SAE5.EMS.01- MENER UNE ETUDE STATISTIQUE DANS UN DOMAINE D'APPLICATION-

TD TP

2024-2025
SEBASTIEN PINEL
sebastien.pinel@univ-perp.fr

TD1- ETUDES STATISTIQUES : NOTIONS & CONCEPTS

1. EXERCICE. DOMAINE D'ETUDES STATISTIQUE

Pour chaque objectif d'étude statistique relier l'objectif scientifique au domaine d'étude associé parmi

Etude statistique et son objectif	Domaine d'étude
1. Améliorer la Convivialité des Sites Web : Identifier les problèmes de convivialité sur un site web en analysant les données d'interaction des utilisateurs et les retours d'information.	Sciences Sociales, Politiques Publiques.
2. Analyser le taux de désaffection des Clients : Comprendre pourquoi les clients quittent un service d'abonnement et élaborer des stratégies pour réduire le taux de désaffection.	Marketing, Publicité.
3. Évaluer l'Efficacité de la Publicité : Déterminer l'efficacité de différentes campagnes publicitaires en mesurant les changements dans la notoriété de la marque et les ventes.	Finance, Gestion de Portefeuille.
4. Étudier la Segmentation du Marché : Segmenter un marché cible en fonction de critères démographiques, psychographiques et de comportement d'achat.	Éducation, Planification Scolaire.
5. Évaluer les Risques dans les Portefeuilles Financiers : Quantifier et gérer les risques dans un portefeuille d'investissement en analysant les corrélations entre les actifs et la volatilité.	Criminologie, Application des Lois.
6. Prédire les Inscriptions Étudiantes : Prévoir le nombre d'étudiants qui s'inscriront à un programme universitaire pour l'année académique suivante.	Marketing, Gestion de la Relation Client.
7. Investiguer les Modèles de Criminalité : Identifier les modèles spatiaux et temporels dans les données criminelles pour informer les stratégies des forces de l'ordre.	Logistique, Gestion de la Chaîne d'Approvisionnement.
8. Optimiser la Chaîne d'Approvisionnement : Améliorer l'efficacité de la chaîne d'approvisionnement en analysant les schémas de demande, les délais de livraison et les niveaux de stock.	Marketing, Études de Marché.
9. Mesurer la Fidélité des Clients : Mesurer la fidélité des clients et les taux de rétention grâce à des enquêtes et à l'analyse du comportement des clients.	Informatique, UX/UI Design.
10. Évaluer l'Impact d'une Politique Publique : Évaluer l'impact d'une politique gouvernementale ou d'une intervention sur un résultat socio-économique spécifique.	Marketing, Service Client.

2. EXERCICE TYPES D'ETUDES I

1. Les psychologues de l'éducation étudient l'impact de différents types d'enseignement sur l'apprentissage. Dans une étude, les chercheurs ont enseigné une leçon de mathématiques à des élèves de 1ere année en utilisant le cycle d'enseignement "Inventer pour préparer l'apprentissage (IPL)". Un deuxième groupe d'élèves a reçu un enseignement traditionnel "dire et pratiquer". Après les cours, les deux groupes ont étudié seuls un exemple de problème mathématique. Ils ont ensuite passé un test qui comprenait des problèmes semblables à ceux de l'exemple travaillé. La revue Cognition and Instruction a publié les résultats en 2004.

Choisissez la meilleure description de cette étude :

a) Une étude expérimentale d'une association entre deux variables.
b) Une étude d'observation d'une association entre deux variables.
c) Une étude d'observation pour faire une estimation ou une affirmation à propos d'une population

2. Pendant 17 ans, des chercheurs ont étudié un échantillon de 707 personnes issues d'une même communauté. Ils ont enregistré le nombre d'heures passées par chaque individu devant la télévision pendant l'adolescence et le début de l'âge adulte. Plus tard, ils ont enregistré le nombre d'actes agressifs commis par les personnes participant à l'étude. Le magazine Science a publié les résultats en 2002 dans un article intitulé "Television Viewing and Aggressive Behavior during Adolescence and Adulthood".

Choisissez la meilleure description de cette étude :

Réponses

a) Une étude d'observation d'une association entre deux variables.
b) Une expérience visant à établir une relation de cause à effet entre deux variables
c) Une étude d'observation pour faire une estimation ou une affirmation sur une population

3. EXERCICE TYPES D'ETUDES II

Pour chaque étude, dire si l'étude est une étude d'expériences, étude d'observations ou une méta-analyse.

Étude 1

De nombreux étudiants écoutent de la musique en étudiant. L'écoute de la musique améliore-t-elle l'apprentissage ?
Questions de recherche spécifiques : La majorité des étudiants écoutent-ils de la musique pendant qu'ils étudient ? La majorité des étudiants pensent-ils qu'écouter de la musique améliore leur apprentissage ?
Pour répondre à ces questions, les étudiants en statistiques mènent une enquête dans leurs autres cours. Ils posent les deux questions suivantes :
Écoutez-vous de la musique lorsque vous étudiez ?
Pensez-vous qu'écouter de la musique améliore votre concentration et votre mémoire ?

Étude 2

De nombreux étudiants écoutent de la musique en étudiant. L'écoute de la musique améliore-t-elle l'apprentissage ?
Question de recherche spécifique : Lorsque nous comparons des étudiants qui étudient en écoutant de la musique à des étudiants qui étudient dans un environnement calme, quel groupe donne de meilleures notes pour la compréhension de ce qu'ils ont étudié ?
Pour répondre à cette question, le chercheur divise la classe en deux groupes : (1) ceux qui écoutent de la musique lorsqu'ils étudient et (2) ceux qui n'écoutent pas de musique lorsqu'ils étudient. Les élèves tiennent un journal pendant une semaine. Chaque fois qu'ils étudient, ils notent les informations suivantes :
Durée de la session d'étude (en minutes) : Une évaluation de la façon dont ils ont compris ce qu'ils ont étudié, sur une échelle de 1 à 10 : 1 = aucune compréhension, 10 = excellente compréhension.

Étude 3

De nombreux étudiants écoutent de la musique en étudiant. L'écoute de la musique améliore-t-elle l'apprentissage ?
Question de recherche spécifique : L'écoute de la musique améliore-t-elle la capacité des étudiants à identifier rapidement les informations ?
Pour étudier cette question, le chercheur utilise des grilles de mots à trouver. Elle divise la classe en deux groupes. Les élèves d'un côté de la salle font une grille de mots à trouver pendant 3 minutes tout en écoutant de la musique sur un iPod. Les élèves de l'autre côté de la salle font une grille de mots à trouver de mots pendant 3 minutes sans musique. L'instructeur calcule le nombre moyen de mots trouvés par chaque groupe.

Figure 1. IPod (avant les smartphones)

4. Une étude a pris un échantillon aléatoire d'étudiants et les a interrogés sur leurs horaires de coucher. Les données ont montré que les personnes qui dormaient au moins 8 heures avant le jour de l'examen avaient plus de chances d'obtenir de bonnes notes que celles qui dormaient moins de 8 heures.

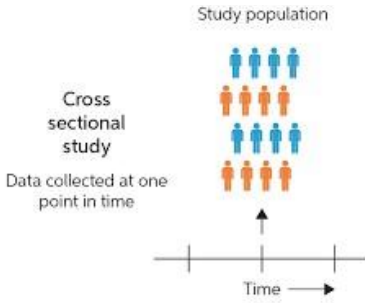
5. Dans le cadre d'une étude, des personnes ont été réparties au hasard entre deux groupes. Le groupe 1 a été invité à suivre un programme d'études strict pendant une période déterminée, tandis que le groupe 2 a été invité à étudier de la même manière qu'auparavant. Les chercheurs ont cherché à savoir quel groupe obtenait les meilleurs résultats aux examens.

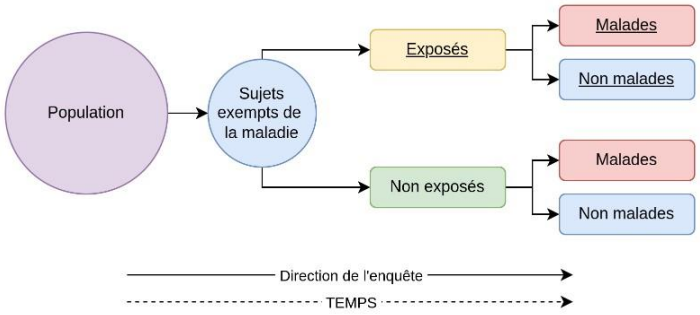
6.Une étude a pris un échantillon aléatoire de personnes et a examiné leurs habitudes en matière de tabagisme. Chaque personne a été classée comme petit, moyen ou gros fumeur. Le chercheur a examiné le niveau de stress de chaque groupe.

4. EXERCICE. TYPES D'ETUDES OBSERVATIONNELLES.

La figure ci-dessous présente trois types d'études observationnelles.

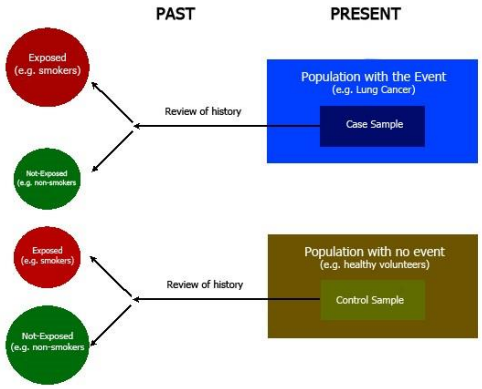
ETUDES TRANSVERSALES : observation de variables d'une population à un « instant » donné. Elles permettent d'observer la fréquence de survenue d'un phénomène de santé, dans une population, à un moment précis. Les informations sont recueillies sur une période brève et fournissent un indicateur statique de morbidité : la prévalence.





ETUDES LONGITUDINALES/COHORTES PROSPECTIVES suivi de la fréquence de ce phénomène de santé au cours du temps et fournissent un indicateur dynamique de morbidité : l'incidence.

ÉTUDE CAS-TÉMOINS. Pour étudier des facteurs potentiellement en cause, choix de la population est fait la base d'une population i) de malades, ii) témoin non-malade (mêmes caractéristiques générales).
Les deux groupes des malades et des témoins non malades sont étudiés grâce à des questionnaires et « remonter le temps ». L'objectif est d'analyser un certain nombre de caractéristiques qui ont pu potentiellement influencer la survenue de cette maladie.



En vous aidant des schémas et des descriptions associées, donner les réponses associées au QCM ci-dessous.

1. Un groupe de chercheurs étudie la prévalence de l'hypertension artérielle chez les personnes âgées de 65 ans et plus dans une ville donnée à un moment précis. Quel type d'étude observationnelle est-ce ?
a) Étude de cohorte prospective
b) Étude cas-témoins
c) Étude transversale

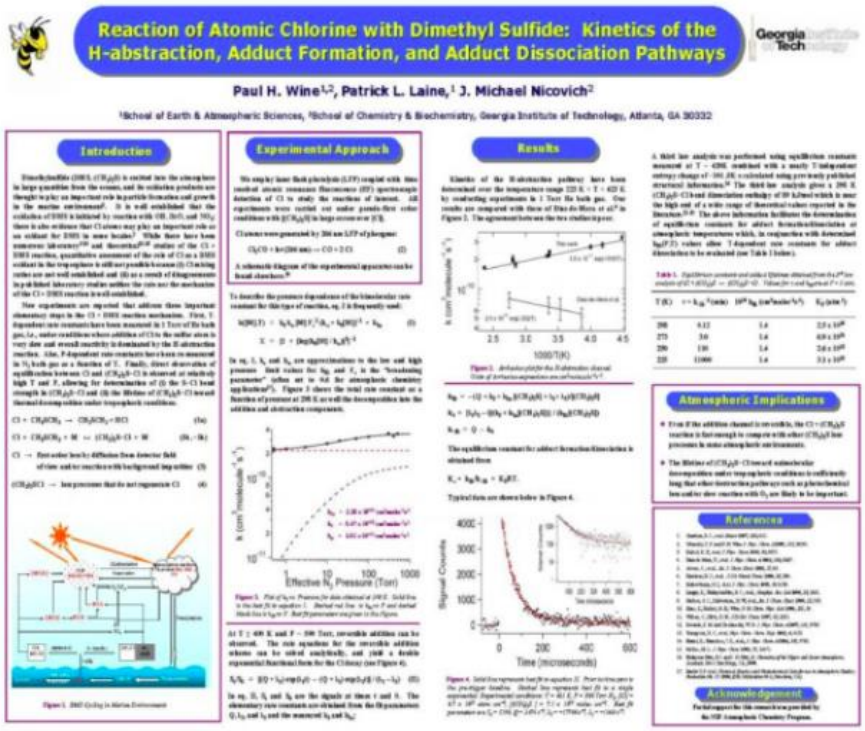
2. Des chercheurs examinent les dossiers médicaux de patients atteints de maladies cardiaques pour identifier les facteurs de risque qui ont contribué au développement de ces maladies. Quel type d'étude observationnelle est-ce ?
a) Étude de cohorte prospective
b) Étude cas-témoins
c) Étude de cohorte rétrospective

3. Une enquête de recherche visant à explorer les facteurs de risque associés au développement du cancer du poumon. Le protocole suivant est retenu :
Sélection des Cas : Les chercheurs identifient un groupe d'individus ayant reçu un diagnostic de cancer du poumon. Ces individus sont considérés comme les "cas".
Sélection des Témoins : Un groupe témoin est sélectionné, composé d'individus qui n'ont pas de cancer du poumon, mais qui sont similaires à d'autres caractéristiques aux cas. Ces individus servent de groupe de comparaison.
Collecte de Données : Des informations détaillées sur les caractéristiques démographiques, le mode de vie et d'autres facteurs de risques sont recueillies. Cela peut inclure des données sur l'histoire du tabagisme, l'exposition aux toxines environnementales, l'historique professionnel, les antécédents familiaux de cancer, et plus encore.
Quel type d'étude observationnelle est-ce ?
a) Étude de cohorte prospective
b) Étude cas-témoins
c) Étude transversale

5. EXERCICE. EVALUATION DE POSTERS

Pour les posters ci-dessous, compléter avec une les points positifs et négatifs (remplir en face des points). Il vous faut trouver au minimum le nombre de point proposés.

