

TDU.

Introduction à R pour les enquêtes de santé publique

Cahier d'exercices pour le jour 2



Public Health
Agency of Canada

Agence de la santé
publique du Canada

Canada

Contenu

Remerciement	2
Exercice de mise en pratique	2
Instructions	2
Introduction	3
1. Configurez votre espace de travail.....	5
2. Chargez les paquets.....	8
3. Chargez les données.....	9
4. Nettoyez les données :	11
5. Visualisez les données	18
6. Première tâche bonus.....	38
7. Deuxième tâche bonus	42
8. Créez un rapport automatisé	47

Remerciement

Le jour 2 repose sur du matériel conçu en collaboration avec la spécialiste en la matière Emma Cumming du Service de la santé publique du Canada. L'Unité d'apprentissage et de perfectionnement exprime sa reconnaissance envers ses importantes contributions à ce cours.

Exercice de mise en pratique

Instructions

Les apprenants et apprenantes ont un scénario, des questions et des tâches avec le code associé pour effectuer chaque tâche. Nous recommandons aux :

Utilisateurs et utilisatrices novices (maîtres d'équipage) : Utilisez ce cahier d'exercices ainsi que le ou les scripts R fournis par le biais de GitHub. En vous servant de ce cahier d'exercices comme guide, exécutez le code que nous vous avons fourni étape par étape pour comprendre ce à quoi chaque partie du code sert et ce que font les différentes fonctions. À ce stade-ci, ne vous préoccupez pas de rédiger ou de corriger le code.

Utilisateurs et utilisatrices de niveau débutant/intermédiaire (seconds capitaines et secondes capitaines) : Le code R est fourni sous forme de capture d'écran dans ce cahier d'exercices. Vous devriez avoir une assez bonne compréhension du codage pour avoir une

idée générale du code en le lisant (avec la documentation d'aide, quelques recherches Google au besoin et les commentaires se trouvant dans les scripts fournis par l'entremise de GitHub). Notre but est de vous faire écrire le code à partir du guide à mesure que vous avancez dans le scénario. Consultez les scripts fournis dans GitHub si vous rencontrez des problèmes.

Utilisateurs et utilisatrices avancés (capitaines au long cours) : Nous vous encourageons à essayer d'écrire votre propre code où vous voulez ainsi qu'à le comparer avec le code utilisé pour l'exercice ainsi qu'à aider vos pairs si des questions surviennent. Consultez les scripts fournis dans GitHub si vous rencontrez des problèmes.

Avancez autant que vous le pouvez dans cet exercice en deux heures et cinquante minutes (maximum). Ne vous en faites pas si vous avez besoin de plus de temps. La courbe d'apprentissage pour R est abrupte et les apprenants ainsi que les apprenantes bénéficieront encore plus du temps consacré à la mise en pratique. Nous planifierons un webinaire de suivi d'une durée d'une heure après le cours pour faire le point sur les exercices (date à venir). Avant cela, si vous avez besoin d'aide avec le matériel de cours, contactez les personnes responsables de l'animation de votre cours par le biais de Slack ou par courriel.

Introduction

Une mobilisation a lieu et vous faites partie de l'intervention dans le cadre d'une éclosion de TB touchant les membres des Premières Nations à l'intérieur et à l'extérieur d'une réserve. La réserve touchée (population de 2 000 personnes) se trouve dans une région nordique éloignée, la ville la plus proche étant à 25 km (population mixte de colons et de Premières Nations, population de 7 500 personnes). Les cas de TB en dehors de la réserve sont tous liés à une maison de chambres dans la ville voisine. Les autorités sanitaires locales vous ont fourni des données à nettoyer pour analyse. Le site souhaite que vous enregistriez tous votre code et vos étapes dans un fichier R Markdown, afin que l'analyse puisse être répétée après votre départ, au besoin. Votre rapport R Markdown sera « rendu » ou exporté en un document Word.

Il existe trois façons de rendre des rapports statistiques dans R Markdown : en format .html; en format .pdf; et en format .docx (document Word). Pour cet exercice, vous produirez un rapport au format Word. Les avantages principaux de ce format sont que la plupart des personnes connaissent bien Word et que les collaborateurs et collaboratrices pourront commenter/modifier votre document après que vous ayez publié le rapport. Pour rendre un document R Markdown dans Word, l'installation de certains paquets (*packages*) ayant été « développés pour faciliter la production de documents Word ainsi que de présentations PowerPoint à partir de et avec R » peut être très utile. Ces paquets fonctionnent bien avec les paquets « **tidyverse** » tels que « **tidyr** » et « **ggplot2** ».

- « **officer**¹ » : permet de générer des documents Word ou PowerPoint avec R Markdown
- « **officetdown**² » : facilite le formatage de documents Microsoft Word produits par des documents R Markdown, incluant la mise en forme de paragraphes, de sections ainsi que de tableaux, de références et de légendes.
- « **Flextable**² » : permet de créer facilement de beaux tableaux pour les rapports.

¹ <https://ardata-fr.github.io/officeverse/>

1. Configurez votre espace de travail

Commencez par organiser votre espace de travail. Si vous ne l'avez pas déjà fait, créez de nouveaux dossiers sur votre ordinateur pour organiser vos documents pour les exercices du jour 2 comme vous l'avez fait pour ceux du jour 1 :

1. À l'intérieur du dossier « IntroToR » que vous avez créé au jour 1, créez un sous-dossier pour le jour 2 et nommez-le « Exercise_Day2 ».
2. À l'intérieur du dossier « Exercise_Day2 », créez les nouveaux sous-dossiers suivants : « output », « data » et « scripts ».
3. Déplacez les fichiers pour le jour 2 à partir de GitHub vers leurs dossiers respectifs. (Rappel : Vous pouvez créer vos propres scripts ou travailler à partir de ceux que nous vous avons fourni dans GitHub)

Notez que le respect de cette méthode d'organisation des dossiers et de création de nouveaux scripts vous permettra de réussir la dernière étape de cet exercice de mise en pratique durant laquelle un rapport automatisé devra être créé à l'aide de R Markdown.

Dans RStudio :

L'un des avantages de configurer un fichier « .RProj » est qu'il nous permet de reprendre où on s'est arrêté lors des exercices du jour 1. Cela signifie que notre répertoire de travail, notre historique et notre environnement seront déjà configurés pour nous lorsque nous ouvrirons le fichier de projet. Génial, n'est-ce pas?

Maintenant que vous avez organisé vos dossiers de projets :

Tâche	Code
Ouvrez votre projet R (« IntroToR.Rproj ») en double cliquant sur le fichier de projet que vous avez créé hier ou en ouvrant RStudio puis en cliquant sur « File », puis sur	

<p>« Open Project » pour ensuite sélectionner le fichier de projet que vous avez créé au jour 1.</p>	
<p>Si votre environnement de travail contient toujours des objets du jour 1, nettoyez-le pour pouvoir recommencer à zéro avec le jour 2 et éviter toute confusion. Sélectionnez « Session », puis « Clear Workspace » à partir de la fenêtre RStudio ou exécutez la commande suivante dans la console (moins optimal) :</p>	<pre>rm(list = ls())</pre>
<p>Ouvrez un nouveau script et sauvegardez le sous le nom de « 01_1_define_path.R » dans votre dossier de scripts.</p>	
<p>Si vous ne l'avez pas déjà fait pendant l'exercice du jour 1, installez le paquet « here ». L'installation des paquets ne doit être effectuée qu'une fois.</p>	<pre>install.packages("here")</pre>

Créez un objet qui dirigera R vers le dossier dans lequel toutes vos données brutes sont sauvegardées. Exécutez l'énoncé.	<code>data_folder <- here::here("Exercise_Day2", "data")</code>
Créez un objet qui dirigera R vers le dossier dans lequel toutes les figures ou données coupées seront sauvegardées. Exécutez l'énoncé.	<code>output_folder <- here::here("Exercise_Day2", "output")</code>
Créez un objet qui dirigera R vers le dossier dans lequel vous sauvegarderez tous vos scripts R associés à ce projet. Exécutez l'énoncé.	<code>scripts_folder <- here::here("Exercise_Day2", "scripts")</code>
Assurez-vous d'inclure un titre dans votre script avec des détails concernant le but, l'auteur, la date, les modifications et d'autres notes pertinentes.	
Sauvegardez votre script en tant que « 01_1_define_paths.R ».	

