ainsi le problème de codage des lettres accentuées.

N.B.: Lorsque vous sauvegardez un script dans RStudio, il est possible qu'une fenêtre vous demande de choisir un "encoding", c'est-à-dire un standard de codage des caractères. Cela se produit généralement lorsque vous avez utilisé des lettres accentuées dans votre script. Vous pouvez choisir 'UTF-8'. Inversement, si, lorsque vous ouvrez un script, des caractères étranges s'affichent, c'est que RStudio n'a pas utilisé le bon "encoding" pour décoder le fichier. Dans ce cas, essayez le menu `File > Reopen with encoding...`

1- Tracé des densités de probabilité de la taille d'une femme, d'un homme et d'un individu quelconque de la population P

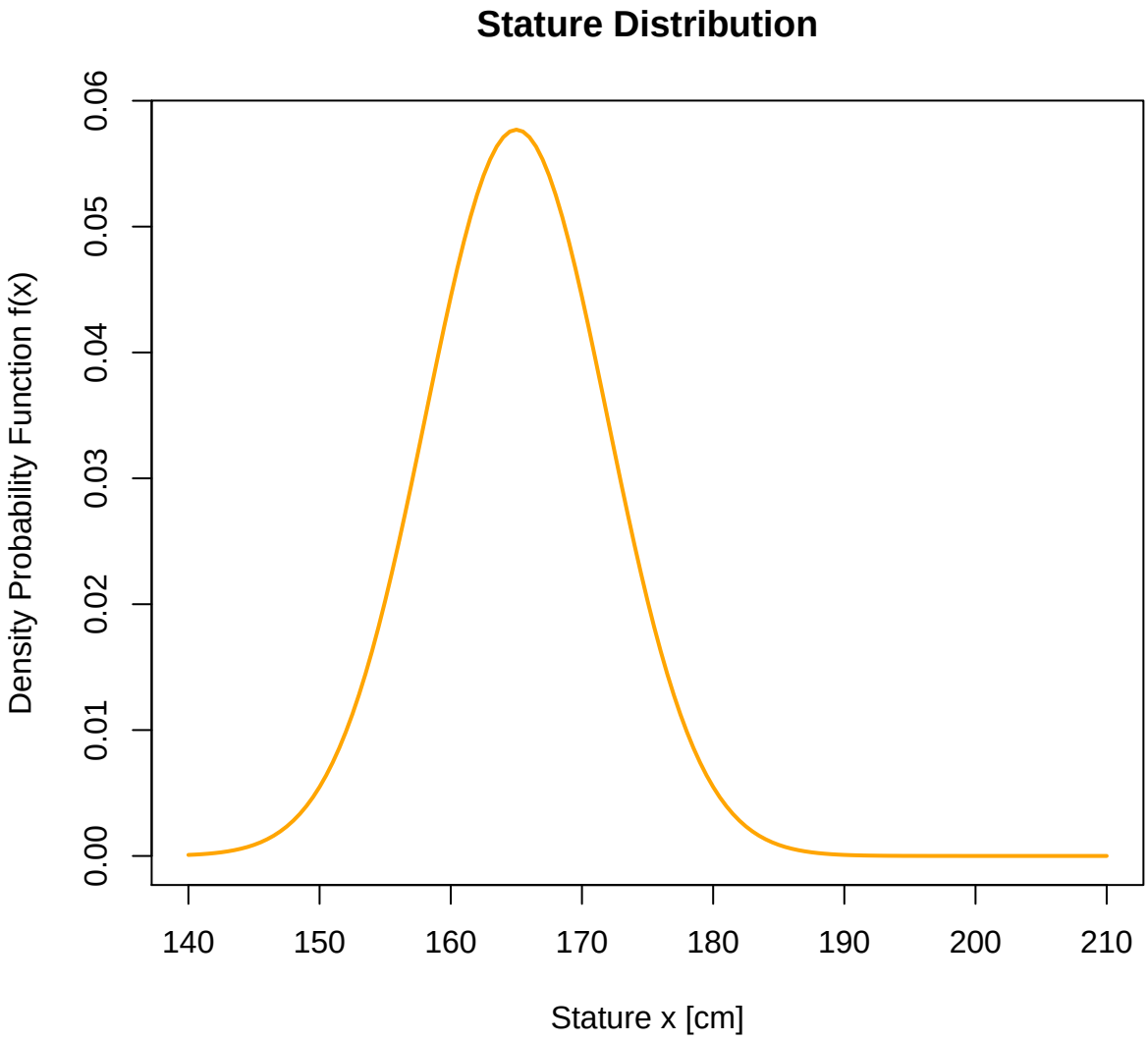
Jusqu'à présent, nous avons utilisé la fonction `dnorm(...)` avec un seul paramètre, `x`, pour calculer la fonction de densité de la distribution normale centrée-réduite. En réalité, on peut ajouter deux arguments supplémentaires, pour calculer la fonction de densité d'une distribution normale quelconque.