	Négatifs	Positifs
Choix des couleurs General Des graphiques		
Graphique Couleurs Pertinence		
Texte Orthographe Détail Typographie En gras Surligné Alinéa, Bullet	Faute Trop long	Sans faute Ecourté
Structure générale	Organisation confuse	Organisation claire
Structure détail Références Logo (financeur, entreprise laboratoire) Introduction Objectifs Résultats/discussion Conclusion	Absence	Présence

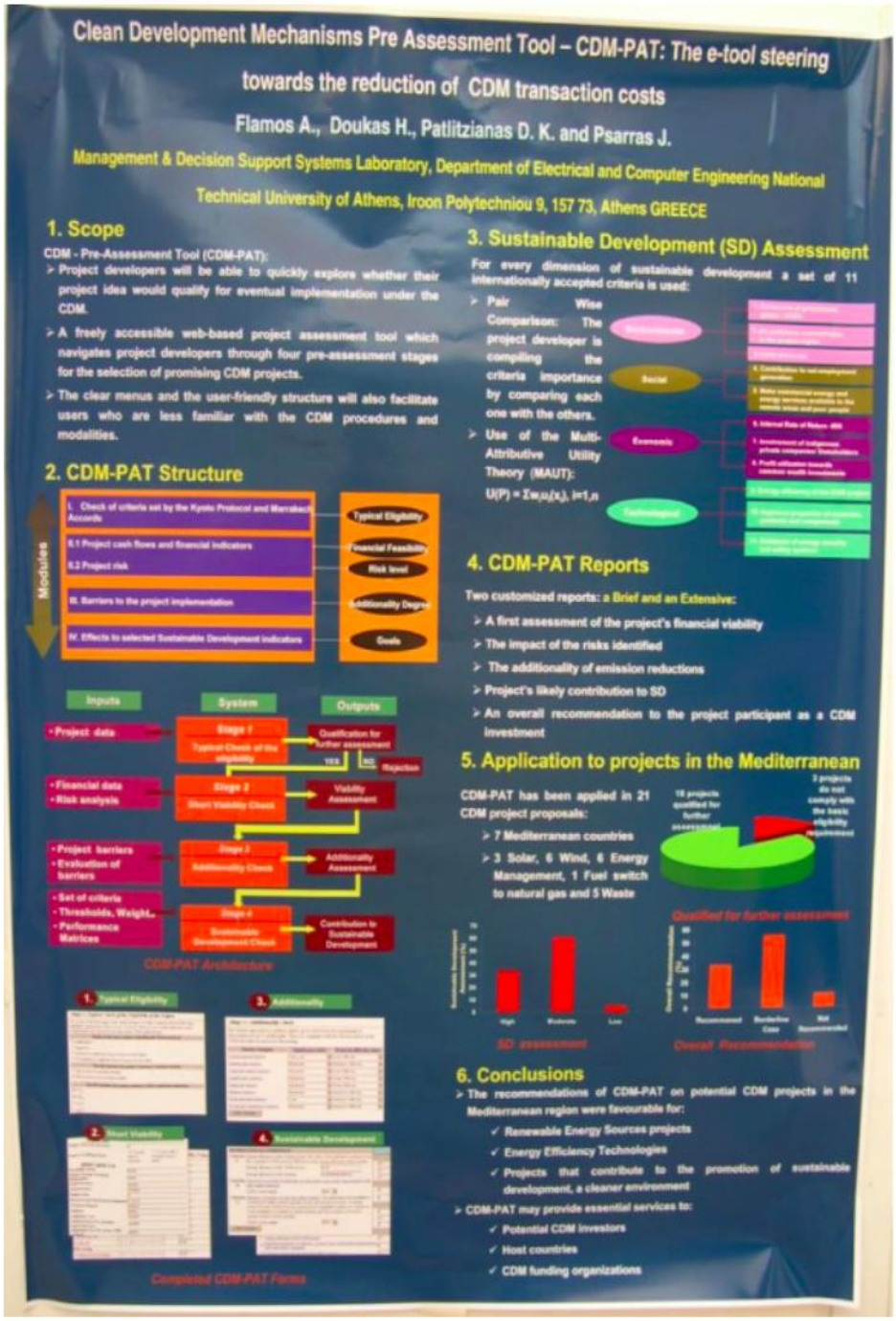


Points + :

- …
- …
- …

Points - :

- …
- …
- …
- …

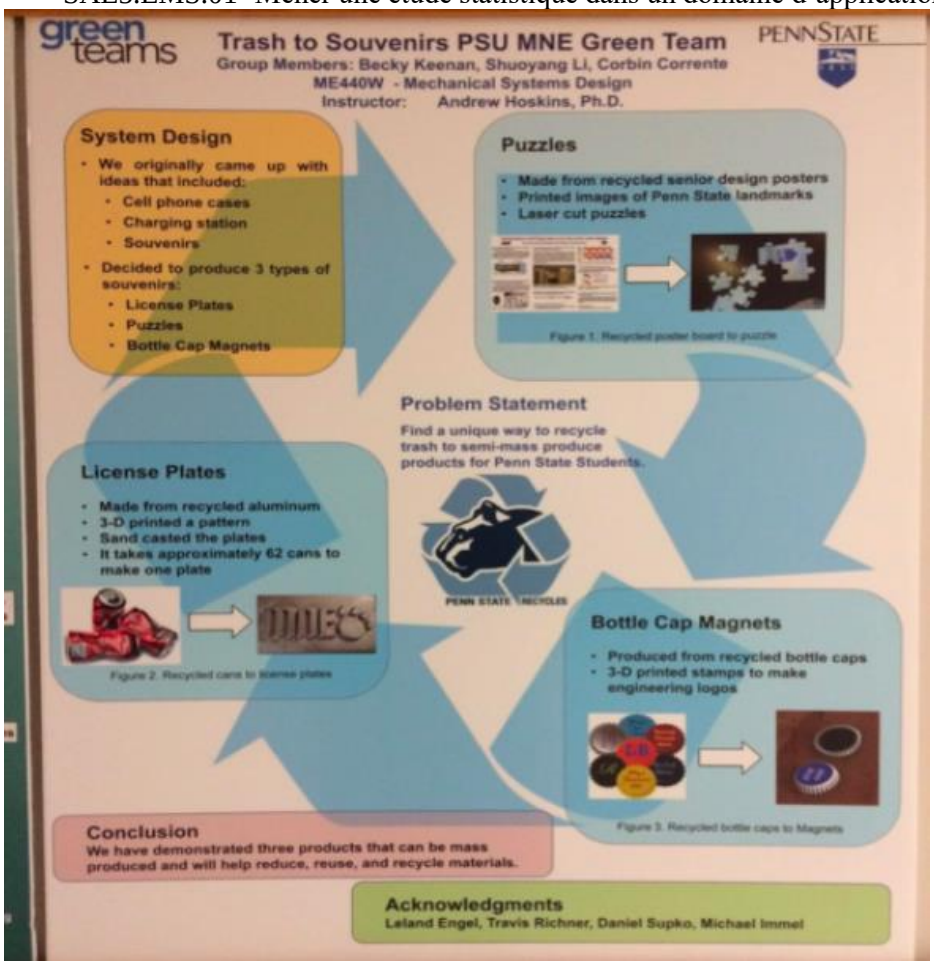


Points + :

- …
- …
- …

Points - :

- …
- …
- …
- …

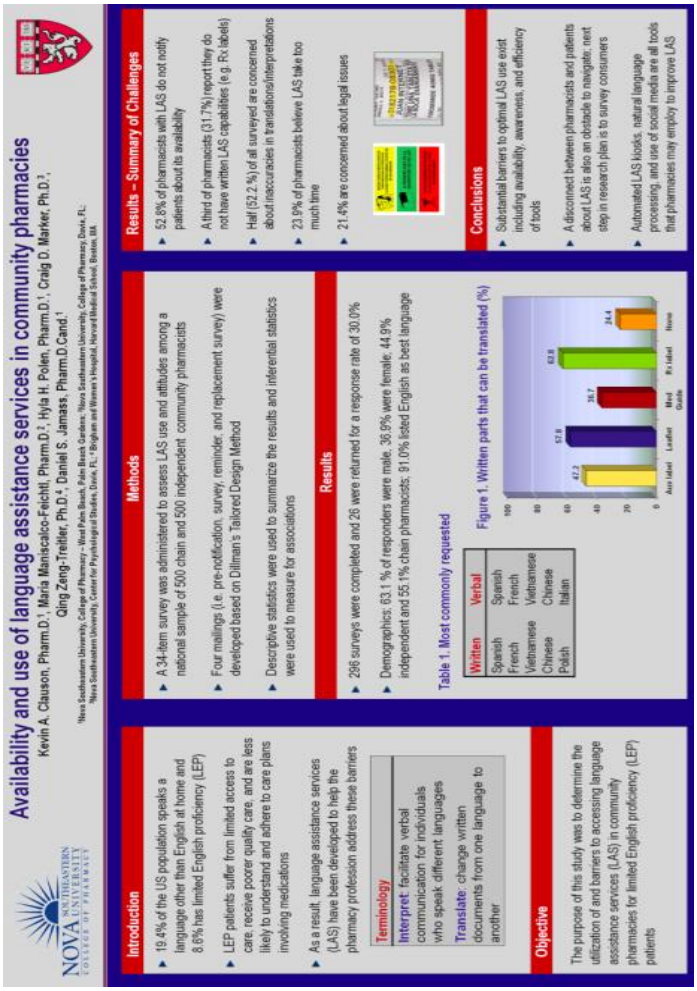


Points + :

- …
- …
- …
- …

Points - :

- …
- …
- …
- …



Points + :

- …
- …
- …
- …
- …
- …

Points - :

- …
- …
- …
- …
- …
- …
- …
- …

6. EXERCICE. TRAITEMENT HORMONAL SUBSTITUTIF

Lorsque les femmes sont ménopausées, la production d'hormones dans leur corps change. Ces changements hormonaux peuvent provoquer divers symptômes qui peuvent être atténués par un traitement hormonal substitutif. Dans les années 1980, le traitement hormonal substitutif était courant aux États-Unis.

Au début des années 1990, des études d'observation ont suggéré que le traitement hormonal substitutif présentait des avantages supplémentaires, notamment une réduction du risque de maladie cardiaque. Dans ces études d'observation, les chercheurs ont comparé les femmes qui prenaient des hormones à celles qui n'en prenaient pas.
Les dossiers médicaux ont montré que les femmes prenant des hormones après la ménopause présentaient une incidence plus faible de maladies cardiaques. Mais les femmes qui prennent des hormones sont différentes des autres femmes. Elles ont tendance à être plus riches et plus éduquées, à avoir une meilleure alimentation et à se rendre plus souvent chez le médecin. Ces femmes ont de nombreuses habitudes et avantages qui contribuent à une bonne santé, il n'est donc pas surprenant qu'elles aient moins d'infarctus. Mais ces études permettent-elles de conclure que les hormones sont à l'origine de la réduction du nombre d'infarctus ? Non. Les résultats sont faussés par l'influence de ces autres facteurs.