2. Chargez les paquets

Chargez les paquets dont vous aurez besoin pour ce projet :

Tâche	Code	Bibliothèque (<i>Library</i>) – information pour référence
Ouvrez un nouveau script et sauvegardez-le dans votre dossier « Exercise_Day2/scripts » sous le nom de « 01_2_load_libraries.R ».		
Installez les paquets qui ne sont pas déjà installés dans R Studio sur votre ordinateur. ***Veuillez noter que cela ne doit être fait qu'une fois et que ce n'est pas nécessaire pour les paquets installés au jour 1.	<pre>install.packages("igraph") install.packages("tidygraph") install.packages("ggraph") install.packages("flextable") install.packages("incidence") install.packages("officer") install.packages("officedown")</pre>	igraph : https://igraph.org/r/ (EN) tidygraph : https://tidygraph.data-imaginist.com/ (EN) ggraph : https://ggraph.data-imaginist.com/ (EN) flextable : https://davidgohel.github.io/flextable/ (EN) incidence : https://www.repidemicsconsortium.org/incidence/ (EN)

Introduction à R – Cahier d'exercices du jour 2

Chargez les paquets installés dont vous aurez besoin pour votre projet chaque fois que vous les utiliserez.	<pre>library(here) library(readr) # for reading csv files library(readxl) # for reading excel files library(tidyverse) library(scales) library(padr) library(writexl) library(fs) library(RColorBrewer) library(ggrepel) library(ggpubr) library(zoo) library(igraph) # need this to do social network analysis with tidygraph library(tidygraph) # for social network analysis in tidyverse language library(ggraph) # for plotting library(flextable) # makes lovely formattable tables library(viridis) library(incidence) # R epidemics consortium package for epicurves library(officer) library(officedown) library(lubridate) # handles dates</pre>	officer : https://davidgohel.github.io/officer/ (EN) officedown : https://davidgohel.github.io/officedown/ (EN)
Assurez-vous d'inclure un titre dans votre script avec des détails concernant le but, l'auteur, la date, les modifications et d'autres notes pertinentes.		
Sauvegardez votre script en tant que « 01_2_load_libraries.R » .		

3. Chargez les données

Chargez les données dont vous aurez besoin dans RStudio :

Tâche	Code
Ouvrez un nouveau script et sauvegardez-le dans votre dossier « Exercise_Day2/scripts » sous le nom de « 01_3_load_data.R ».	
Chargez les données requises pour cette analyse. Expliquez dans vos propres mots ce que le code à droite permet de faire. Quelle est l'utilité des arguments « trim_ws », « col_names » et « na » dans la fonction « read_excel() »? Indice : Si vous avez des doutes, faites une recherche avec la fonction « read_excel() » dans les fichiers d'aide.	<pre>cases <- read_excel(here("Exercise_Day2","data", "tb_cases.xlsx"), trim_ws = TRUE, col_names = TRUE, na = "Unknown") contacts <- read_excel(here("Exercise_Day2","data","tb_contacts.xlsx"), trim_ws = TRUE, col_names = TRUE, na = "Unknown")</pre>
Exécutez la commande « utils::View(cases) » dans la console. Remarque : Le « V » de « View » est en	<pre>utils::view(cases) utils::view(contacts)</pre>

majuscule. Que se passe-t-il? Comment peut-on la comparer avec la fonction « view() »? Selon vous, laquelle de ces commandes vous est la plus utile lorsque vous effectuez une révision des données pour le nettoyage?	
Assurez-vous d'inclure un titre dans votre script avec des détails concernant le but, l'auteur, la date, les modifications et d'autres notes pertinentes.	
Sauvegardez votre script en tant que « 01_3_load_data.R ».	

Ressource : <https://rveryday.wordpress.com/2016/11/29/examine-a-data-frame-in-r-with-7-basic-functions/> (EN)

Notez que ces fonctions ne font pas partie de « tidyverse ».

4. Nettoyez les données :

Maintenant que les données sont chargées dans R Studio, vous devez les nettoyer et les traiter.

Tâche	Code
-------	------

<p>Ouvrez un nouveau script et sauvegardez-le dans votre dossier « Exercise_Day2/scripts » sous le nom de « 02_1_clean_data.R ».</p>	
<p>D'abord, observez les variables dans les tableaux de données de vos cas et contacts en utilisant la fonction « str ». Cette dernière montre la structure de votre tableau de données et inclut les informations suivantes : nombre de rangées et de colonnes, nom des colonnes, classe de chacune des colonnes (type de données stockées – c.-à-d. caractère, numérique, etc.) ainsi que les quelques premières observations sur chacune des variables. Observez les types de variables dans vos ensembles de données de cas et de contacts. Quelles variables doivent être converties à partir de chaîne?</p>	<pre>str(cases) str(contacts)</pre>
<p>Convertissez les variables de texte en chaîne dans l'ensemble de données des cas en variables de facteurs (<i>factor</i>). Celles-ci peuvent être utilisées pour représenter des données</p>	<pre>cases[sapply(cases, is.character)] <- lapply(cases[sapply(cases, is.character)], as.factor)</pre>

catégoriques (ordonnées ou désordonnées). La conversion facilitera les activités de traçage dans le cadre de cet exercice.	
Convertissez les variables de texte en chaîne dans les ensembles de données des contacts en variables de facteurs.	<pre>contacts[sapply(contacts, is.character)] <- lapply(contacts[sapply(contacts, is.character)], as.factor)</pre>
Regardez les niveaux que vous venez de créer en utilisant la fonction « sapply » et en précisant les options de niveau. Remarque : Les variables qui ne sont pas classées en tant que facteurs se verront attribuer le niveau « NULL ».	<pre>sapply(cases, levels) sapply(contacts, levels)</pre>
Créez une nouvelle variable qui indiquera le niveau d'infectiosité des cas selon les variables suivantes et nommez-la « Infectiousness » : « Tb_type », « Cavitation » et « Smear2 ». Fiez-vous aux critères suivants afin d'attribuer les niveaux d'infectiosité	<pre>cases <- cases %>% mutate(Infectiousness = case_when(Tb_type == "Non-respiratory" ~ "Low", # Logic: If TB type is non-respiratory, infectiousness is low Cavitation == "Cavities" & Smear2 == "Positive" ~ "Very High", # Logic: If case has cavities and is smear positive, infectiousness is very high Cavitation == "No cavities" & Smear2 == "Positive" ~ "High", # Logic: If case has no cavities but is smear positive, infectiousness is high Smear2 == "Negative" & Tb_type == "Respiratory" ~ "Moderate"), # Logic: If case is smear negative but respiratory TB, case is moderately infectious Infectiousness = factor(Infectiousness, levels = c("Low", "Moderate", "High", "Very High")))</pre>

<p>« Low », « Moderate », « High » et « Very high »²:</p> <ul style="list-style-type: none"> - Si le type de TB est non respiratoire (« Non-respiratory »), le degré d'infectiosité (« Infectiousness ») sera faible (« Low »). - Si le cas a un frottis négatif (« smear negative ») mais que le type de TB est respiratoire (« Respiratory »), le degré d'infectiosité (« Infectiousness ») sera modéré (« Moderate »). - Si le cas n'a pas de cavités (« No cavities ») mais qu'il a un frottis positif (« smear positive »), « Infectiousness » de « High ». - Si le cas a des cavités (« Cavities ») et qu'il a un frottis positif (« smear positive »), le degré d'infectiosité (« Infectiousness ») sera très élevé (« Very high »). 	
--	--

² Afin de réviser, consultez la section 30 intitulée *Conditional Operations* du site de *R for Epidemiology* (en anglais seulement).

<p>Comme une instruction IF, les arguments sont évalués en ordre, vous devez donc procéder du plus précis au plus général.</p> <p>Cette création de variables utilise la fonction « mutate » de « dplyr » et la fonction « case_when() ». Les valeurs que vous voulez donner à votre nouvelle variable devraient se voir attribuer le symbole suivant : « ~ ».</p> <p>La fonction « factor » nous permet d'établir l'ordre des niveaux de variables. Pourquoi pensez-vous que l'on établit l'ordre de niveaux des variables désordonnées?</p>	
<p>Vérifiez si les niveaux sont définis correctement. Pour ce faire, vous pouvez utiliser la fonction « group_by » et créer un tableau. Regroupez les données selon la nouvelle variable que vous venez de créer (« Infectiousness ») et n'oubliez pas d'indiquer le</p>	<pre>cases %>% group_by(Infectiousness) %>% count(Tb_type, Cavitation, Smear2)</pre>

<p>nombre de variables que vous voulez inclure dans votre vérification (« Tb_type », « Cavitation », et « Smear2 »). Est-ce que « Infectiousness » a été correctement catégorisée?</p>	
<p>Créez une nouvelle variable affichant la date du diagnostic sous forme d'année et de mois et nommez-la « Diagnosis_month ».</p> <p>Remarque : Dans la console, essayez de vérifier votre travail en créant un tableau à double entrée avec les variables « Diagnosis_month » et « Diagnosis_date » en utilisant la fonction de tableau.</p> <p>Remarque : Cette dernière est une fonction de base de R, elle ne provient pas de « tidyverse ». Ainsi, vous devrez ajouter un nom pour le tableau de données et le signe « \$ » avant chacune des variables que vous incluez dans votre code (p. ex. « table(cases\$Diagnosis_month », « cases\$Diagnosis_date »).</p>	<pre>cases <- cases %>% mutate(Diagnosis_month = as.yearmon(Diagnosis_date))</pre>