- `mean`: l'espérance de X,
- `sd`: l'écart-type de X.

En utilisant la fonction `dnorm(...)` et une procédure similaire à celle présentée dans l'exercice 2, tracez en orange la courbe de la fonction de densité de probabilité de la taille d'une femme pour des tailles variant entre 140 cm et 210 cm.

```
## ToDo (hint: see Exo-2-2)
```

Vous devez obtenir le graphique suivant:



Superposez sur ce graphique la courbe de la fonction de densité de probabilité de la taille d'un homme et celle de la taille d'un individu quelconque de la population P.

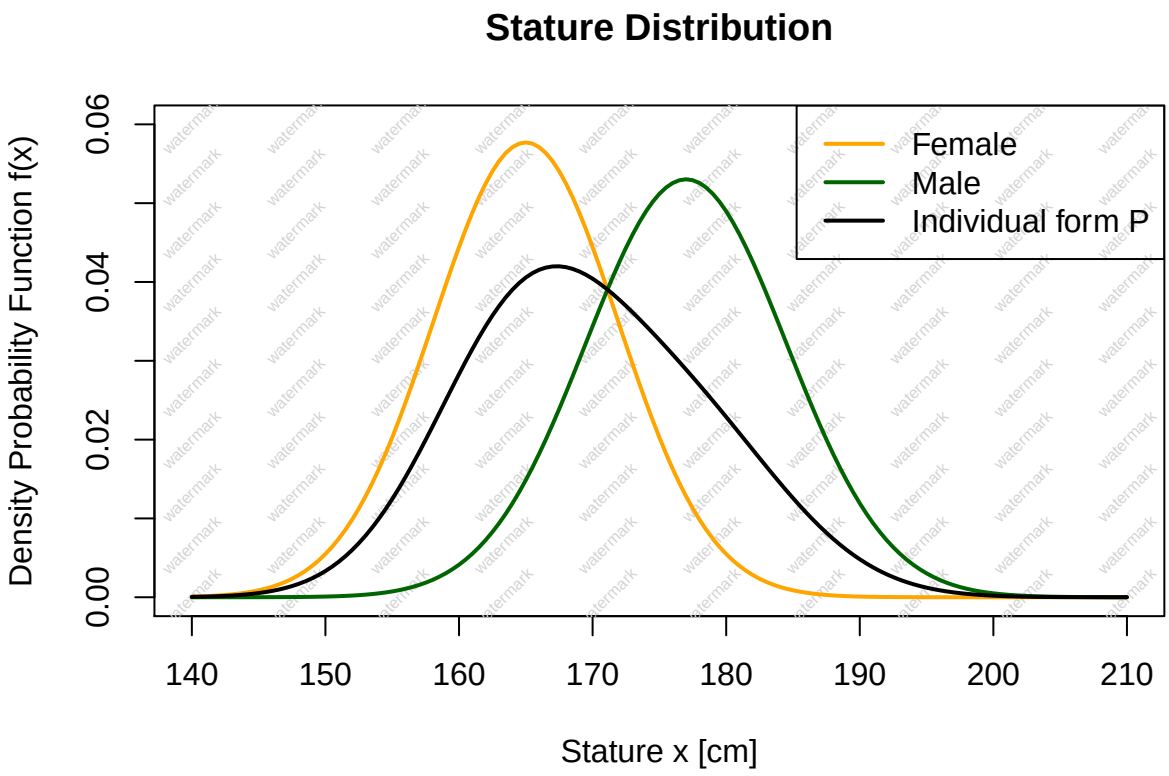
```
ValX <- seq(from=140, to=210, by=0.5)

# Probability Density Function of stature for a random male from population P
VectDensHom <- dnorm(x=ValX, mean=mu_H, sd=sqrt(sig2_H))
lines(ValX, VectDensHom, col="darkgreen", lwd=2)

# Probability Density Function of stature for a random individual from population P
VectDens <- p_F*dnorm(x=ValX, mean=mu_F, sd=sqrt(sig2_F)) + p_H*dnorm(x=ValX, mean=mu_H, sd=sqrt(sig2_H
lines(ValX, VectDens, col="black", lwd=2)

# Add a legend to the plot
legend("topright", legend=c("Female", "Male", "Individual form P"), col=c("orange", "darkgreen", "black
```