En 2022, la *Women's Health Initiative* a parrainé une expérience à grande échelle et bien conçue pour étudier les conséquences sur la santé du traitement hormonal substitutif. Dans le cadre de cette expérience, les chercheurs ont assigné au hasard plus de 16 000 femmes à l'un des deux traitements. Un groupe a pris des hormones. L'autre groupe a pris un placebo. Un placebo est une pilule sans principe actif qui ressemble à la pilule hormonale. L'expérience a été menée en double aveugle. En aveugle signifie que les femmes ne savaient pas si elles recevaient des hormones ou un placebo. Le double aveugle signifie que les informations étaient codées, de sorte que les chercheurs qui administraient les pilules ne savaient pas quel traitement les femmes recevaient. Après 5 ans, le groupe prenant des hormones présentait une incidence plus élevée de maladies cardiaques et de cancer du sein. C'est exactement le résultat inverse de celui obtenu dans les études d'observation ! En fait, les différences étaient si importantes que les chercheurs ont mis fin prématurément à l'expérience. Les *National Institutes of Health* ont déclaré que les études d'observation étaient erronées. Le traitement hormonal substitutif des symptômes de la ménopause est aujourd'hui rarement utilisé.

Quel est le point principal ?
Une étude d'observation peut fournir des preuves d'un lien ou d'une association entre deux variables. Mais d'autres facteurs peuvent également influencer les résultats. Ces autres facteurs sont appelés variables confusionnelles. L'influence des variables confusionnelles sur la variable réponse est l'une des raisons pour lesquelles une étude d'observation fournit des preuves faibles, et potentiellement trompeuses, d'une relation de cause à effet. Une expérience bien conçue prend des mesures pour éliminer les effets des variables confusionnelles, notamment l'affectation aléatoire des personnes aux groupes de traitement, l'utilisation d'un placebo ou des conditions en aveugle. Grâce à ces précautions, une expérience bien conçue fournit des preuves convaincantes de la relation de cause à effet.

1. Complétez :
Dans les études sur la substitution hormonale, le fait que la femme prenne ou non des hormones est la variable ……..
Le fait qu'une femme participant à l'étude ait eu ou non une crise cardiaque est la variable ……...

2. Quels sont les facteurs susceptibles de constituer des variables confusionnelles dans l'étude d'observation ? Comment l'expérience a-t-elle permis de contrôler les effets de ces facteurs ?

7. MONTAGE DE KAKEMONO

Par groupe, monter et démonter les kakemonos fournis par l'enseignant.

8. EXERCICE. VALIDATION DE POSTERS ETUDIANTS.

Pour les poster proposé par les étudiants lors d'une SAE, utiliser la grille de notation pour proposer une note sur 20.

	/2	/3	/3	/4	/4	/4	Points négatifs	/20
Projet	**Présentation agréable**	**Texte (longueur typologie)**	**Texte (orthographe, vocabulaire)**	**Illustrations**	**Découpage**	**Exactitude**	**Autre (préciser)**	**Total**
Cocktail Master								
Fusée								
Planes								
Popcorn								
Thé								
Water Pong								

TD2- DATAVIZZ SOUS PYTHON

LIBRAIRIES de DataViz sous Python (liste non-exhaustive)

Matplotlib	Plotly	bqplot	imageio	Folium
Seaborn	geoplotlib	Bokeh	mpld3	Cartopy
Plotnine(ggplot)	Gleam	D3Blocks	Matplotlib + ipywidgets	pygal
Bokeh	Leather	Altair	Streamlit	missingno

INITIATION A LA CARTOGRAPHIE
Voir document *IntroductionAuSIG__SIGWithFolium.pdf*

1. FAIRE UN GIF SOUS PYTHON

Le **Graphics Interchange Format (GIF)** est un format d'image bitmap développé par une équipe du fournisseur de services en ligne CompuServe en 1987. Il est largement utilisé sur le Web en raison de son large soutien et de sa portabilité.
Le format prend en charge jusqu'à 8 bits par pixel pour chaque image, ce qui permet à une seule image de référencer sa propre palette de 256 couleurs différentes choisies dans l'espace colorimétrique RVB 24 bits. Il prend également en charge les animations et permet une palette séparée de 256 couleurs maximum pour chaque image. Ces limitations de la palette rendent le format GIF moins adapté à la reproduction de photographies en couleur avec des dégradés de couleurs, mais bien adapté aux images plus simples telles que les graphiques ou les logos.
Les images GIF sont compressées à l'aide de la technique de compression de données sans perte Lempel-Ziv-Welch (LZW) afin de réduire la taille du fichier sans dégrader la qualité visuelle.

Pour créer le GIF, appliquer le code suivant :

```
import imageio.v2 as imageio
import os
MainDir = "VotreChemin/DataForTP_20231014/AralSeaDrying"
files = os.listdir(MainDir)
Output_file = "C:/Users/sebpi/Desktop/mygif.gif"
images_path = [os.path.join(MainDir,file) for file in files]

images = []
for img in images_path:
    images.append(imageio.imread(img))

imageio.mimsave(Output_file, images, fps=4,loop=10)
```

Lire le GIF crée avec un outil de votre choix (e.g., un navigateur)

Quel est le défaut majeur de ce GIF ?

Le GIF crée est constitué de 4 images de la mer d'Aral à 4 dates différentes.
Quelles sont les années de ces dates ?
Quel est le message que le GIF cherche à faire passer ?
(si vous n'êtes pas au courant faire une recherche rapide sur internet)

A quoi servent les options
Loop : ……. *Fps* : ……
2. Application : création de votre GIF
Prendre 5 photos avec votre mobile sous Python faire une GIF.

2. CARTES INTERACTIVES SOUS PYTHON

1. Chargement des librairies et du jeu de données

```
import folium
from folium import plugins
from folium.plugins import BeautifyIcon
import pandas as pd
import webbrowser
import numpy as np
from folium.plugins import FloatImage
from folium.features import DivIcon
```

Chargement des données.
Nous chargeons les coordonnées géographique (latitude, longitude) de 6 points de collectes et de l'aéroport le plus proche.

```
F_in_StationLocations = "VotreChemin/StationsLocation.xlsx"
F_out = "VotreCheminPourSauvegarderLaCarte/ManausMap.html"

df_StationsLocation = pd.read_excel(F_in_StationLocations)
df_StationsLocation.columns = ['Label', 'Lat', 'Lon']
df_airport=pd.DataFrame(data=np.array([[-3.15,-59.9833]])
,columns=['latitude','longitude'])
```

Il s'agit de données de projet de recherche. Il y a 6 points de collecte sur une rivière et une station météorologique. En inspectant les données, donner

Les coordonnées de la station météorologique :
Les noms des 6 points de collecte :

2. Carte basique

```
study_zone_map = folium.Map(location=[-3.1190, -60.0217],
                        tiles = 'OpenStreetMap',
                        width="%10",height="%100",)
study_zone_map.save(F_out)
```

Pour afficher la carte si vous êtes dans Jupyter

```
study_zone_map
```

Si vous êtes dans un autre éditeur que Jupyter

```
webbrowser.open(F_out)
```

Modifier les options pour obtenir une carte qui s'affiche sur tout l'écran.
Dans quel pays sommes-nous ?

En vous aidant de l'aide, décrire à quoi servent les 4 options appelées dans la fonction *folium.Map()*.

Selon l'aide combien y a-t-il de fonds de carte ?

Il est possible de charger beaucoup de fond de cartes disponibles sur internet : https://leaflet-extras.github.io/leaflet-providers/preview/

3. Ajout d'éléments

3.1 Ajoutons un marqueur quelconque.

```
icon_star = BeautifyIcon(icon='star', inner_icon_style='color:black;font-size:15px;',
                         background_color='transparent', border_color='transparent',)
```

```
# add the icon to the map
folium.Marker(location=[df_airport.latitude.values,df_airport.longitude.values],
              tooltip="Click me!",
              popup='Airport in the Popup',
              icon=icon_star,).add_to(study_zone_map)
```

Afficher votre carte. Qu'avez-vous rajouté ?

3.2 Ajout de marqueurs de type cercle.

```
folium.Circle(location=[df_StationsLocation['Lat'][0],df_StationsLocation['Lon'][1]],
               tooltip="Click me!",
               popup=df_StationsLocation['Label'][0],
               color='crimson',fill=True,radius=15).add_to(study_zone_map)
```

Afficher votre carte. Qu'avez-vous rajouté ?

3.3 Ajout de labels sur la carte.

```
folium.map.Marker(
  location=[df_airport.latitude.values,df_airport.longitude.values],
  icon=DivIcon(icon_size=(150,36),
               icon_anchor=(0,0),
               html='Airport in the map',)
  ).add_to(study_zone_map)
```

Afficher votre carte. Qu'avez-vous rajouté ?

3.4. Boucle.
Faire une boucle pour réaliser la carte ci-contre où on présente tous les labels, les 6 points de collecte (icone « cercles rouges ») et l'aéroport (icone « étoile noire »).

3.5 Ajout de mini-carte zoom

```
minimap = plugins.MiniMap(zoom_level_offset=-7)
study_zone_map.add_child(minimap)
```

3.6. Ajout de légende
Une image jpeg a été générée en parallèle et stockée dans *F_LegendStudyZone*. Utiliser un logiciel d'Edition graphique pour visualiser l'image. Que contient elle ?

Un rajoute cette image dans un coin de la carte (ajuster les option *bottom left* et *scale*).

```
legend_html = '''
<div style="position: fixed; top: 50px; left: 50px; z-index: 9999;
background-color: white; border:2px solid grey; border-radius:3px; padding: 10px;">
     <i class="fa-solid fa-church" style="color:blue"></i><span>   Places to
see </span><br>
     <i class="fa-solid fa-umbrella-beach" style="color:green"></i><span>  
Beach</span><br>
     <i class="fa fa-star" style="color:black"></i><span>   Meteorological
station</span><br>
     <i class="fa-solid fa-circle" style="color:red"></i><span>   River
sample</span><br>
</div>
'''
study_zone_map.get_root().html.add_child(folium.Element(legend_html))
```

Sauver, puis afficher cette nouvelle carte.
Qu'a-t-on ajouté ?
En quel langage est écrit la légende ?

3.7. Ajout d'échelle et de flèche du nord
L'échelle et la flèche du nord sont des éléments indispensables en cartographie.
Folium a déjà une échelle pré-enregistrée. IL suffit de l'activer

```
study_zone_map.control_scale = True
```

Il faut ensuite récupérer un pointeur nord (libre) puis l'afficher

```
north_arrow_url                                                                =
'https://upload.wikimedia.org/wikipedia/commons/8/84/North_Pointer.svg'
# Add the north arrow image to the map
FloatImage(north_arrow_url,
           bottom=75, left=85,scale=0.2).add_to(study_zone_map)
```

Afficher votre carte. Vérifiez que vous avez bien rajouté échelle et e pointeur vers le nord.