<p>Utilisez les fonctions « mutate » et « case_when » pour créer une nouvelle variable pour contenir les groupes d'âge suivants pour les contacts et nommez-la « Agegroup » :</p> <p>« Less than 10 years »</p> <p>« 10-19 years »</p> <p>« 20-39 years »</p> <p>« 40-59 years »</p> <p>« 60+ years »</p>	<pre>contacts <- contacts %>% mutate(Agegroup = case_when(Contact_age_years >= 60 ~ "60+ years", Contact_age_years >= 40 & Contact_age_years <= 59 ~ "40-59 years", Contact_age_years >= 20 & Contact_age_years <= 39 ~ "20-39 years", Contact_age_years >= 10 & Contact_age_years <= 19 ~ "10-19 years", Contact_age_years <= 9 ~ "Less than 10 years"), Agegroup = factor(Agegroup, levels = c("Less than 10 years", "10-19 years", "20-39 years", "40-59 years", "60+ years"))</pre>
<p>Créez une nouvelle variable pour contenir les groupes d'âge des cas en utilisant les mêmes groupes d'âge que ceux utilisés pour les contacts et nommez-la « Agegroup ».</p>	<pre>cases <- cases %>% mutate(Agegroup = case_when(Age_years >= 60 ~ "60+ years", Age_years >= 40 & Age_years <= 59 ~ "40-59 years", Age_years >= 20 & Age_years <= 39 ~ "20-39 years", Age_years >= 10 & Age_years <= 19 ~ "10-19 years", Age_years <= 9 ~ "Less than 10 years"), Agegroup = factor(Agegroup, levels = c("Less than 10 years", "10-19 years", "20-39 years", "40-59 years", "60+ years"))</pre>
<p>Vérifiez si vous avez correctement classé vos groupes d'âge en utilisant la fonction « table ». Remarque : Cette dernière est une fonction de base de R, elle ne provient pas de « tidyverse ». Ainsi vous devrez ajouter un nom pour le tableau des données et le signe « \$ » avant chacune des</p>	<pre>table(contacts\$Contact_age_years, contacts\$Agegroup) table(cases\$Age_years, cases\$Agegroup)</pre>

variables que vous incluez dans votre code (c.-à-d. « <code>cases\$Diagnosis_years</code> »).	
Assurez-vous d'inclure un titre dans votre script avec des détails concernant le but, l'auteur, la date, les modifications et d'autres notes pertinentes.	
Sauvegardez votre script en tant que « <code>02_1_clean_data.R</code> ».	

5. Visualisez les données

Tracez les données des cas selon la caractéristique de temps. En prime (si vous avez encore du temps)! Essayez d'explorer des thèmes différents.

Tâche	Code
Ouvrez un nouveau script et sauvegardez-le dans votre dossier « <code>Exercise_Day2/scripts</code> » sous le nom de « <code>03_1_plot_case_time.R</code> ».	

Tracez les données des cas par mois du diagnostic.

D'abord, résumez (« summarise ») les données des cas dans un tableau dénombrant le nombre de cas selon le mois de leur diagnostic, puis redirigez ce tableau de fréquence dans une commande « GGLOT » pour obtenir un diagramme à barres dont l'axe de x est « Diagnosis Month », et l'axe des y, « Count ». Ajustez votre tableau selon les consignes suivantes :

- Établissez la hauteur des barres pour qu'elles correspondent aux valeurs des données plutôt que leur nombre (à l'aide de « stat="identity" »).
- Utilisez un format de mois et d'année, c.-à-d. « Aug 2020 ».
- Utilisez le thème minimal (« theme_minimal() »).
- Ajoutez les étiquettes suivantes : pour l'axe des x, « Month and Year of Diagnosis »; et pour l'axe des y, « Case Count ».

Ressource supplémentaire pour de l'aide dans le formatage des dates :

<https://www.r->

```
cases %>%
  group_by(Diagnosis_month) %>%
  summarise(Count = n()) %>%

  ggplot(aes(x=as.Date(Diagnosis_month), y=Count)) +
  geom_bar(stat="identity")+
  scale_x_date(date_labels = "%b %Y") +
  theme_minimal() +
  ggtitle("TB case diagnoses, May-October 2013 ") +
  ylab("Case count") + xlab("Month and Year of Diagnosis")
```

bloggers.com/2013/08/date-formats-in-r/ (EN)	
<p>Sauvegardez la figure de résultat en une image au format .jpeg dans votre dossier « output » et nommez-la « plot_cases_month ».</p> <p>Remarque : la fonction « paste0() » colle toutes les chaînes de texte que vous avez fournies, sans espaces entre elles.</p>	<pre>ggsave(filename = paste0(output_folder, "/plot_cases_month.jpeg"), width = 7, height = 4)</pre>
<p>Tracez les données de cas par mois du diagnostic et par leur genre.</p> <p>D'abord, résumez (« summarise ») les données des cas dans un tableau dénombrant les cas selon le mois de leur diagnostic et selon leur genre, puis redirigez ce tableau de fréquence dans une commande « GGLOT » pour obtenir un diagramme à barres dont l'axe de x est « Diagnosis Month », et l'axe des y, « Count ». Dans les graphiques à barres, vous pouvez utiliser des couleurs pour stratifier vos données par le biais d'une autre variable (dans ce cas, le genre) en spécifiant « fill= Gender » dans l'appel « aes() ».</p>	<pre>cases %>% group_by(Diagnosis_month, Gender) %>% summarise(Count = n()) %>% ggplot(aes(x=as.Date(Diagnosis_month), fill = Gender, y = Count)) + geom_bar(stat="identity", position = "stack") + scale_x_date(date_labels = "%b %Y") + theme_minimal() + ylab("Case count") + xlab("Month and Year of Diagnosis") + scale_fill_manual(values=c("green", "orange"))</pre>

<p>Utilisez le même formatage que le graphique précédent en incluant toutefois les changements suivants :</p> <ul style="list-style-type: none"> - Faites en sorte que vos barres de traçage soient empilées (en utilisant « position = "stack" ») plutôt qu'une à côté de l'autre (c.-à-d. « position = "dodge" »). - Ajoutez les étiquettes suivantes : pour l'axe des x, « Month and Year of Diagnosis »; pour l'axe des y, « Case Count ». <p>Précisez les couleurs des barres en attribuant le vert pour les femmes et l'orange pour les hommes à l'aide de l'option « scale_fill_manual ». L'ordre selon lequel vous énumérez ces couleurs correspondra à l'ordre de tout facteur de variable. Si vous ne connaissez pas l'ordre, vous pouvez toujours vérifier à l'aide de la fonction « levels() » et du code « levels(cases\$Gender) ».</p>	
<p>Sauvegardez la figure de résultat en une image au format .jpeg dans votre dossier « output » et nommez-la « plot_cases_month_gender ».</p>	<pre>ggsave(filename = paste0(output_folder, "/plot_cases_month_gender.jpeg"), width = 7, height = 4)</pre>

<p>Tracez les données des cas par mois du diagnostic et par leur niveau d'infectiosité.</p> <p>D'abord, résumez (« summarise ») les données des cas dans un tableau dénombrant les cas selon le mois de leur diagnostic et selon leur statut d'infectiosité, puis redirigez ce tableau de fréquence dans une commande « GGLOT » pour obtenir un diagramme à barres dont l'axe de x est « Diagnosis Month », et l'axe des y, « Count ». Établissez les couleurs des barres : « green=low »; « yellow=moderate »; « orange=high » et « red=very high ». Pour vérifier l'ordre des niveaux pour la variable « Infectiousness », entrez le code « levels(cases\$Infectiousness) ».</p> <p>Utilisez le même formatage que le graphique précédent.</p>	<pre>cases %>% group_by(Diagnosis_month, Infectiousness) %>% summarise(Count = n()) %>% ggplot(aes(x=as.Date(Diagnosis_month), fill = Infectiousness, y = Count)) + geom_bar(stat="identity", position = "stack")+ scale_x_date(date_labels = "%b %Y") + theme_minimal() + ylab("Case count") + xlab("Month and Year of Diagnosis") + scale_fill_manual(values=c("green", "yellow", "orange", "red"))</pre>
<p>Sauvegardez le résultat en une image au format .jpeg dans votre dossier « output » et nommez-la « plot_cases_month_infectiousness ».</p>	<pre>ggsave(filename = paste0(output_folder, "/plot_cases_month_infectiousness.jpeg"), width = 7, height = 4)</pre>

Introduction à R – Cahier d'exercices du jour 2

Assurez-vous d'inclure un titre dans votre script avec des détails concernant le but, l'auteur, la date, les modifications et d'autres notes pertinentes.	
Sauvegardez votre script en tant que « 03_1_plot_case_time.R ».	