Vous devez obtenir le graphique suivant:



Une fois le graphique généré dans la fenêtre 'Plots' de RStudio, exportez-le sous format PNG vers un fichier nommé `Stature_PDF.png`. NB: Vous ne devez pas reproduire la marque "watermark" grise.

Dans le cadre de ce TP, il n'est pas nécessaire de produire une commande R pour cette tâche: utilisez la procédure manuelle et interactive décrite dans le fichier `Sauvegarde_Plot_RStudio.pdf` présent sur Moodle.

2- Calcul de probabilités

Comme pour la fonction `dnorm(...)`, la fonction `pnorm(...)` accepte les paramètres spécifiant l'espérance et l'écart-type d'une distribution normale quelconque: `mean` et `sd`.

On souhaite calculer la probabilité qu'une femme de la population P mesure entre 173.3 cm et 180.0 cm:

$$\mathbb{P}_F(173.3 < X < 180)$$

- Calculez cette probabilité en ajustant les paramètres `mean` et `sd` de la fonction `pnorm(...)` pour qu'ils correspondent à ceux d'une loi normale $\mathcal{N}(\mu = \mu_F, \sigma^2 = \sigma_F^2)$.

```
## ToDo
```

- Recalculez cette probabilité en utilisant la fonction `pnorm(...)` sans utilisation des paramètres `mean` et `sd`.

```
## ToDo
```

On souhaite maintenant calculer la probabilité qu'un individu pris au hasard dans la population P soit plus petit que 180 cm:

$$\mathbb{P}(X < 180)$$

- Calculez cette probabilité en utilisant la fonction `pnorm` et les proportions `p_F` et `p_H`:

```
## ToDo
```

Quelle est la probabilité qu'un individu pris au hasard dans la population P soit plus grand que vous?