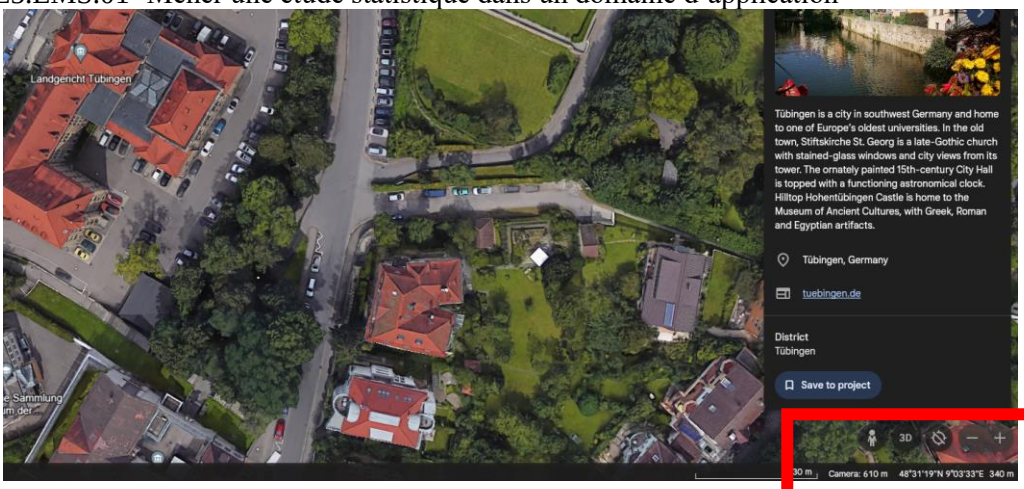
4. Publions la carte en ligne
4.1. Se connecter à son espace Github
4.2. Créer un nouveau repository (espace de dépôt) que vous appellerez *EasyMapWebsite* avec les options *public* et *readme.md* cochées.
4.3. Setting>Pages>Build and deployment>Branch>main/(root) →save
4.4. Upload *DataForTP/Logo_IUT_SD_Carcassonne.jpg* et *ManausMap.html*
4.5. Créer un fichier *_config.yml*

```
theme: jekyll-theme-minimal
title: Simple Map Website
description: This website demonstrates how to easily publish and display an interactive
map made with Folium.
logo: Logo_IUT_SD_Carcassonne.jpg
#You     can     also     add     a     logo     from     an     image     URL,     logo:
https://upload.wikimedia.org/wikipedia/commons/3/3f/UPVD_Logo_2023.jpg
```

4.6. Setting>Pages> copier l'adresse du lien (du type : *https://spinel1385.github.io/EasyMapWebsite/*)
4.7. Create an index.md file and fill it with the following command

```
Easy map website
Here s my easy map website !
<iframe src="ManausMap.html" height="500" width="500"></iframe>
 You can explore this map at https://spinel1385.github.io/EasyMapWebsite
```

5. Application : cartographie de la ville de vos parents
5.1 A l'aide de https://earth.google.com/, déterminer 5 lieux à visiter dans la ville de vos parents (votre maison, la mairie, le centre de culte, l'école, le psychologue de votre chat ou autres).
Google Earth fournit les coordonnées degrés, minutes, seconds (DMS) (en bas à droite de votre image). Noter les 5



5.2 Convertir en degrés décimaux (utiliser ChatGPT pour générer une fonction qui fait la conversion).
5.3 Stocker les 5 informations dans un dataframe avec les colonnes (lat, lon, label)
5.4 Réaliser une carte interactive qui montrera les 5 points
5.5 La sauvegarder au format html
5.6 La mettre en ligne (sur un dépôt GitHub par exemple)
5.7 L'envoyer à votre famille par email

TP1_TP2- ETUDE DE CAS GUIDEE

Objectifs

Objectifs de Data Science	Objectifs statistiques
Scraper/récupérer des données directement sur un site web (rvest)	Discuter de l'impact du biais d'auto-déclaration sur les réponses aux enquêtes
Filtrage des données (dplyr)	Définir et créer un tableau de contingence
Écrire des fonctions pour manipuler les données de façon répétitive	Test du chi-deux pour l'indépendance : Mise en œuvre et interprétation
Travailler avec des chaînes de caractères (stringr)	
Remodeler les données dans différents formats (tidyr)	
Visualisation des données (ggplot2) avec des labels (directlabels)	
Graphiques *multiplots* (cowplot)	

Lors de la conception d'un rapport, n'oubliez pas les parties suivantes

Avant le corps du rapport	Après en annexe
Sommaire de l'étude	Diagramme de Gantt de vos activités
Liste des figures	Figures complémentaires
Liste des tables	Informatique : code, environnement
Glossaire	Autres documents que vous jugez utiles

ETUDE DE CAS GUIDEE : santé mentale chez la jeunesse américaine

Le présent rapport est **classiquement** articulé de la manière suivante :

1. Introduction
2. Méthodes
3. Matériel (Données)
4. Résultats 1 (DataViz)
5. Résultats 2 (Analyses statistiques)
6. Discussions
7. Conclusion (graphique et textuelle)
8. Annexes

1. INTRODUCTION

1.1. Motivation

Selon un récent rapport, les taux de dépression semblent augmenter chez les jeunes Américains depuis 2010 environ. Une étude récente montre également que les jeunes semblent rechercher davantage de soins auprès des services de santé mentale.

Cette étude de cas explorera :

- l'évolution des taux d'épisode majeur de dépression (EMD) depuis les années 2000 et dans les différents sous-groupes de jeunes (âge, genre)
- l'évolution des taux de traitement de la dépression chez les jeunes.

Selon le rapport DSM 5, les principaux symptômes d'un EDM sont les suivants :

Troubles du sommeil	Déficit de concentration
Trouble de l'appétit	Retard psychomoteur ou agitation
Déficit d'intérêt (anhédonie)	Culpabilité (dévalorisation, désespoir, regrets)
Déficit d'énergie	Suicidalité

Figure 2. Dépression : symptômes et traitements

Cette étude de cas est motivée par les deux articles suivants : Twenge et al., (2019) et Olfson et al., (2014)

Les principales conclusions du premier article sont :

- Les taux d'EDM au cours de la dernière année ont augmenté de 52 % entre 2005 et 2017 chez les adolescents âgés de 12 à 17 ans et de 63 % entre 2009 et 2017 chez les jeunes adultes âgés de 18 à 25 ans.
- Les EDM sévères ont également augmenté chez les jeunes adultes âgés de 18 à 25 ans entre 2008 et, avec des augmentations moins constantes et plus faibles chez les adultes âgés de 26 ans et plus.
- Les tendances culturelles (essor de la communication numérique et la diminution de la durée du sommeil) contribuant à l'augmentation des troubles de l'humeur et des pensées et comportements suicidaires depuis le milieu des années 2000, notamment, peuvent avoir eu un impact plus important sur les jeunes, créant ainsi un effet de cohorte.

Les principales conclusions du deuxième article sont :
- Les soins de santé mentale des jeunes ont augmenté plus rapidement que ceux des adultes
- Entre 1995-1998 et 2007-2010, les consultations donnant lieu à des diagnostics de troubles mentaux ont augmenté significativement plus vite pour les jeunes.

En conclusion, alors que la dépression semble être en augmentation chez les jeunes, ces derniers semblent également rechercher davantage de soins de santé mentale.

1.2. Objectif de l'étude et principales questions

Dans cette étude de cas, nous utiliserons les données de la *National Survey on Drug Use and Health* (NSDUH) liées au traitement et au taux d'EDM pour étudier comment les tendances en matière de santé mentale ont évolué au fil du temps et comment les différents groupes se comparent. Ces données ont également été utilisées dans le premier article cité dans la section motivation.

L'objectif de cette étude est d'apporter des éléments de réponse aux questions suivantes :

1. Comment les taux de dépression chez les jeunes Américains ont-ils évolué depuis 2004, selon les données de la NSDUH ? Comment les taux ont-ils varié entre les différents sous-groupes de jeunes (âge, genre, origine ethnique) ?
2. Les services de santé mentale semblent-ils atteindre davantage de jeunes ? Là encore, comment les taux diffèrent-ils entre les différents sous-groupes de jeunes (âge, genre, origine ethnique) ?

1.3 Contexte de l'étude

Pour contextualiser l'examen de la santé mentale des jeunes Américains, nous commençons par étudier le taux de suicide aux États-Unis. Selon le Centre pour le contrôle et la prévention des maladies (CDC) des Etats-Unis, le taux de suicide a augmenté pour les deux genres.

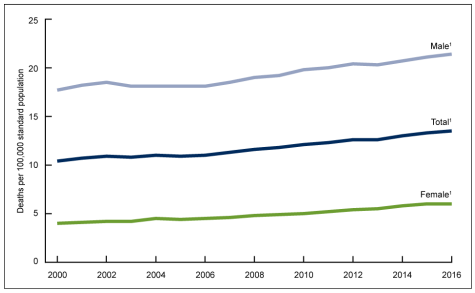


Figure 3. Evolution du taux de suicide au Etats-Unis (source)

Si le suicide semble augmenter chez les jeunes, il semble également augmenter dans la plupart des groupes d'âge aux États-Unis au cours des quinze dernières années, tant chez les femmes que chez les hommes :

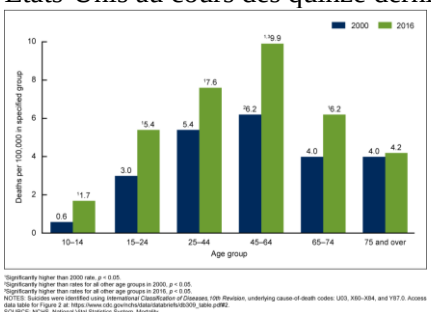


Figure 4. Taux de suicide pour les femmes, par âge aux Etats-Unis(source)

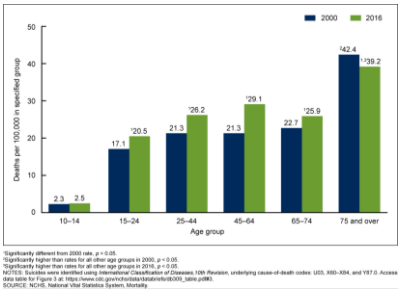


Figure 5. Taux de suicide pour les hommes, par âge aux Etats-Unis(source)

Depuis 2008, le suicide est la dixième cause de décès pour tous les âges aux États-Unis.

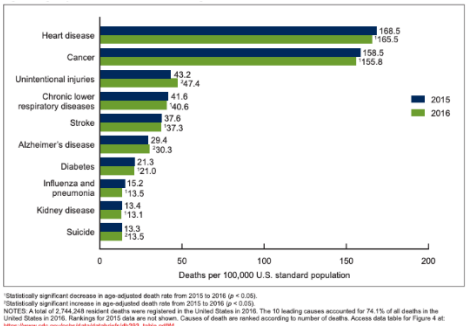


Figure 6. Cause de décès au Etats-Unis en 2015 et 2016 (source)

En 2016, le suicide est devenu la deuxième cause de décès chez les jeunes.
Ainsi, bien que le suicide soit en augmentation dans la plupart des groupes d'âge, le suicide est l'une des deux principales causes de décès chez les jeunes. Cela justifie donc un examen plus approfondi de la santé mentale des jeunes Américains.

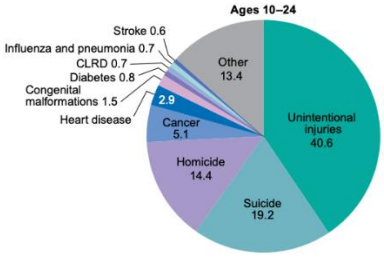


Figure 7. Cause de décès au Etats-Unis chez les jeunes (source)

Historiquement, les taux de suicide étaient beaucoup plus élevés avant 1950, mais on constate une augmentation au cours des 20 dernières années.

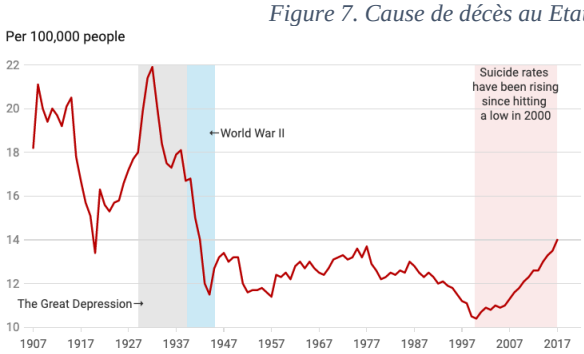


Figure 8. Evolution du taux de suicide (1907-2017) aux Etats-Unis, source : U.S. Centers for Disease Control and Prevention

Outre les États-Unis, d'autres pays connaissent également une augmentation des taux de dépression chez les jeunes. Voir ce rapport de l'Organisation mondiale de la santé (OMS) sur les taux de dépression dans d'autres pays. Voir ici pour une discussion intéressante sur les causes possibles de l'augmentation des taux de dépression.

2. METHODES

2.1. Importer des données du web : data scrapping

Les données sont souvent disponibles en ligne. Parfois, les données peuvent être téléchargées sur une page web sous la forme d'un fichier texte délimité ou d'un fichier Excel. Cependant, il arrive que les données ne soient pas disponibles de cette manière, comme les données de l'enquête NSDUH.

Comment procéder dans ce cas ?

- Copier manuellement chaque donnée dans un autre fichier que nous devrions également importer, mais ce processus est souvent inefficace, sujet à des erreurs et non reproductible. Supposons que nous voulions effectuer une analyse l'année suivante sur les données de l'année suivante et qu'elles soient formatées de la même manière.
- Lorsque les données sont disponibles au format PDF, le package `pdftools` permet de telles extractions. Voir cette autre étude de cas et cette autre étude de cas pour deux méthodes de travail avec les PDF.
- Utiliser `R` pour récupérer les données sur le web ! (Web scrapper)

Principales étapes du web scraping Le web scraping est le processus d'extraction des données d'une page web. Voici ses deux étapes principales :

- dans notre navigateur web, identifier **l'emplacement** des données sur la page web qui sera scrappée.
- dans l'environnement de programmation `R`, enregistrer l'**élément** de la page web dans un **objet** R.