Tracez les contacts selon leur démographie. En prime (si vous avez encore du temps)! Essayez d'explorer des thèmes différents.
<https://ggplot2.tidyverse.org/reference/ggtheme.html> (EN)

Tâche	Code
Ouvrez un nouveau script et sauvegardez-le dans votre dossier « Exercise_Day2/scripts » sous le nom de « 03_2_plot_contact_demographics.R ».	

Tracez les données des cas par mois du diagnostic et par leur genre.

D'abord, résumez (« summarise ») les données des cas dans un tableau de fréquence dénombrant les groupes d'âge parmi les groupes de contacts et selon le mois de leur diagnostic et selon leur genre, puis redirigez ce tableau de fréquence dans une commande *GGPLOT* dont l'axe de x est « Agegroup », « fill » est « Gender », et l'axe des y, « Count ».

Modifiez votre graphique selon les consignes de mise en page suivantes :

- Ajustez la hauteur des barres pour qu'elle soit équivalente à la valeur des données et non leur nombre (en utilisant « stat="identity" »).
- Placez vos barres côte à côte (en utilisant « Position = "dodge" »).
- Utilisez le thème minimal (« theme_minimal() »).
- Ajoutez les étiquettes suivantes : pour l'axe des x, « Age group »; et pour l'axe des y, « Contacts ».
- Déterminez les couleurs des barres : vert pour les femmes et orange pour les hommes (en utilisant l'option « scale_fill_manual »).

```
contacts %>%
  group_by(Agegroup, Gender) %>%
  summarise(Count = n()) %>%

  ggplot(aes(x= Agegroup, y = Count, fill = Gender)) +
  geom_bar(stat="identity", position = "dodge")+
  theme_minimal() +
  ylab("# Contacts") + xlab("Age group") +
  scale_fill_manual(values=c("green", "orange"))
```


Pour vérifier les niveaux « Gender », entrez le code suivant : « levels(contacts\$Gender) ».	
Sauvegardez la figure de résultat en une image au format .jpeg dans votre dossier « output » et nommez-la « plot_contacts_agegender_count ».	<code>ggsave(filename = paste0(output_folder, "/plot_contacts_agegender_count.jpeg"), width = 7, height = 4)</code>
Tracez les données des cas par groupe d'âge et selon les proportions de genre. D'abord, faites un tableau de fréquences incluant les groupes d'âge des contacts et les genres des cas. Ensuite, vous devrez les détacher (« ungroup ») pour éviter que vos proportions soient calculées dans la première variable, dans ce cas-ci, « Agegroup ». Une fois que vous les avez détachés, créez une variable pour calculer vos proportions en établissant le total en tant que dénominateur et nommez-la « Proportion ». Redirigez-les dans <i>GGPLOT</i> et nommez l'axe de x « Agegroup », et l'axe des y, « Proportion », puis indiquez « fill » suivi de « Gender ».	<pre>contacts %>% group_by(Agegroup, Gender) %>% summarise(Count = n()) %>% ungroup() %>% mutate(Proportion = round(100*Count/sum(Count),1)) %>% ggplot(aes(x= Agegroup, y = Proportion, fill = Gender)) + geom_bar(stat="identity", position = "dodge")+ theme_minimal() + ylab("% total contacts") + xlab("Age group") + scale_fill_manual(values=c("green", "orange")) + geom_text(aes(label=paste0(Proportion, "%")), position=position_dodge(width=0.9), vjust=-0.25)</pre>

<p>Utilisez le même formatage que le graphique précédent, mais effectuez les changements suivants :</p> <ul style="list-style-type: none"> - Ajoutez les étiquettes suivantes : pour l'axe des x, « Age group »; pour l'axe des y, « % total contacts ». - Ajoutez des étiquettes à chacune des barres en utilisant l'option « aes(label) ». Précisez que vous voulez afficher la « Proportion » et un signe « % ». De plus, précisez la position de vos étiquettes en utilisant « position_dodge » (pour que les étiquettes se retrouvent au-dessus des barres de votre graphique) et définissez la taille de la police de l'étiquette. 	
<p>Sauvegardez la figure de résultat en une image au format .jpeg dans votre dossier « output » et nommez-la « plots_contacts_agegender_prop ».</p>	<pre>ggsave(filename = paste0(output_folder, "/plot_contacts_agegender_prop.jpeg"), width = 7, height = 4)</pre>
<p>En prime (si vous avez encore du temps)!</p> <p>Essayez d'exclure la ligne suivante du code ci-dessus :</p> <p>« ungroup() %>% »</p> <p>Quel effet cela a-t-il sur votre graphique?</p>	

Créez deux graphiques dans une figure pour inclure les contacts par âge et par genre ainsi que s'ils résident dans la réserve ou à l'extérieur de celle-ci.

D'abord, faites un tableau de fréquences incluant les groupes d'âge des contacts et les nombres de cas selon leur genre et leur emplacement. Utilisez l'option « `.drop = FALSE` », qui permet à votre tableau sommaire d'inclure des zéros dans les tableaux et figures. Redirigez le tableau de fréquences dans *GGPLOT* et nommez l'axe des x « Agegroup » et l'axe des y, « Count », et indiquez « fill » suivi de « Gender ».

Utilisez le même formatage que le graphique précédent, mais effectuez les changements suivants :

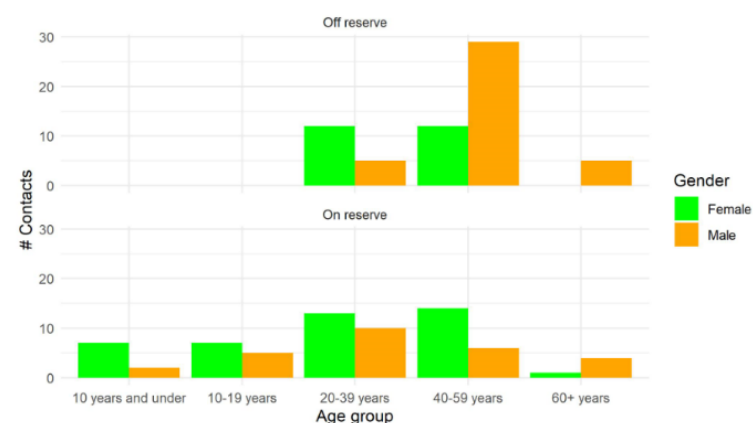
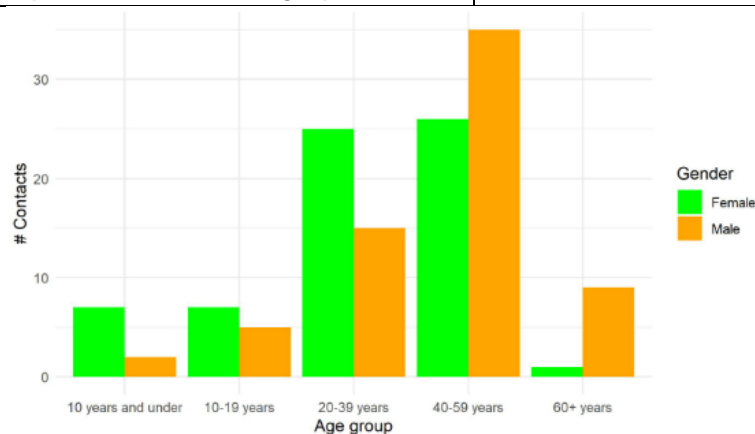
- Positionnez la barre et son tracé côte à côte tout en conservant la largeur de la barre à l'aide du code suivant : `position = « position_dodge(preserve = "single") »`.
- Ajoutez les étiquettes suivantes : pour l'axe des x, « Age group »; pour l'axe des y, « # contacts ».
- Utilisez l'option de « `facet_wrap` » pour créer un panneau fondé sur l'emplacement des contacts

```
contacts %>%
  group_by(Agegroup, Gender, Contact_location, .drop = FALSE) %>%
  summarise(Count = n()) %>%

  ggplot(aes(x= Agegroup, y = Count, fill = Gender)) +
  geom_bar(stat="identity", position = position_dodge(preserve = "single"))+
  theme_minimal() +
  ylab("# Contacts") + xlab("Age group") +
  scale_fill_manual(values=c("green", "orange")) +
  facet_wrap(vars(Contact_location), nrow =2 )
```

Introduction à R – Cahier d'exercices du jour 2

(« Contact_location »). Indiquez que vous voulez deux rangées (« nrow=2 »).	
Sauvegardez la figure de résultat en une image au format .jpeg dans votre dossier « output » et nommez-la « plots_contacts_location ».	<code>ggsave(filename = paste0(output_folder, "/plot_contacts_location.jpeg"), width = 7, height = 4)</code>
Assurez-vous d'inclure un titre dans votre script avec des détails concernant le but, l'auteur, la date, les modifications et d'autres notes pertinentes.	
Sauvegardez votre script en tant que « 03_2_plot_contact_demographics ».	