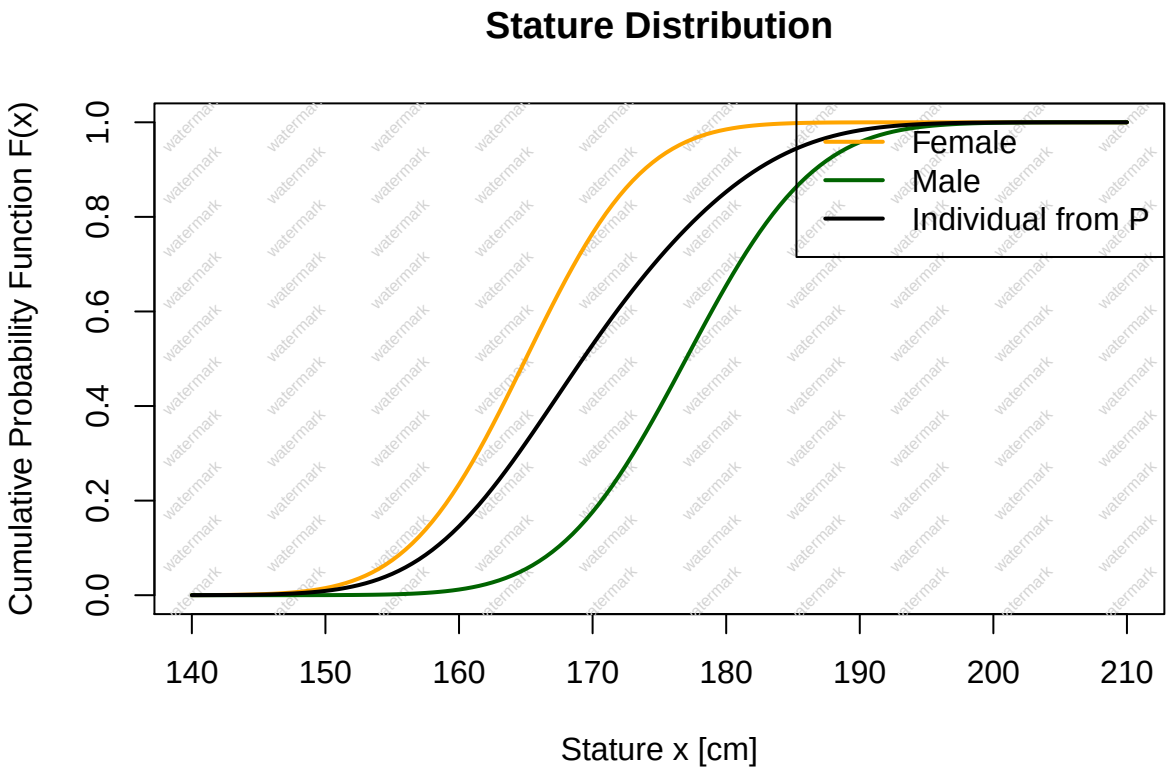
```
## ToDo
```

3- Tracé des fonctions de répartition de la taille d'un femme, d'un homme et d'un individu quelconque de la population P

En utilisant la fonction `pnorm(...)` et une procédure similaire à celle utilisée pour les fonctions de densité, tracez sur un même graphique les courbes des fonctions de répartition de la taille d'une femme, d'un homme et d'un individu de P pour des tailles variant entre 140 cm et 210 cm.

```
## ToDo (hint: see Exo-3-1)
```

Vous devez obtenir le graphique suivant:



Une fois le graphique généré dans la fenêtre 'Plots' de RStudio, exportez-le sous format PNG vers un fichier nommé `Stature_CDF.png`. NB: Vous ne devez pas reproduire la marque "watermark" grise.

Dans le cadre de ce TP, il n'est pas nécessaire de produire une commande R pour cette tâche d'exportation: utilisez la procédure manuelle et interactive décrite dans le fichier `Sauvegarde_Plot_RStudio_2020.pdf` présent sur Moodle.

4- Calcul de quantiles

Calculez le 80e percentile de la distribution de la taille d'un femme et celui de la taille d'un homme, c'est-à-dire les valeurs x_{F80} et x_{H80} telles que:

$$\mathbb{P}_F(X < x_{F80}) = 0.80$$

```
## ToDo
```

et

$$\mathbb{P}_H(X < x_{H80}) = 0.80$$

```
## ToDo
```

Pour ces calculs, utilisez la fonction `qnorm` pour la répartition d'une loi normale $\mathcal{N}(\mu_F, \sigma_F^2)$ ou $\mathcal{N}(\mu_H, \sigma_H^2)$ puis pour la répartition d'une loi normale centrée-réduite $\mathcal{N}(\mu = 0, \sigma^2 = 1)$.

Contributions

- conception des exercices: A. Badel, L. Regad, B. Toupance
- rédaction: B. Toupance