L'**emplacement** des données sur la page web qui sera scannée peut-être identifiée à l'aide d'un langage appelé XPath (abréviation de XML Path Language). Il est utilisé pour identifier des parties (dans ce cas appelées éléments) d'un document écrit dans le langage XML. Le XML (abréviation de Extensible Markup Language (langage de balisage extensible)), est fréquemment utilisé pour les documents sur l'internet, à l'instar du HTML. L'une des principales différences entre ces deux langages est que le HTML ne fournit pas d'informations structurelles, alors que le XML en fournit. Ces informations structurelles peuvent être utilisées pour analyser les documents afin de pouvoir extraire d'un site web uniquement les données qui nous intéressent.

Autres ressources de web scrapping : Vignette, Blog

2.2. Statistiques

Outre les visualisations de moyennes ou pourcentage, ce rapport met en œuvre le Test du Chi-2 de Pearson d'indépendance. Le test du χ^2 de Pearson d'indépendance est utilisé pour déterminer s'il existe une différence statistiquement significative entre les fréquences attendues et les fréquences observées dans une ou plusieurs catégories d'un tableau de contingence.

L'hypothèse nulle H0 est "les variables sont indépendantes", ou que "la différence entre la proportion des fréquences observées et celle des fréquences attendues est égale à zéro".

Pour réaliser ce test, nous

- créons d'abord un tableau de contingence, qui dans ce cas pourrait également être appelé tableau 2x2 (tableau de contingence simple avec deux niveaux pour deux variables).
- calculons la statistique du χ^2 et un degré de liberté df
- calculons la p-value d'apres la statistique du χ^2 et une distribution du $\chi^2(df)$.

Dans R, le test du χ^2 de Pearson d'indépendance est réalisé via les fonctions (au choix)

- la fonction `chisq.test()` du package `stats`
- la fonction `prop_test()` du package `rstatix` qui fournit des informations plus détaillées

2.3. Fichier RmD et packages nécessaires

Créer un fichier TP1_EtudeDeCasGuidee.Rmd avec l'entête YAML et le 1er chunck qui fixent les conditions d'exécution des chuncks du reste du document.

````
---
title: "TP1-2: Etude de cas guidée : santé mentale chez la jeunesse americaine"
output:
  word_document:
    toc: yes
    number_sections: true
  pdf_document:
    toc: no
    number_sections: true
    pandoc_args:
    includes:
      in_header: header.tex
---

```{r setup, include=FALSE}
library(knitr)
options(crayon.enabled = NULL)
knitr::opts_chunk$set(include = TRUE, comment = NA, echo = TRUE,
                      message = FALSE, warning = FALSE, cache = FALSE,
                      fig.align = "center", out.width = '90%')
rmarkdown:::perf_timer_reset_all()
rmarkdown:::perf_timer_start("render")
```
````

Chargeons-les packages nécessaires à l'étude :

```
library(here)
library(rvest)
library(dplyr)
library(magrittr)
library(stringr)
library(tidyr)
library(tibble)
library(purrr)
library(ggplot2)
library(directlabels)
library(scales)
library(forcats)
library(ggthemes)
library(rstatix)
library(cowplot)
```

```
# install.packages("OCSdata")
library(OCSdata) #https://github.com/opencasestudies/OCSdata OpenCase study data
library(fs)  # For creating directories
```

Table 1. Details sur les packages.

Package	Utilisés dans cette etude
here	charger et enregistrer facilement des données
rvest	récupérer des pages web
dplyr	filtrer les données pour des groupes spécifiques, pour remplacer des valeurs spécifiques par "NA", pour renommer des variables et pour exécuter des fonctions sur des variables multiples
magrittr	utiliser et réaffecter des objets de données à l'aide de l'opérateur %<>%pipe
stringr	manipuler des chaînes de caractères
tidyr	changer la forme ou le format des tibbles pour qu'ils soient larges et longs
tibble	créer des tableaux et convertir les valeurs d'une colonne en noms de lignes
purrr	appliquer une fonction à chaque colonne d'un tableau ou à chaque tableau d'une liste
ggplot2	créer des plots
directlabels	ajouter des étiquettes directement aux lignes dans les tracés
scales	créer un graphique permettant de voir à quoi ressemblent les différents types de lignes
forcats	réorganiser le facteur dans les plots
ggthemes	créer un graphique pour voir à quoi ressemblent les différents types de lignes
rstatix	effectuer un test de proportionnalité
cowplot	combiner des plots ensemble
OCSdata	accéder aux fichiers de données OCS et les télécharger

Pour créer nos graphiques, nous allons utiliser le ggplot2 package. Rappelons les principales terminologies de `ggplot2`

- **ggplot** - la fonction principale dans laquelle vous spécifiez l'ensemble de données et les variables à tracer (c'est ici que nous définissons les noms des variables `x` et`y`)
- **geoms** - objets géométriques, e.g. `geom_point()`, `geom_bar()`, `geom_line()`, `geom_histogram()` - **aes** - esthétique : forme, transparence, couleur, remplissage, types de lignes
- **échelles** - définition de la manière dont vos données seront représentées : continues, discrètes, logarithmiques, etc.

3. DONNEES

3.1. Présentation

Les données sont issues de l'enquête National Survey on Drug Use and Health (NSDUH) fourni par Substance Abuse and Mental Health Services Administration (SAMHSA), un organisme de U.S. Department of Health and Human Services (DHHS).

Cette enquête a débuté en 1971 et est menée chaque année dans les 50 États et le district de Columbia. Environ 70 000 personnes (âgées de 12 ans et plus) sont interrogées chaque année sur des questions liées à la santé. Seules les personnes civiles, non institutionnalisées, sont incluses dans l'enquête. Les ménages sont sélectionnés au hasard, puis un enquêteur professionnel se rend à l'adresse en question et demande à un ou deux résidents de l'interroger. L'enquêteur apporte avec lui un ordinateur portable que les participants utilisent pour remplir l'enquête, qui dure généralement une heure. Si un participant choisit de participer, il reçoit 30 dollars en espèces. Toutes les informations recueillies sont confidentielles et servent à la surveillance des maladies et à l'orientation des politiques publiques, notamment en ce qui concerne la consommation de drogues et d'alcool et la santé mentale. Voir ici pour plus de détails sur l'enquête. Les données sont mises à la disposition du public en ligne sur le site Substance Abuse & Mental Health Data Archive.

Sur le site avec les données de l'enquête, les résultats sont affichés dans de nombreux tableaux. Il n'existe aucun moyen évident de télécharger les données directement à partir de ce site web.

Si vous cliquez sur le bouton TOC dans le coin supérieur gauche, vous serez dirigé vers un autre site web, où un grand document PDF contenant tous les résultats peut être téléchargé. Nous souhaitons étudier l'évolution des taux de dépression et l'interaction des jeunes avec les services de santé mentale. Les tableaux suivants nous intéressent :

Table 2. Liste des tables scrappées dans cette étude

Table	Type	Details
Tableau 11.1A	Non-démographique	Milieux où des services de santé mentale ont été reçus au cours de l'année écoulée parmi les personnes âgées de 12 à 17 ans : nombres en milliers, 2002-2018
Tableau 11.1B	Non-démographique	Milieux où des services de santé mentale ont été reçus au cours de l'année écoulée chez les personnes âgées de 12 à 17 ans : pourcentages, 2002-2018
Tableau 11.2A	Démographique en comptage	Épisode dépressif majeur au cours de l'année écoulée chez les personnes âgées de 12 à 17 ans, par caractéristiques démographiques : Nombre en milliers, 2004-2018
Tableau 11.2B	Démographique en pourcentage	Épisode dépressif majeur au cours de l'année écoulée chez les personnes âgées de 12 à 17 ans, par caractéristiques démographiques : Pourcentage, 2004-2018
Tableau 11.3A	Démographique en comptage	Épisode dépressif majeur avec déficience grave au cours de la dernière année chez les personnes âgées de 12 à 17 ans, selon les caractéristiques démographiques : Nombre en milliers, 2006-2018
Tableau 11.3B	Démographique en pourcentage	Épisode dépressif majeur avec déficience grave au cours de la dernière année chez les personnes âgées de 12 à 17 ans, selon les caractéristiques démographiques : Pourcentages, 2006-2018
Tableau 11.4A	Démographique en comptage	Traitement de la dépression au cours de la dernière année chez les personnes âgées de 12 à 17 ans ayant eu un épisode dépressif majeur au cours de la dernière année, selon les caractéristiques démographiques : Nombre en milliers, 2004-2018
Tableau 11.4B	Démographique en pourcentage	Traitement de la dépression au cours de l'année écoulée chez les personnes âgées de 12 à 17 ans ayant eu un épisode dépressif majeur au cours de l'année écoulée, par caractéristiques démographiques : Pourcentage, 2004-2018

Notre objectif est d'intégrer ces données dans R afin de pouvoir les explorer.

Remarque : définition d'un EDM selon la NSDUH, en 2018,

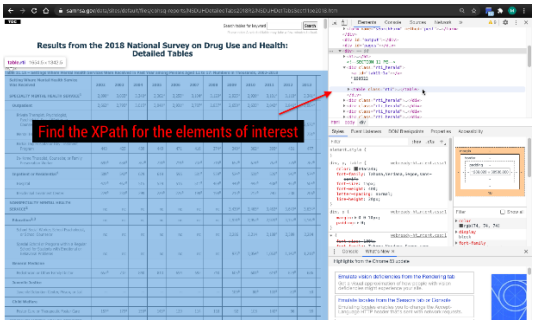
- Les répondants ont été définis comme ayant eu un EDM au cours des 12 derniers mois s'ils ont eu au moins une période de 2 semaines ou plus au cours de l'année écoulée au cours de laquelle ils ont eu une humeur d'humeur dépressive ou de perte d'intérêt ou de plaisir dans les activités quotidiennes, accompagnée de problèmes de sommeil, d'alimentation, d'énergie, de concentration ou d'estime de soi. Les questions de l'EDM sont basées sur les critères diagnostiques du DSM-5. Certaines formulations des questions pour les adolescents âgés de 12 à 17 ans et les adultes âgés de 18 ans ou plus, afin de les rendre plus accessibles
- Les adolescents ont été définis comme souffrant d'un EDM avec une déficience sévère si leur dépression leur causait de graves problèmes dans leur capacité à accomplir les tâches ménagères, à réussir au travail ou à l'école, à s'entendre avec leur famille, à se sentir à l'aise dans la vie de tous les jours, au travail ou à l'école, de s'entendre avec leur famille ou d'avoir une vie sociale.

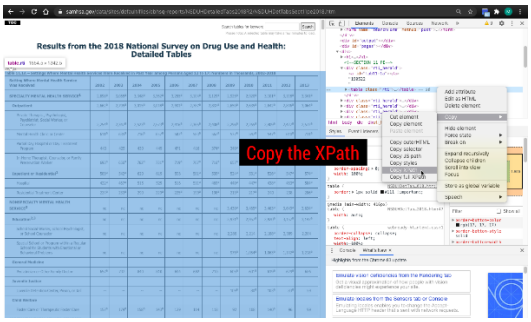
3.2. Importation via web scrapping (package `rvest`)

Le package `rvest` permet d'importer les données directement à partir des tableaux du site web. Les deux étapes du web scrapping peuvent être décomposées :

- Identifier l'emplacement des données qui seront récupérées
  - Aller sur web page avec tous les tableaux qui nous intéressent.
  - cliquer avec le bouton droit de la souris et sélectionner "Inspecter" (la page web). Une fenêtre s'ouvre. Cette fenêtre nous permet de jeter un coup d'œil sur les mécanismes internes de la page web.
  - passer le pointeur sur les composants de l'élément (page web) jusqu'à ce que les données soient trouvées
    Pour extraire les données de la page web, nous devons d'abord en apprendre un peu plus sur les composants qui en font la page web qu'elle est. En passant notre souris sur les éléments de la page web, nous mettons en évidence la section de la page web qu'ils représentent. En survolant plusieurs éléments et en cliquant sur les éléments à droite de l'écran, nous pouvons identifier l'élément qui contient les données que nous recherchons.
  - Copier XPath des données recherchées (Pour la 1ère table, XPath est `/html/body/div[4]/div[1]/table`)
- Enregistrer l'élément de la page web dans un objet dans R
  - Importer le code html de la page web (fonction `read_html()` du paquet `rvest`)
  - extraire des morceaux de documents HTML (page web) à l'aide de XPath (fonction `html_elements()` du paquet `rvest`)
  - analyser les données extraites dans un dataframe (convertissons ce tableau html en un dataframe en utilisant la fonction `html_table()` du paquet `rvest`)

Vous trouverez ci-dessous un aperçu de la procédure.