Créez un tableau de fréquences bien formaté pour le lieu des infections en utilisant « Flextables » :

Tâche	Code
<p>Ouvrez un nouveau script et sauvegardez-le dans votre dossier « Exercise_Day2/scripts » sous le nom de « 03_3_tab_case_site.R ».</p>	
<p>D'abord, résumez (« summarise ») les données des cas dans un tableau de fréquence selon le lieu d'infection. Nommez le nouveau tableau de données « case_site_infection ». Puisque nous voulons avoir le nombre de cas par lieu d'infection, vous devrez utiliser la fonction « group_by », puis « site of infection ».</p> <p>Pour afficher votre tableau, vous pouvez cliquer sur le nouveau tableau contenant vos données nommé « case_site_infection » ou inscrire le code « case_site_infection ».</p>	<pre>case_site_infection <- cases %>% group_by(site_infection) %>% summarise(Count = n())</pre>
<p>Pour ajouter une rangée de totaux, vous devez créer un tableau d'une rangée (nommé « totals_site_infection ») qui ne regroupe pas les variables pour avoir le nombre total.</p> <p>Utilisez les fonctions suivantes afin de créer votre tableau :</p> <ul style="list-style-type: none"> - « Summarise » : donne le nombre total de cas. - « Mutate » : vous permet de créer une nouvelle colonne afin de correspondre au tableau de fréquences que vous avez créé plus tôt, vous 	<pre>totals_site_infection = cases %>% summarise(Count = n()) %>% mutate(Site_infection = "Total") %>% select(Site_infection , Count)</pre>

<p>pouvez ensuite y annexer le nombre total de rangées.</p> <ul style="list-style-type: none"> - « Select » : limite les colonnes pour correspondre au tableau de fréquences créé plus tôt. <p>Pour afficher votre tableau, vous pouvez cliquer sur le nouveau tableau que vous avez créé nommé « totals_site_infection » ou inscrire le code « totals_site_infection ».</p>	
<p>Associez le tableau avec le nombre total de rangées en utilisant la fonction « rbind ».</p> <p>Remarque : La fonction « rm » est utilisée pour retirer des objets inutiles pour l'analyse. Dans cet exemple, « totals_site_infection » a été retiré puisqu'il n'était plus nécessaire après la combinaison des tableaux.</p>	<pre>case_site_infection <- rbind(case_site_infection, totals_site_infection) ; rm(totals_site_infection)</pre>
<p>Créez des « flextables » à l'aide les options suivantes :</p> <ul style="list-style-type: none"> - Définissez la couleur de l'arrière-plan de l'en-tête à gris (« #E6E6E6 ») en utilisant la fonction « bg ». - Mettez la police de l'en-tête en gras en utilisant la fonction « bold ». - Modifiez la taille de la police à 10 points dans toutes les parties du tableau en utilisant la fonction « fontsize ». - Changez le type de police pour « Arial » en utilisant la fonction « font ». - Centrez et justifiez le corps de votre texte en utilisant la fonction « align ». 	<pre>tab_case_site_infection <- flextable(case_site_infection) %>% bg(bg = "#E6E6E6", part = "header") %>% bold(part = "header") %>% fontsize(size = 10, part = "all") %>% font(part = "all", fontname = "Arial") %>% align(align = "center", part = "all") %>% set_header_labels(Site_infection = "Site of infection") %>% align(j=c("Site_infection"), align = "left") %>% bold(i = 6, bold = TRUE, part = "body") %>% width(j = 1, width = 1.5) %>% set_caption(" Site of TB infection", style = "Table Caption", autonum = "autonum")</pre>

<ul style="list-style-type: none"> - Modifiez l'en-tête de colonne « site_infection » pour « Site of Infection » en utilisant la fonction « set_header_labels ». - Alignez la première colonne à gauche (« Sight_infection ») en utilisant la fonction « align ». - Mettez en gras la police de la rangée « Total » en utilisant la fonction « bold » (Indice : c'est la sixième rangée). - Ajustez la largeur de la première colonne (« Site_infection ») à 1,5 pouce en utilisant la fonction « width ». - Ajoutez le titre : « Site of TB infection » à la figure en utilisant la fonction « set_caption ». Utilisez « autonom » pour que le titre de la légende de cette figure affiche un numéro dans Word. <p>L'article suivant offre un excellent aperçu des « flextables » et de la manière dont on peut les formater : https://davidgohel.github.io/flextable/articles/overview.html (EN).</p>	
Imprimez le tableau.	<code>tab_case_site_infection</code>
Nettoyez votre espace de travail en retirant tout objet qui n'a pas sa place dans l'analyse en utilisant la fonction	<code>rm(case_site_infection)</code>

Introduction à R – Cahier d'exercices du jour 2

« rm ». Dans cette situation, retirez « case_site_infection ».	
Assurez-vous d'inclure un titre dans votre script avec des détails concernant le but, l'auteur, la date, les modifications et d'autres notes pertinentes.	
Sauvegardez votre script en tant que « 03_3_tab_case_site.R ».	

Site of TB infection

Site of infection	Count
Abdominal	1
Meningeal	1
Miliary	1
Pleural	4
Pulmonary TB	4
Total	11

Dessinez un graphique de réseau social afin d'illustrer les relations entre les cas et les contacts. D'abord, nous devons transformer nos tableaux de données des cas et des contacts en tableaux de données de liens ou de nœuds. Il y a des milliers de façons de le faire! Le code ci-dessous n'est qu'un exemple parmi tant d'autres.

Tâche	Code
Ouvrez un nouveau script et sauvegardez-le dans votre dossier « Exercise_Day2/scripts » sous le nom de « 04_1_plot_sna.R ».	

<p>Créez un tableau de données de liens en utilisant la fonction « edge ». Cela établit la relation entre les cas et les contacts. Nous n'avons besoin que de deux variables : les identifiants des cas (« CaseID2 ») et les identifiants des contacts (« ContactID2 »).</p> <p>Renommez les colonnes ainsi : « CaseID2= "from" » et « ContactID2= "to" ».</p> <p>Réorganisez les colonnes (en utilisant « arrange ») de façon à ce que « from » soit en premier.</p>	<pre>edges <- contacts %>% select(CaseID2, ContactID2) %>% rename(from = CaseID2, to = ContactID2) %>% arrange(from,to)</pre>
<p>Créez un tableau de données de nœuds pour les données des contacts et nommez-le « nodes_a ».</p> <p>Réduisez la liste « case-contact » pour n'indiquer que les contacts uniques, excluant ainsi les cas qui sont aussi déclarés comme contacts.</p> <p>Excluez toutes les rangées qui contiennent une chaîne de texte « CASE » dans le « ContactID » (en utilisant « filter »).</p> <p>Sélectionnez les variables dont on a besoin (« ContactID2 », « Gender », « Contact_location », « Agegroup », « Contact_age_years »).</p>	<pre>nodes_a <- contacts %>% filter(!grep1("CASE", ContactID)) %>% select(ContactID2, Gender, Contact_location, Agegroup, Contact_age_years) %>% rename(ID = ContactID2, Location = Contact_location, Age_years = Contact_age_years) %>% distinct() %>% mutate(Classification = "contact")</pre>