Pour extraire le tableau 11.1A du site web, nous allons donc procéder avec l'une de ces 2 URLs :

```
NSDUH_url <- "https://www.samhsa.gov/data/sites/default/files/cbhsq-reports/NSDUHDetailedTabs2018R2/NSDUHDetTabsSect11pe2018.htm"
NSDUH_url <- "https://www.opencasestudies.org/ocs-bp-youth-mental-health/data/raw/samhsa.gov_2020_tables.htm"
```

Utilisons le pipe operateur %>% avec les fonctions suivantes :
• la fonction read_html() sauvegarde le document html de la page web dans R.
• la fonction html_elements() du package rvest sélectionne uniquement l'élément Table 11.1A de la page web.
• la fonction html_table() du package rvest analyse l'objet html dans un dataframe.
• la sortie est une liste avec un élément de html_table(). Aussi, pour extraire les données de la liste nous utiliserons les crochets [[]] pour sélectionner le premier élément de la liste.

```
table11.1a <- NSDUH_url %>%
  xml2::read_html() %>%
  rvest::html_elements(xpath = '/html/body/div[4]/div[1]/table') %>%
  rvest::html_table()
table11.1a <- table11.1a[[1]]
```

3.3. Ecriture d'une fonction pour récupérer plusieurs tables

Pour récupérer les autres tables, une solution consiste

- à copier et à coller le code que nous avons écrit ci-dessus
- à créer une fonction R pour accomplir cette tâche de manière succincte.

Dans ce cas, `XPATH` sera utilisé comme un "argument d'entrée" pour la fonction, qui sera remplacé par un XPath réel et ensuite utilisé dans les étapes suivantes pour extraire les données de chaque table pour laquelle un XPath est fourni. Nous appellerons cette fonction `scraper`.

Répétons le processus ci-dessus pour les autres tables qui nous intéressent.

```
scraper <- function(XPATH){
  table <- NSDUH_url %>%
    read_html() %>%
    html_elements(xpath = XPATH) %>%
    html_table()
  output <- table[[1]]
  output
}
```

Nous pouvons maintenant appliquer la fonction que nous avons créée à chacun des XPaths pour chacun des tableaux du site web que nous souhaitons utiliser dans notre analyse de données.

```
table11.1b <- scraper(XPATH = "/html/body/div[4]/div[2]/table")
table11.2a <- scraper(XPATH = '/html/body/div[4]/div[3]/table')
table11.2b <- scraper(XPATH = '/html/body/div[4]/div[4]/table')
table11.3a <- scraper(XPATH = '/html/body/div[4]/div[5]/table')
table11.3b <- scraper(XPATH = '/html/body/div[4]/div[6]/table')
table11.4a <- scraper(XPATH = '/html/body/div[4]/div[7]/table')
table11.4b <- scraper(XPATH = '/html/body/div[4]/div[8]/table')
```

Nous avons réussi à importer les données du web dans R. Nous enregistrons les données importées dans un répertoire appelé *data>imported*.

```
fs::dir_create(here("data", "imported")) #create the folders
save(table11.1a, table11.1b, table11.2a, table11.2b,
     table11.3a, table11.3b, table11.4a, table11.4b,
     file = here::here("data", "imported", "imported_data.rda"))
```

Ensuite, nous devons structurer les données en vue de les analyser.

3.4. Data wrangling (structuration des données)

3.4.1. (Re-)chargement des données

Si vous reprenez le TP, vous pouvez charger vos données structurées d'un des trois manières suivantes :

```
load(file = here::here("data", "imported", "imported_data.rda"))
```

Sinon après avoir chargé le package `OCSdata`, vous pouvez charger les données importées.

```
install.packages("OCSdata")
library(OCSdata)
imported_data("ocs-bp-youth-mental-health", outpath = getwd())
load(here::here("OCSdata", "data", "imported", "imported_data.rda"))
```

Enfin, si le paquet ne fonctionne pas, vous pouvez également trouver un fichier RDA des données ici ou ici. Téléchargez ce fichier et placez-le dans votre répertoire de travail actuel dans un sous-répertoire "*data>imported*.

```
#load(here::here("data", "imported", "imported_data.rda"))
```

Maintenant que nous avons importé les données, structurons un 1er tableau. Qu'entendons-nous par "structurer" ? nous avons l'intention de mettre les données dans ce que l'on appelle un format "ordonné". Cela signifie que les données

- les données contiennent une seule ligne pour chaque observation
- les données présentent un seul aspect de chaque observation dans une seule colonne
- les données sont rectangulaires (c'est-à-dire qu'il n'y a pas de cellules vides)
- les valeurs contenues dans les cellules sont dans un format utile pour la visualisation et l'analyse.

Étant donné que les données semblent être formatées de manière similaire dans chacun des tableaux, il est probable que les mesures que nous prenons pour manipuler ce premier tableau seront également nécessaires pour manipuler les tableaux suivants. Cette similitude permettra d'automatiser le processus d'extraction.

3.4.2. Cas de la Table11.1a

Jetons un coup d'œil à notre tableau sur le site web pour voir ce qu'il faut faire pour le mettre dans un format "ordonné".
Tout d'abord, nous pouvons voir que nous devons supprimer la légende de notre tableau.

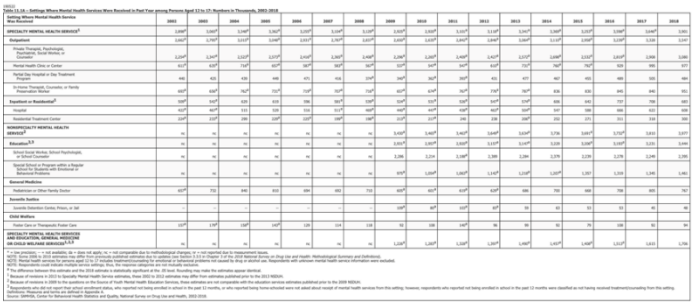


Figure 9. Tableau 11.A

En utilisant la fonction class qui décrit les types d'objets de données dans R, le tableau 11.1a est un dataframe typique.

```
class(table11.1a)
```

Nous pouvons regarder la dernière ligne en utilisant la fonction `tail` du package `dplyr`. Nous pouvons spécifier que nous ne voulons voir que la dernière ligne en utilisant l'argument `n = 1`. Pour utiliser les fonctions `dplyr`, nous devons d'abord transformer ce tableau en un tibble (version `tidyverse` d'un dataframe) via la fonction `as_tibble()`)

```
table11.1a %>%
  dplyr::as_tibble() %>%
  tail(n = 1)
```

Nous pouvons voir que la légende est répétée pour chaque colonne. Examinons maintenant la colonne de l'année 2004.

```
table11.1a %>%
  dplyr::as_tibble() %>%
  dplyr::select(`2004`) %>%
  tail(n = 1)
```

Sauvegardons ceci dans un objet appelé `legend` afin de nous y référer plus tard :

```
legend <- table11.1a %>%
  as_tibble() %>%
  select(`2004`) %>%
  tail(n = 1)
```

Une autre façon de regarder la dernière ligne est d'utiliser la fonction `n()` du package `dplyr`. Cette fonction peut être utilisée à l'intérieur d'autres fonctions `dplyr` et elle compte le nombre total d'observations d'un groupe. Dans la fonction `slice()` du package `dplyr`, elle vous permet de faire référence à la longueur totale de l'objet.

```
table11.1a %>%
  dplyr::as_tibble() %>%
  dplyr::slice(n())
```

Nous allons utiliser un opérateur de pipe double d'affectation composée `%<>%` du package `magrittr`. Cela nous permet d'utiliser le `tableau11.1a` comme entrée et de le réassigner à la fin après que toutes les étapes aient été effectuées.

Nous transformons les données en tibble, qui est la version `tidyverse` d'un dataframe en utilisant la fonction `as_tibble()`. Nous utilisons la fonction `slice()` du package `dplyr` pour supprimer cette ligne, en utilisant la fonction `slice` pour sélectionner de la première ligne en utilisant `1:` à l'avant-dernière ligne en utilisant `n()-1`.

```
table11.1a %<>%
  dplyr::as_tibble() %>%
  slice(1:(n()-1))
```

Examinons maintenant les données :

```
slice_head(table11.1a, n = (length(pull(table11.1a, `2002`))))
```

```
# A tibble: 20 × 18
   Setting Where Mental Healt…¹ `2002` `2003` `2004` `2005` `2006` `2007` `2008`
   <chr>                        <chr>  <chr>  <chr>  <chr>  <chr>  <chr>  <chr>
 1 "SPECIALTY MENTAL HEALTH SE… "2,89… "3,06… "3,34… "3,36… "3,25… "3,10… "3,12…
 2 "Outpatient"                 "2,66… "2,79… "3,01… "3,04… "2,93… "2,78… "2,83…
 3 "Private Therapist, Psychol… "2,25… "2,34… "2,52… "2,57… "2,41… "2,36… "2,40…
 4 "Mental Health Clinic or Ce… "611a" "635a" "716a" "657a" "587a" "583a" "567a"
 5 "Partial Day Hospital or Da… "440"  "425"  "439"  "449"  "471"  "416"  "374a"
 6 "In-Home Therapist, Counsel… "693a" "656a" "762a" "731a" "719a" "707a" "716a"
 7 "Inpatient or Residential1"  "509a" "542a" "629"  "619"  "596"  "581a" "539a"
 8 "Hospital"                   "422a" "467a" "515"  "529"  "516"  "511a" "469a"
 9 "Residential Treatment Cent… "224a" "233a" "299"  "229a" "225a" "199a" "198a"
10 "NONSPECIALTY MENTAL HEALTH… "nc"   "nc"   "nc"   "nc"   "nc"   "nc"   "nc"
11 "Education2,3"               "nc"   "nc"   "nc"   "nc"   "nc"   "nc"   "nc"
12 "School Social Worker, Scho… "nc"   "nc"   "nc"   "nc"   "nc"   "nc"   "nc"
13 "Special School or Program … "nc"   "nc"   "nc"   "nc"   "nc"   "nc"   "nc"
14 "General Medicine"           ""     ""     ""     ""     ""     ""     ""
15 "Pediatrician or Other Fami… "657a" "732"  "840"  "810"  "694"  "692"  "710"
16 "Juvenile Justice"           ""     ""     ""     ""     ""     ""     ""
17 "Juvenile Detention Center,… "--"   "--"   "--"   "--"   "--"   "--"   "--"
18 "Child Welfare"              ""     ""     ""     ""     ""     ""     ""
19 "Foster Care or Therapeutic… "157a" "179a" "158a" "143a" "129"  "114"  "118"
20 "SPECIALTY MENTAL HEALTH SE… "nc"   "nc"   "nc"   "nc"   "nc"   "nc"   "nc"
# ℹ abbreviated name: ¹`Setting Where Mental Health ServiceWas Received`
# ℹ 10 more variables: `2009` <chr>, `2010` <chr>, `2011` <chr>, `2012` <chr>,
#   `2013` <chr>, `2014` <chr>, `2015` <chr>, `2016` <chr>, `2017` <chr>,
#   `2018` <chr>
```

Nous pouvons voir que la légende ne fait plus partie des données.

Utilisons maintenant la légende pour recoder les données. Il y a beaucoup de valeurs différentes pour les données manquantes, que nous voudrions remplacer par `NA`. Utilisons la fonction `pull()` du package `dplyr` pour jeter un coup d'oeil aux données de la légende :

```
dplyr::pull(legend, `2004`)

[1] "* = low precision; -- = not available; da = does not apply; nc = not comparable due to methodologica
l changes; nr = not reported due to measurement issues.\r\nNOTE: Some 2006 to 2010 […] for Behavioral Hea
lth Statistics and Quality, National Survey on Drug Use and Health, 2002-2018."
```

Question 1 : les cinq types de données sont : `.....`, `.....`, `.....`, `.....`, et `.....`. (Compléter)

Nous pouvons utiliser la fonction `na_if()` pour recoder ces valeurs en `NA`.

```
table11.1a %<>%
# apply na_if to all columns
  mutate(across(everything(), ~dplyr::na_if(.x, "nc"))) %>%
  mutate(across(everything(), ~dplyr::na_if(.x, "--"))) %>%
  mutate(across(everything(), ~dplyr::na_if(.x, ""))) %>%
  mutate(across(everything(), ~dplyr::na_if(.x, "*")))
head(table11.1a)

# A tibble: 6 × 18
  Setting Where Mental…¹ `2002` `2003` `2004` `2005` `2006` `2007` `2008` `2009`
  <chr>                  <chr>  <chr>  <chr>  <chr>  <chr>  <chr>  <chr>  <chr>
1 "SPECIALTY MENTAL HEA… 2,898a 3,065a 3,348a 3,362a 3,255a 3,104a 3,129a 2,925a
2 "Outpatient"           2,662a 2,795a 3,015a 3,048a 2,931a 2,787a 2,837a 2,650a
… … … … … … … … … … …
5 "Partial Day Hospital… 440    425    439    449    471    416    374a   340a
6 "In-Home Therapist, C… 693a   656a   762a   731a   719a   707a   716a   657a
# i abbreviated name: ¹`Setting Where Mental Health ServiceWas Received`
# i 9 more variables: `2010` <chr>, `2011` <chr>, `2012` <chr>, `2013` <chr>,
#   `2014` <chr>, `2015` <chr>, `2016` <chr>, `2017` <chr>, `2018` <chr>
```

Examinons les noms des colonnes de notre tableau :

```
colnames(table11.1a)

 [1] "Setting Where Mental Health ServiceWas Received"
 [2] "2002"
 […] …
[17] "2017"
[18] "2018"
```

Renommons la première colonne en utilisant la fonction `rename()` du package `dplyr`.

```
table11.1a %<>%
  dplyr::rename(MHS_setting = `Setting Where Mental Health ServiceWas Received`)
head(table11.1a)

# A tibble: 6 × 18
  MHS_setting     `2002` `2003` `2004` `2005` `2006` `2007` `2008` `2009` `2010`
  <chr>           <chr>  <chr>  <chr>  <chr>  <chr>  <chr>  <chr>  <chr>  <chr>
1 "SPECIALTY MEN… 2,898a 3,065a 3,348a 3,362a 3,255a 3,104a 3,129a 2,925a 2,920a
2 "Outpatient"    2,662a 2,795a 3,015a 3,048a 2,931a 2,787a 2,837a 2,650a 2,635a
… … … … … … … … … … …
6 "In-Home Thera… 693a   656a   762a   731a   719a   707a   716a   657a   674a
# i 8 more variables: `2011` <chr>, `2012` <chr>, `2013` <chr>, `2014` <chr>,
#   `2015` <chr>, `2016` <chr>, `2017` <chr>, `2018` <chr>
```

Vous remarquerez que certaines valeurs des colonnes des années sont suivies d'un "a". Selon la légende, cela indique que "la différence entre cette estimation et l'estimation de 2018 est significative au niveau α=.05". Bien que cette information soit utile, elle rend difficile l'utilisation de nos valeurs numériques, c'est pourquoi nous voulons la supprimer. Étant donné que les valeurs "a" minuscules apparaissent dans les valeurs de la colonne `MHS_setting` (comme outp**a**tient), nous voulons nous assurer que nous ne supprimons pas toutes les valeurs "a" du tableau, mais seulement celles des colonnes des années.

Comment faire ? Nous utiliserons

- le package `stringr` pour modifier les chaînes de caractères (fonction `str_replace()` pour remplacer, `str_remove()` pour supprimer).
- les fonctions `dplyr` `mutate()`, `select()` et `across()` pour spécifier les colonnes que nous voulons modifier.

Actuellement, toutes nos données sont des chaînes de caractères comme l'indique le `<chr>` sous le nom des colonnes.

Voici une partie d'une notice d'information sur la manipulation des chaînes de caractères.

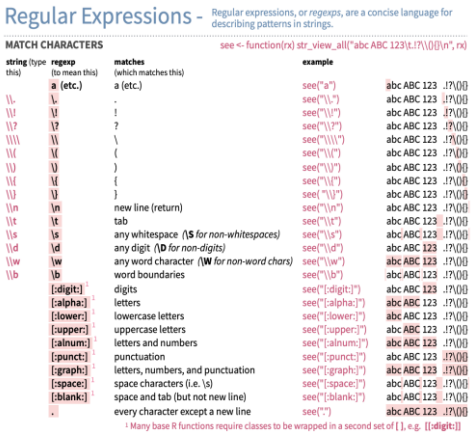


Figure 10. Manipulation des chaines de caractères dans R (1er extrait)

D'après cette notice, nous pouvons nous référer

- à n'importe quel chiffre (comme 1, 2, 3, etc.) en tant que `[:digit :]`.
- à n'importe quel signe de ponctuation comme `[:punct :]`.
- à un espace ou une tabulations via`[:blank :]`.

Si nous regardons de plus près la première colonne de notre tableau (en utilisant la fonction `pull()` du package `dplyr`), il y a aussi quelques grands espaces blancs, quelques valeurs numériques, des instances de `"\r\n"`, ainsi que quelques virgules et autres signes de ponctuation que nous voudrions supprimer de cette colonne.

```
table11.1a %>%
  pull(MHS_setting)

 [1] "SPECIALTY MENTAL HEALTH SERVICE1"
 [2] "Outpatient"
 [3] "Private Therapist, Psychologist,\r\n   Psychiatrist, Social Worker, or\r\n   Counselor"
 […] …
[18] "Child Welfare"
[19] "Foster Care or Therapeutic Foster Care"
[20] "SPECIALTY MENTAL HEALTH SERVICES\r\nAND EDUCATION, GENERAL MEDICINE\r\nOR CHILD WELFARE SERVICES1,2
,3"
```

Nous pouvons utiliser

- la fonction `str_remove_all()` (variante de la fonction `str_remove()`) du package `stringr` pour supprimer ces caractères indésirables de cette colonne en particulier. Cette fonction psupprime toutes les occurrences des caractères spécifiés dans chaque ligne plutôt que la première occurrence (ce que fait `str_remove()`).
- la fonction `mutate()`, nous spécifions que nous voulons changer cette colonne particulière et la remplacer par une version de cette colonne qui ne contient pas de caractères qui sont des chiffres, la chaîne `r\n` que nous avons vue, ou des signes de ponctuation.
- l'argument `string =`pour specifier la colonne `MHS_setting` en utilisant et que
- l'argument `pattern =` pour specifier les motifs à trouver et à supprimer
- le symbole | entre chaque motif qui permet de rechercher plusieurs motifs en même temps.

```
table11.1a %<>%
mutate(MHS_setting =
         str_remove_all(string = MHS_setting,
                        pattern = "[:digit:]|\r\n|[:punct:]|"))
head(table11.1a)

# A tibble: 6 × 18
  MHS_setting     `2002` `2003` `2004` `2005` `2006` `2007` `2008` `2009` `2010`
  <chr>           <chr>  <chr>  <chr>  <chr>  <chr>  <chr>  <chr>  <chr>  <chr>
1 SPECIALTY MENT… 2,898a 3,065a 3,348a 3,362a 3,255a 3,104a 3,129a 2,925a 2,920a
2 Outpatient      2,662a 2,795a 3,015a 3,048a 2,931a 2,787a 2,837a 2,650a 2,635a
… … … … … … … … … … …
5 Partial Day Ho… 440    425    439    449    471    416    374a   340a   362a
6 InHome Therapi… 693a   656a   762a   731a   719a   707a   716a   657a   674a
# ℹ 8 more variables: `2011` <chr>, `2012` <chr>, `2013` <chr>, `2014` <chr>,
#   `2015` <chr>, `2016` <chr>, `2017` <chr>, `2018` <chr>
```

Il y a parfois plus d'un espace. Aussi, nous voulons également remplacer les espaces par un seul espace. Nous pouvons spécifier que nous voulons que toute occurrence de 1 ou plus soit remplacée en utilisant la notation `{1,}`. Voir ici pour une explication de cette notation sur la notice d'information :

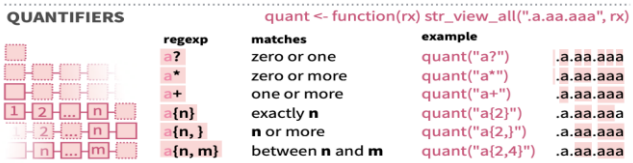


Figure 11. Manipulation des chaines de caractères dans R (2eme extrait)

Nous allons donc utiliser la fonction `str_replace_all()` du paquet `stringr`. Dans ce cas, nous devons également spécifier un remplacement avec l'argument `replacement =`. Nous l'utiliserons pour spécifier un espace.

```
table11.1a %<>%
mutate(MHS_setting =
         str_replace_all(string = MHS_setting,
                         pattern = "[:blank:]{1,}",
                         replacement = " "))
head(table11.1a)

# A tibble: 6 × 18
  MHS_setting     `2002` `2003` `2004` `2005` `2006` `2007` `2008` `2009` `2010`
  <chr>           <chr>  <chr>  <chr>  <chr>  <chr>  <chr>  <chr>  <chr>  <chr>
1 SPECIALTY MENT… 2,898a 3,065a 3,348a 3,362a 3,255a 3,104a 3,129a 2,925a 2,920a
2 Outpatient      2,662a 2,795a 3,015a 3,048a 2,931a 2,787a 2,837a 2,650a 2,635a
… … … … … … … … … … …
5 Partial Day Ho… 440    425    439    449    471    416    374a   340a   362a
6 InHome Therapi… 693a   656a   762a   731a   719a   707a   716a   657a   674a
# ℹ 8 more variables: `2011` <chr>, `2012` <chr>, `2013` <chr>, `2014` <chr>,
#   `2015` <chr>, `2016` <chr>, `2017` <chr>, `2018` <chr>
```

Ensuite, nous allons supprimer les valeurs `"a"` et les virgules du corps de la table en utilisant à nouveau la fonction `str_remove_all()`. Cette fois-ci, pour spécifier que nous voulons toutes les colonnes sauf la première colonne appelée `MHS_setting`, nous pouvons utiliser la fonction `across()` du package `dplyr`.

Cela nous permet de spécifier quelles colonnes nous voulons muter en utilisant l'argument `.cols =`. Nous pouvons sélectionner toutes les colonnes sauf la première appelée `MHS_setting` en utilisant le signe moins `-` devant le nom de la colonne.

```
table11.1a %<>%
  mutate(dplyr::across(.cols = -MHS_setting,
                       stringr::str_remove_all, "a|,"))

head(table11.1a)

# A tibble: 6 × 18
  MHS_setting     `2002` `2003` `2004` `2005` `2006` `2007` `2008` `2009` `2010`
  <chr>           <chr>  <chr>  <chr>  <chr>  <chr>  <chr>  <chr>  <chr>  <chr>
1 SPECIALTY MENT… 2898   3065   3348   3362   3255   3104   3129   2925   2920
2 Outpatient      2662   2795   3015   3048   2931   2787   2837   2650   2635
… … … … … … … … … … …
5 Partial Day Ho… 440    425    439    449    471    416    374    340    362
6 InHome Therapi… 693    656    762    731    719    707    716    657    674
# ℹ 8 more variables: `2011` <chr>, `2012` <chr>, `2013` <chr>, `2014` <chr>,
#   `2015` <chr>, `2016` <chr>, `2017` <chr>, `2018` <chr>
```

Notre tableau a bien meilleure allure !
Nous voulons aussi changer nos valeurs pour qu'elles soient numériques plutôt que des caractères afin de pouvoir les utiliser dans des fonctions mathématiques. Nous pouvons utiliser la fonction de base `as.numeric()`. Nous utiliserons à nouveau la fonction `across()` pour indiquer les variables que nous souhaitons modifier.

```
table11.1a %<>%
  mutate(across(.cols =-MHS_setting, as.numeric))

head(table11.1a)

# A tibble: 6 × 18
  MHS_setting     `2002` `2003` `2004` `2005` `2006` `2007` `2008` `2009` `2010`
  <chr>            <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
```

```
1 SPECIALTY MENT…   2898   3065   3348   3362   3255   3104   3129   2925   2920
2 Outpatient        2662   2795   3015   3048   2931   2787   2837   2650   2635
… … … … … … … … … … …
5 Partial Day Ho…    440    425    439    449    471    416    374    340    362
6 InHome Therapi…    693    656    762    731    719    707    716    657    674
# ℹ 8 more variables: `2011` <dbl>, `2012` <dbl>, `2013` <dbl>, `2014` <dbl>,
#   `2015` <dbl>, `2016` <dbl>, `2017` <dbl>, `2018` <dbl>
```

Nous aimerions également

- ajouter une variable `type` et `subtype`, qui spécifie les catégories d'environnements où les traitements ont été reçus
- supprimer quelques lignes qui sont complètement vides. Il s'agit des lignes dont les valeurs de la première colonne sont "Médecine générale", "Justice pour mineurs" et "Protection de l'enfance". Si nous regardons le site web, nous pouvons voir qu'il s'agissait de lignes principales sans données.