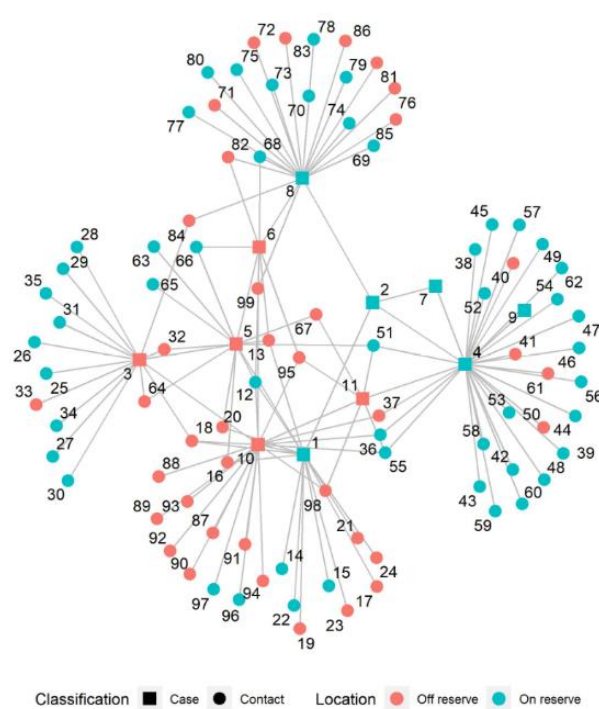
<p>Renommez les variables de manière appropriée (« ID = ContactID2 », « Location = Contact_location », « Age_years = Contact_age_years »).</p> <p>Dédupliquez (en utilisant la fonction « distinct »). Sélectionnez seulement les rangées uniques.</p> <p>Créez une nouvelle colonne (en utilisant « mutate ») et assignez à tous la classification de « Contact ».</p>	
<p>Créez un tableau de données de nœuds à partir de vos données de cas et nommez-le « nodes_b ».</p> <p>Sélectionnez les variables dont vous avez besoin (« CaseID2 », « Gender », « Location », « Agegroup », « Age_years »).</p> <p>Renommez « ID = CaseID2 ».</p> <p>Créez une nouvelle colonne (en utilisant « mutate ») et assignez à tous la classification de « Case ».</p>	<pre>nodes_b <- cases %>% select(CaseID2, Gender, Location, Agegroup, Age_years) %>% rename(ID = CaseID2) %>% mutate(Classification = "Case")</pre>
<p>Reliez les deux tableaux de données correspondants (en utilisant « rbind »). Pour ce faire, les colonnes doivent avoir les mêmes noms et être dans le même ordre.</p>	<pre>nodes <- rbind(nodes_a, nodes_b) ; rm(nodes_a, nodes_b)</pre>

<p>Retirez les tableaux de données « nodes_a » et « nodes_b », puisqu'ils ne sont plus nécessaires (en utilisant « rm »).</p> <p>Observez les tableaux de données de nœuds que vous avez créés. Sont-ils reliés correctement?</p>	
<p>Convertissez notre tableau de liens en un objet structuré « tbl_graph » en utilisant la fonction « as_tbl_graph() » de <i>tidygraph</i>. On peut ensuite y soumettre plusieurs types de données d'entrée différents comme « data.frame », « matrix », « dendrogram », « igraph », etc.</p> <p>Renommez respectivement « edges » et « nodes » par « edges_full » et « nodes_full » (pour clarifier les choses durant le codage).</p> <p>Sélectionnez les variables dont on a besoin (« ID », « Classification », « Location »).</p> <p>Classez par « ID » (en utilisant « arrange »).</p>	<pre>nodes_full <- nodes %>% select(ID, Classification, Location) %>% arrange(ID) edges_full <- edges</pre>
<p>Créez un objet de réseau <i>tidygraph</i>. Indiquez à <i>tidygraph</i> quel tableau de données correspond aux « nodes » et « edges ».</p> <p>Remarque : L'option « directed=FALSE » spécifie que les relations sont non directionnelles dans ce cas.</p>	<pre>network_full <- tbl_graph(nodes = nodes_full, edges = edges_full, directed = FALSE)</pre>

<p>Définissez les étiquettes qui apparaîtront sur le graphique.</p> <p>Dans ce cas, « ID » représente ce que l'on veut indiquer comme étiquette (en utilisant la fonction « select »).</p> <p>Essayez d'exécuter le code avec et sans l'énoncé « %>% pull() » à la fin. À quoi sert la fonction « pull() » dans cette situation selon vous?</p>	<pre>label_full <- nodes_full %>% select(ID) %>% pull()</pre>
<p>Créez un graphique de réseau (notez comme cela ressemble à l'utilisation de <i>ggplot!</i>).</p> <p>Ajoutez les modifications suivantes au formatage de votre tracé :</p> <ul style="list-style-type: none"> - Définissez la couleur des lignes connectant les liens pour gris (« bdbdbd ») (en utilisant « geom_edge_link0 »). - Définissez la couleur de « node » pour qu'elle s'ajuste en fonction de la variable « Location » et de la forme de « Classification » (en utilisant « aes »). - Définissez le degré de transparence des points (en utilisant « alpha »). - Définissez la taille des points à 4 (en utilisant « size »). - Sélectionnez votre palette de couleur (en utilisant « scale_fill_brewer »). - Ajoutez les étiquettes que vous avez choisies au courant de l'étape précédente (nommées « label_full ») et formatez les tableaux pour ne pas qu'ils se chevauchent (en utilisant « repel= TRUE »). 	<pre>network_full %>% ggraph() + geom_edge_link0(color = "#bdbdbd") + geom_node_point(aes(colour = Location, shape = Classification), alpha = 1, size = 4) + scale_fill_brewer(type = "qual", palette = 5) + geom_node_text(label = label_full, repel = TRUE, point.padding = 0.1, segment.color = NA) + theme(panel.background = element_blank(), plot.title = element_text(size = 20), legend.position = "bottom") + scale_shape_manual(values = c(15,16))</pre>

<ul style="list-style-type: none"> - Ajustez l'espacement des étiquettes selon les points en établissant une option pour l'argument « <code>point.padding</code> ». - Retirez les lignes reliant les étiquettes aux points en spécifiant « <code>NA</code> » dans l'argument « <code>segment.color</code> ». - Définissez l'arrière-plan pour qu'il soit vide (en utilisant « <code>panel.background=element(blank)</code> »). - Changez la taille de la police du titre à 20 points (en utilisant « <code>element_text(size=20)</code> »). - Déplacez la légende vers le bas (en utilisant « <code>legend.position= "bottom"</code> »). - Définissez les formes que vous voulez utiliser pour les points en utilisant « <code>scale_shape_manual</code> ». Définissez les cas en tant que carrés remplis (option 15) et les contacts en tant que cercles remplis (option 16). - Consultez le lien suivant pour plus d'options de formes : http://sape.inf.usi.ch/quick-reference/ggplot2/shape (EN) 	
<p>Sauvegardez le tracé de résultat en une image au format .jpeg dans votre dossier des extraits approprié et nommez-le « <code>plot_sna_location</code> ».</p> <p>Remarque : Nous voulons que l'image soit assez large (6"x7").</p>	<pre>ggsave(filename = paste0(output_folder, "/plot_sna_location.jpeg"), width = 6, height = 7)</pre>
<p>Assurez-vous d'inclure un titre dans votre script avec des détails concernant le but, l'auteur, la</p>	

date, les modifications et d'autres notes pertinentes.	
Sauvegardez votre script en tant que « 04_1_plot_sna ».	



6. Première tâche bonus

Tracez une courbe épidémique illustrant la semaine lors de laquelle les symptômes sont apparus incluant les niveaux d'infectiosité codés à l'aide de couleurs différentes :

Remarque : Le paquet « Incidence » ne fait pas partie de *tidyverse*. Par conséquent, il utilise les notations de la programmation de base de R (p. ex. pour travailler avec des données, vous devez nommer le tableau de données ainsi que les variables en les séparant à l'aide du signe « \$ » ou de l'argument « `df$var1` »). Cependant, il est un paquet très simple d'utilisation pour ses usagers épidémiologistes puisqu'il a été réalisé par la *R Epidemics Consortium* (RECON).

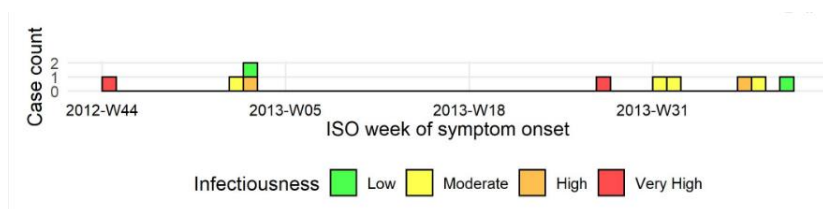
Tâche	Code
Ouvrez un nouveau script et sauvegardez-le dans votre dossier « Exercise_Day2/scripts » sous le nom de « 05_1_plot_epicurve.R ».	
Vérifiez le format de date (en utilisant « <code>class</code> »).	<pre>class(cases\$`symptom_date`) cases\$symptom_date = as.Date(cases\$symptom_date, format = "%d-%m-%Y")</pre>
<p>Ce paquet fonctionne en reconnaissant les dates en tant que « <code>Date</code> », et non « <code>POSIXct</code> » (pour en savoir plus, saisissez « <code>?POSIXct</code> » dans la console).</p> <p>Convertissez vos données vers le format de date « <code>%d-%m-%Y</code> », si nécessaire (en utilisant « <code>as.Date</code> »).</p>	
Créez un thème de format pour la courbe épidémique en utilisant le langage <i>ggplot2</i> pour effectuer les	<pre>my_theme <- theme_minimal(base_size = 12) + theme(panel.grid.minor = element_blank()) + theme(legend.position="bottom") + theme(axis.text.x = element_text(angle = 0, hjust = 0.5, vjust = 0.25, color = "black"))</pre>

<p>modifications de mise en forme suivantes :</p> <ul style="list-style-type: none"> - Utilisez le thème minimal (c.-à-d. sans arrière-plan, annotations, etc.) en utilisant « theme_minimal ». - Indiquez la taille de police de base (tous les éléments de texte du graphique) à 12 points (« base_size=12 »). - Supprimez le quadrillage secondaire. - Déplacez la légende en bas (« legend.position=bottom »). - Ajustez la hauteur et l'angle de l'axe des x et attribuez le noir comme couleur de police. 	
<p>Créez un objet d'incidence en fonction de la date d'apparition des symptômes par intervalle de 7 jours regroupé par degré d'infectiosité.</p>	<pre>i.7 <- incidence(cases\$Symptom_date, interval = 7, groups = cases\$Infectiousness)</pre>
<p>Tracez l'objet d'incidence (pour créer la courbe épidémique). Appliquez le thème que vous avez créé ci-dessus (« my_theme ») et apportez-y les modifications suivantes :</p> <ul style="list-style-type: none"> - Représentez chaque cas dans la figure par un rectangle 	<pre>plot(i.7, show_cases = TRUE, border = "black", labels_week = TRUE) + my_theme + scale_y_continuous(breaks=seq(0, 3, 1)) + scale_fill_manual(values=c("green", "yellow", "orange", "red")) + labs(x = "ISO week of symptom onset", y = "Case count", fill="Infectiousness") + coord_fixed(ratio = 7)</pre>

<p>unique (« show_cases = TRUE »).</p> <ul style="list-style-type: none"> - Assurez-vous que les étiquettes de valeur de l'axe des x reflètent les séparations selon la semaine, contrairement à quelque chose comme le 1^{er} jour du mois par exemple (« labels_Week = TRUE »). - Ajouter une bordure noire autour de chacun des cas (« border="black" »). - Déterminez l'échelle de votre axe des y (les valeurs s'étendent de 0 à 3 par 1). - Définissez la couleur de chaque niveau d'infectiosité comme indiqué ci-dessous : « green=low »; « yellow=moderate »; « orange=high »; « red=very high » (en utilisant « scale_fill_manual »). - Changez le titre de l'axe des x pour « ISO week of symptom onset » et celui de l'axe des y pour « Case count » (en utilisant la fonction « labs »). - Indiquez « Infectiousness » dans votre légende à l'aide de la fonction « label » (« fill='infectiousness' »). 	
---	--

Introduction à R – Cahier d'exercices du jour 2

- Représentez individuellement les cas par des carrés, en accordant 7 jours par carré sur la ligne des x (en utilisant « coord_fixed(ratio = 7) »).	
Sauvegardez la figure de résultat en une image au format .jpeg dans votre dossier des extrants approprié à l'aide de « ggsave » et nommez-la « plot_cases_epiweek ».	<code>ggsave(filename = paste0(output_folder, "/plot_cases_epiweek.jpeg"), width = 7, height = 2)</code>
Sauvegardez votre script en tant que « 05_1_plot_epi_curve ».	



7. Deuxième tâche bonus

Créez un tableau présentant l'emplacement des cas et des contacts à l'intérieur et à l'extérieur de la réserve.

Tâche	Code
Ouvrez un nouveau script et sauvegardez-le dans votre dossier	

<p>« Exercise_Day2/scripts » sous le nom de « 05_2_tab_location.R ».</p>	
<p>Vous devrez d'abord avoir préparé vos données!</p> <p>Créez un nouveau tableau de données contenant les deux colonnes « Location » (à l'intérieur ou à l'extérieur de la réserve) et « Classification » (cas, contact) que vous nommerez « location_cases ».</p> <p>Extrayez la colonne « Location » du tableau de données des cas (en utilisant « select ») et créez une nouvelle colonne nommée « Classification » (en utilisant « mutate »).</p>	<pre>location_cases <- cases %>% select(Location) %>% mutate(Classification = "Cases")</pre>
<p>Créez un tableau de données de contacts contenant les mêmes colonnes que celles que vous avez créées pour les cas ci-dessus (« Classification » et « Location ») et nommez-le « location_contacts ».</p> <p>Excluez toutes les rangées qui contiennent la chaîne de texte « CASE » dans le « ContactID » (en utilisant « filter » et l'argument « !grepl »). Vous souvenez-vous de ce que font ces fonctions?</p>	<pre>location_contacts <- contacts %>% filter(!grepl("CASE", ContactID)) %>% distinct() %>% select(Contact_location) %>% rename(Location = Contact_location) %>% mutate(Classification = "Contacts")</pre>

<p>Ne sélectionnez que les rangées uniques (en utilisant « distinct ») et extrayez la colonne « Location » (en utilisant « select »). Renommez « contact_location » par « Location » (en utilisant « rename ») pour que cela corresponde aux noms de variables que vous avez créés pour les données « location_cases ». Recréez une nouvelle colonne pour qu'elle fasse la lecture des « Contacts » de toutes les rangées (en utilisant « mutate »). Nommez-la « Classification ».</p>	
<p>Combinez les deux tableaux de données que vous venez de créer en utilisant la fonction « bind » pour que l'emplacement des cas et des contacts se retrouve dans une colonne.</p> <p>Rappel : Vous pouvez utiliser la fonction « rm » afin de retirer les données désormais inutiles pour les tableaux de données (« location_cases », « location_contacts »), et ce, afin de maintenir l'ordre dans votre espace de travail.</p>	<pre>location <- rbind(location_cases, location_contacts) ; rm(location_cases, location_contacts)</pre>

<p>Résumez (« summarise ») la liste des nouveaux emplacements dans un tableau de fréquences avec le nombre de cas et de contacts à l'intérieur de la réserve ou à l'extérieur de celui-ci.</p> <p>Utilisez les fonctions « group_by » et « summarise » pour grouper et totaliser les données par classification et par lieu. Créez une colonne avec le pourcentage de « location » (en utilisant « mutate »).</p>	<pre>location <- location %>% group_by(Classification, Location) %>% summarise(Count = n()) %>% mutate(Percent = round(100*Count/sum(Count),1))</pre>
<p>Affichez le tableau de fréquences (nommez-le « tab_location ») en tant que « flextable » et affinez sa mise en forme à l'aide des options suivantes :</p> <ul style="list-style-type: none"> - Définissez la couleur de l'arrière-plan de l'en-tête à gris (« #E6E6E6 ») (en utilisant « bg »). - Mettez la police de l'en-tête en gras en utilisant la fonction « bold ». - Définissez la taille de police à 10 points dans tout le tableau (en utilisant « fontsize »). - Changez le type de police et choisissez « Arial » (en utilisant « font »). - Justifiez tout le texte au centre (du corps et de l'en-tête) en utilisant « align ». 	<pre>tab_location <- flextable(location) %>% bg(bg = "#E6E6E6", part = "header") %>% bold(part = "header") %>% fontsize(size = 10, part = "all") %>% font(part = "all", fontname = "Arial") %>% align(align = "center", part = "body") %>% align(align = "center", part = "header") %>% align(j=c("Location"), align = "left") %>% set_header_labels(Percent = "Percent (%)") %>% merge_v(j = c(1,2)) %>% set_caption(" Residential location of TB cases and contacts", style = "Table Caption", autonum = "autonum") %>% fix_border_issues() %>% autofit()</pre>

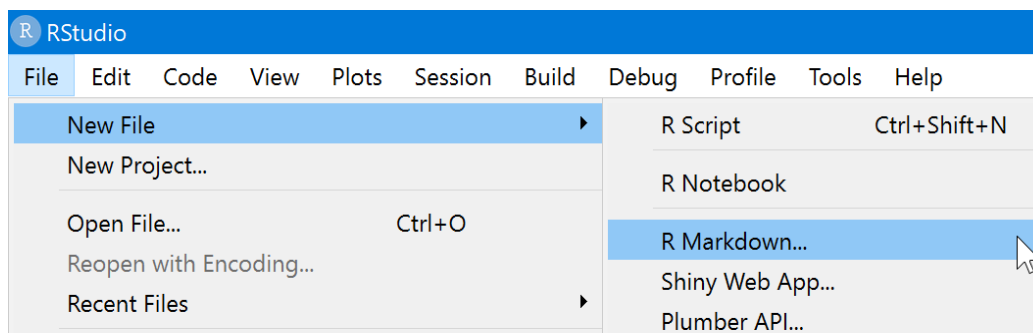
<ul style="list-style-type: none"> - Alignez la première colonne (« Location ») à gauche (en utilisant « align »). - Modifiez le nom de l'en-tête de colonne « Percent » pour « Percent (%) » (en utilisant « set_header_labels »). - Combinez verticalement les rangées dupliquées (colonnes 1 et 2) pour qu'elles ne se répètent pas (en utilisant « merge_v »). - Ajoutez le titre suivant à la figure (en utilisant « set_caption ») : « Residential location of TB cases and contacts ». Définissez le style « Table Caption » et utilisez « autonum » pour que la légende de cette figure soit numérotée dans Word. - Notez que combiner des rangées peut quelquefois supprimer quelques bordures extérieures. Utilisez l'argument « fix_border_issues » pour régler ce problème. - Créez des colonnes bien espacées en termes de largeur (en utilisant « autofit »). 	
Imprimez le tableau.	<code>tab_location</code>
Nettoyez votre espace de travail en retirant tout objet qui n'a pas sa place	<code>rm(location)</code>

dans l'analyse en utilisant la fonction « rm ».	
Assurez-vous d'inclure un titre dans votre script avec des détails concernant le but, l'auteur, la date, les modifications et d'autres notes pertinentes.	
Sauvegardez votre script en tant que « 05_2_tab_location.R ».	

8. Créez un rapport automatisé

Créez un nouveau fichier R Markdown en effectuant le chemin suivant :

« File », « New File », « R Markdown »




Sélectionnez le format des extraits (« Output Format ») par défaut : Word

Cliquez _> OK

Sauvegardez ce fichier sous le nom de « **Day2_final.Rmd** » dans votre dossier de scripts.

Remarque : Trouvez et examinez le document « R Markdown cheat sheet » et le document « TB 101 primer » que nous vous avons fournis dans le matériel de cours. Gardez-les en main afin de les utiliser pour vous guider dans cette section.

Tâche	Code
<p>Réviser le modèle de fichier nouvellement créé de R Markdown :</p> <ul style="list-style-type: none"> Quel type de document sera créé à partir de ce code? Que signifie « <code>{r}</code> » au début du court paragraphe de code? Que signifie « <code>```</code> » à la fin du court paragraphe de code? Qu'arrive-t-il si vous pesez sur le bouton « Knit »  et que vous sauvegardez le document créé? Comment le texte provenant du modèle de fichier de R Markdown apparaît-il 	<pre> 1 --- 2 title: "Untitled" 3 output: word_document 4 --- 5 6 ```{r setup, include=FALSE} 7 knitr::opts_chunk\$set(echo = TRUE) 8 ``` 9 10 ## R Markdown 11 12 This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, 13 PDF, and MS Word documents. For more details on using R Markdown see 14 <http://rmarkdown.rstudio.com>. 15 16 When you click the Knit button a document will be generated that includes both content as 17 well as the output of any embedded R code chunks within the document. You can embed an R code 18 chunk like this: 19 20 ```{r cars} 21 summary(cars) 22 ``` 23 24 ## Including Plots 25 26 You can also embed plots, for example: 27 28 ```{r pressure, echo=FALSE} 29 plot(pressure) 30 ``` 31 32 Note that the <code>echo = FALSE</code> parameter was added to the code chunk to prevent printing of the 33 R code that generated the plot. </pre>


<p>maintenant dans le dossier de résultat? Comment les titres sont-ils ajoutés? Comment la police du texte est-elle mise en gras?</p> <p>Consultez les documents suivants : « R Markdown Reference Guide » et « R Markdown Cheat Sheet ».</p>	
<p>Fermez le nouveau fichier R Markdown. Ouvrez et révisez « Day2.Rmd ». À quoi sert la fonction « source() »³ ?</p>	
<p>Ajoutez deux lignes au paragraphe de titre suivant « date » :</p>	

³ Pour plus d'information sur comment faire des scripts dans R, consultez le lien suivant : <https://www.earthdatascience.org/courses/earth-analytics/multispectral-remote-sensing-data/source-function-in-R/>

<ol style="list-style-type: none"> 1. « Modified by : [ajoutez votre nom] ». 2. « Date modified : [ajoutez la date] ». 	
<p>Réviser et utiliser le code dans le bloc « setup » à droite comme exemple pour :</p> <ol style="list-style-type: none"> a) Définir l'emplacement du paquet « here() » b) Charger les <i>librairies</i> et les données en utilisant « source() » pour accéder aux scripts que vous avez déjà écrits c) Nettoyer les données en utilisant « source() » pour accéder au script de nettoyage des données 	<pre># SETUP: set markdown file global (i.e. overall) options # this sets options for the whole document: here we have specified # not to show/"echo" the code in knitted output, and not to show warning # messages in output. knitr::opts_chunk\$set(echo = FALSE, message = FALSE) #Identify the location of the current script relative to project root directory. here::i_am("Exercise_Day2/scripts/Day2_final.Rmd") # LOAD: packages and data #Use source() to load required libraries and load data, recycling the scripts written earlier source(here::here("Exercise_Day2","scripts","01_2_load_libraries.r")) source(here::here("Exercise_Day2","scripts","01_3_load_data.r")) # CLEAN: inspect data, clean if necessary, and create new variables source(here::here("Exercise_Day2","scripts","02_1_clean_data.r"))</pre>

Expliquez dans vos propres mots ce que le code à droite permet de faire.	<pre>autonum <- run_autonum(seq_id = "tab", bkm = NULL, post_label = ":", pre_label = "Table ")</pre>
Au besoin, modifiez le code à droite pour refléter la figure que vous avez créée lorsque vous avez tracé les cas par mois.	<pre>```{r epicurve, fig.width=7, fig.height=4, fig.cap= "TB cases by date of diagnosis"} knitr::include_graphics(here("Exercise_Day2","output", "plot_cases_month.jpeg")) ```</pre>
Au besoin, modifiez le code à droite pour refléter la figure que vous avez créée lorsque vous avez tracé les cas par mois et par genre.	<pre>```{r , fig.width=7, fig.height=4, fig.cap= "TB cases by date of diagnosis and gender"} knitr::include_graphics(here("Exercise_Day2","output", "plot_cases_month_gender.jpeg")) ```</pre>
Au besoin, modifiez le code à droite pour refléter la figure que vous avez créée lorsque vous avez tracé les cas par mois et par degré d'infectiosité.	<pre>```{r time_infectiousness, fig.width=7, fig.height=4, fig.cap="TB cases by date of diagnosis and infectiousness"} knitr::include_graphics(here("Exercise_Day2","output", "plot_cases_month_infectiousness.jpeg")) ```</pre>
Si vous avez réalisé la courbe épidémique bonus, utilisez le code à droite pour refléter la figure que vous avez créée. Sinon,	<pre>```{r iso_epicurve, fig.width=7, fig.height=2, fig.cap="TB cases by week of symptom onset"} knitr::include_graphics(here("Exercise_Day2","output", "plot_cases_epiweek.jpeg")) ```</pre>

supprimez cette ligne.	
Au besoin, modifiez le code à droite pour refléter la figure que vous avez créée lorsque vous avez tracé les contacts par âge.	<pre>```{r contact_demogs_counts, fig.width=7, fig.height=4, fig.cap= "TB contacts by age group and gender, counts" } knitr::include_graphics(here("Exercise_Day2","output", "plot_contacts_agegender_count.jpeg")) ```</pre>
Au besoin, modifiez le code à droite pour refléter le script que vous avez créé pour analyser les lieux.	<pre>source(here("Exercise_Day2","scripts","03_3_tab_case_site.r"))</pre>
Si vous avez réalisé le tableau bonus, utilisez le code à droite pour refléter le script que vous avez créé. Sinon, supprimez cette ligne.	<pre>source(here("Exercise_Day2","scripts","05_2_tab_location.r"))</pre>
Au besoin, modifiez le code à droite pour refléter la figure que vous avez créée lorsque vous avez tracé le diagramme du réseau social.	<pre>knitr::include_graphics(here("Exercise_Day2","output", "plot_sna_location.jpeg"))</pre>
Bonus! Modifiez le texte à partir du document Word pour que fichier R	

Markdown reflète vos observations.	
Renommez et sauvegardez le document R Markdown. Incluez des détails pertinents dans l'en-tête.	
Tricotez le document Word (« Knit »).	

Quelles modifications apporteriez-vous à ce nouveau document Word selon la méthode que vous avez utilisée pour le codage de vos figures et de vos analyses?

Préférez-vous faire le codage de vos analyses dans un seul fichier R Markdown, ou importer des scripts source individuels comme nous l'avons fait pour les exercices du premier jour? Pourquoi préférez-vous une approche et non l'autre?