Tout d'abord, ajoutons les variables `type` et `subtype` en utilisant la fonction `mutate`.

```
table11.1a %<>%
  mutate(type = c(rep("Specialty", 9), rep("Nonspecialty", 11))) %>%
  mutate(subtype = c("Specialty_total", ("Outpatient", 5),
                     rep("Inpatient", 3), Nonspecialty_total",
                     rep("Education", 3), rep("General_medicine", 2),
                     rep("Juvenile_Justice", 2), rep("Child_Welfare", 2), "combination"))
```

Ajoutons également une variable avec des noms plus courts pour les étiquettes dans les graphiques et les résultats statistiques.

```
table11.1a %<>%
  mutate(short_label = c("Specialty total", "Outpatient total", "Therapist",
                         "Clinic", "Day program", "In-home Therapist", "Inpatient total",
                         "Hospital", "Residential Center", "Nonspecialty total",
                         "School total", "School Therapist", "School Program",
                         "General Medicine", "Family Dr", "Justice System", "Justice System",
                         "Welfare", "Fostercare", "Specialty Combination"))
```

Nous pouvons

- supprimer les lignes vides en utilisant la fonction `filter()` du package `dplyr`.
- spécifier que nous ne voulons pas garder ces lignes en utilisant l'opérateur `!=` not equal to.

```
table11.1a %<>%
  dplyr::filter(MHS_setting != "General_Medicine") %>%
  dplyr::filter(MHS_setting != "Juvenile_Justice") %>%
  dplyr::filter(MHS_setting != "Child_Welfare")
head(table11.1a)

# A tibble: 6 × 21
  MHS_setting     `2002` `2003` `2004` `2005` `2006` `2007` `2008` `2009` `2010`
  <chr>            <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
1 SPECIALTY MENT…   2898   3065   3348   3362   3255   3104   3129   2925   2920
2 Outpatient        2662   2795   3015   3048   2931   2787   2837   2650   2635
… … … … … … … … … … …
5 Partial Day Ho…    440    425    439    449    471    416    374    340    362
6 InHome Therapi…    693    656    762    731    719    707    716    657    674
# ℹ 11 more variables: `2011` <dbl>, `2012` <dbl>, `2013` <dbl>, `2014` <dbl>,
#   `2015` <dbl>, `2016` <dbl>, `2017` <dbl>, `2018` <dbl>, type <chr>,
#   subtype <chr>, short_label <chr>
```

Enfin, nous souhaitons modifier la forme de notre tableau afin de disposer d'une nouvelle colonne représentant l'année et d'une nouvelle colonne représentant la valeur de cette année. Pour ce faire, nous allons allonger notre tableau, ce qui signifie qu'il aura moins de colonnes et plus de lignes. Voir ici pour plus d'informations sur les différents formats de tableaux, généralement appelés "large" et "long" ou parfois "étroit".
Nous allons utiliser la fonction `pivot_longer()` du package `tidyr` pour changer la forme de notre tableau. Il y a 3 arguments principaux dans cette fonction :

- `cols` - qui spécifie les colonnes à réduire
- `names_to` - qui spécifie le nom de la nouvelle colonne qui sera créée et contiendra les noms des colonnes que vous réduisez.
- `values_to` - qui spécifie le nom de la nouvelle colonne qui sera créée et contiendra les valeurs des colonnes que vous réduisez.

Pour spécifier que nous voulons réduire toutes les colonnes qui ont des valeurs d'année, nous pouvons choisir celles qui contiennent la chaîne `"20"` en utilisant la fonction `contains()` de `dplyr`.
Enfin, nous allons rendre la variable `Year` numérique.

Utilisons d'abord la fonction de base `dim()` pour voir les dimensions avant et après l'utilisation de `pivot_longer()`.

```
dim(table11.1a)

[1] 20 21
```

```
table11.1a %<>%
  tidyr::pivot_longer(cols = contains("20"),
                      names_to = "Year",
                      values_to = "Number") %>%
  mutate(Year = as.numeric(Year))
dim(table11.1a)

[1] 340   6

head(table11.1a)

# A tibble: 6 × 6
  MHS_setting                      type      subtype     short_label  Year Number
  <chr>                            <chr>     <chr>       <chr>       <dbl>  <dbl>
1 SPECIALTY MENTAL HEALTH SERVICE Specialty Specialty_… Specialty …  2002   2898
… … … … … … …
6 SPECIALTY MENTAL HEALTH SERVICE Specialty Specialty_… Specialty …  2007   3104
```

Nous constatons que notre tableau est maintenant beaucoup plus long - puisque nous avons 340 lignes.

Question 2 : pourquoi avons-nous 340 lignes ?
Votre réponse : ….

3.4.3. Ecriture d'une fonction de data wrangling pour les tableaux 11.1A-B

Nous voyons maintenant que les variables `Year` et `Number` sont de classe double à cause du `<dbl>` sous le nom de la colonne. Le tableau suivant (Tableau11.1B) est très similaire au Tableau11.1A, tandis que les autres tableaux contiennent des informations démographiques.

Maintenant, nous voulons structurer le tableau 11.1B. Pour ce faire, nous pouvons simplement exécuter ces dernieres étapes en utilisant le `table11.1B` comme entrée. Cependant, dans un but éducatif, nous allons vous montrer comment vous pourriez créer une fonction si nous avions plusieurs tableaux similaires à manipuler.

Pour rappel, voici toutes les étapes que nous avons effectuées pour structurer le `tableau11.1a` :

```
table11.1a %<>%
# make the table into a tibble
  dplyr::as_tibble() %>%
# remove the last row by only keeping the first through the second to last
  slice(1:(n() - 1)) %>%
# make the "nc" values "NA" instead
  dplyr::na_if("nc") %>%
  dplyr::na_if("--") %>%
  dplyr::na_if("") %>%
  dplyr::na_if("*") %>%
# rename the column to the shorter MHS_setting name
  dplyr::rename(MHS_setting =
                  `Setting Where Mental Health ServiceWas Received`) %>%
# remove numbers, carriage return, new lines, and punctuation marks from the values for the MHS_setting c
olumn
  mutate(MHS_setting =
         str_remove_all(string = MHS_setting,
                        pattern = "[:digit:]|\r\n|[:punct:]|")) %>%
# replace the white spaces with a single space
  mutate(MHS_setting =
         str_replace_all(string = MHS_setting,
```

```
                      pattern = "[:blank:]{1,}",
                    replacement = " ")) %>%
# remove "a" and commas from the values in the columns except the column called MHS_setting
  mutate(dplyr::across(.cols = -MHS_setting,
                stringr::str_remove_all, "a|,")) %>%
# make those columns numeric
  mutate(across(-MHS_setting, as.numeric)) %>%
# create a new type column with the values: "Specialty repeated 9 times followed by "Nonspecialty" repeat
ed 11 times
  mutate(type = c(rep("Specialty", 9), rep("Nonspecialty", 11))) %>%
# create a new variable called subtype
  mutate(subtype =c("Specialty_total",
                    rep("Outpatient", 5),
                    rep("Inpatient", 3),
                    "Nonspecialty_total",
                    rep("Education", 3),
                    rep("General_medicine", 2),
                    rep("Juvenile_Justice", 2),
                    rep("Child_Welfare", 2),
                    "combination")) %>%
# create a new variabel called short_label
  mutate(short_label = c("Specialty total", "Outpatient total",
                         "Therapist", "Clinic", "Day program",
                         "In-home Therapist", "Inpatient total",
                         "Hospital", "Residential Center",
                         "Nonspecialty total", "School total",
                         "School Therapist", "School Program",
                         "General Medicine", "Family Dr",
                         "Justice System", "Justice System",
                         "Welfare", "Fostercare",
                         "Specialty Combination")) %>%
# remove rows with General_Medicine as the value in the MHS_setting column because it is empty
  dplyr::filter(MHS_setting != "General_Medicine") %>%
# remove rows with Juvenile_Justice as the value in the MHS_setting column because it is empty
  dplyr::filter(MHS_setting != "Juvenile_Justice") %>%
# remove rows with Child_Welfare as the value in the MHS_setting column because it is empty
  dplyr::filter(MHS_setting != "Child_Welfare") %>%
# make the table in long format
    tidyr::pivot_longer(cols = contains("20"),
                        names_to = "Year",
                        values_to = "Number") %>%
# make the new Year variable to be numeric
   mutate(Year = as.numeric(Year))
```

La dernière fois que nous avons écrit une fonction dans cette étude de cas, nous n'avions qu'une seule entrée. Cette fois-ci, nous avons plusieurs entrées : la table que nous voulons manipuler comme `TABLE`, un nouveau nom pour la première colonne appelé `new_col`, et une entrée appelée `pivot_col`, qui sera le nom de la colonne qui sera créée après le pivotement et qui prendra les valeurs de chacune des années.

```
# Rien a faire dans ce chunck
function(TABLE, new_col, pivot_col){
    <add code here>
  }
```

Nous voulons rendre notre fonction flexible afin qu'elle puisse prendre n'importe quelle valeur pour le nom de la première colonne. Pour sélectionner la première colonne, nous utiliserons le code suivant, où la fonction de base `names()` récupère les noms des colonnes de l'entrée `TABLE`, ce qui est indiqué par le `.` et ensuite pour obtenir seulement le premier nom, le `[1]` est utilisé.

```
# Rien a faire dans ce chunck
function(TABLE, new_col, pivot_col){
  dplyr::as_tibble(TABLE) %>%
  #additional steps
  names(.)[1]
  #additional steps
}
```

Et pour renommer le avec l'entrée `new_col` de la fonction, nous pouvons faire ceci :

```
# Rien a faire dans ce chunck
```

```
function(TABLE, new_col, pivot_col){
  dplyr::as_tibble(TABLE) %>%
  #additional steps
  rename({{new_col}} := names(.)[1])
  #additional steps
}
```

Les doubles crochets `{{}}` nous permettent d'utiliser l'entrée de la fonction appelée `new_col` à l'intérieur de la fonction.
Voir ici pour des informations sur l'opérateur `:=` colon-equals. Cet opérateur est plus souple que l'opérateur normal `=`. Il autorise les expressions des deux côtés, ce qui nous permet d'utiliser une expression (les valeurs de `new_col`) comme valeur d'entrée pour les expressions qui suivent l'opérateur `:=`.
Nous allons également ajouter du code pour supprimer toutes les lignes qui n'ont que des valeurs `NA` d'une manière plus flexible, de sorte que nous n'ayons pas besoin de savoir quelles lignes à l'avance, avec l'aide des fonctions `filter()` et `select()` du package `dplyr`.

Nous allons calculer la somme du nombre de valeurs `NA` sur les lignes des colonnes pour chaque année en utilisant la fonction de base `rowSums()` comme suit :

```
rowSums(is.na(.))
```

Cependant, pour ce faire, nous devons d'abord sélectionner uniquement les colonnes numériques en utilisant `select(., is.numeric)` : `select(., is.numeric)`, où le `.` fait référence au tableau après toutes les étapes de manipulation précédentes dans notre fonction. Il est important de noter que tout ceci doit se produire après avoir converti les valeurs de chaque année en valeurs numériques.

Quoi qu'il en soit, cela ressemble à ceci :

```
# Rien a faire dans ce chunck
rowSums(is.na(select(., is.numeric)))`.
```

Enfin, nous comparons ce résultat au nombre de colonnes numériques en utilisant `length(select(., is.numeric)))` : `length(select(., is.numeric)))`, avec l'idée que si le nombre de valeurs `NA` est inférieur au nombre de colonnes qui pourraient avoir des valeurs `NA`, alors nous savons qu'il ne s'agit pas d'une ligne vide.

Globalement, cela ressemblerait à quelque chose comme ceci après avoir effectué une étape pour convertir les colonnes en valeurs numériques comme nous l'avons fait précédemment :

```
# Rien a faire dans ce chunck
function(TABLE, new_col, pivot_col){
  # previous similar steps to modify and make table values numeric
    filter(rowSums(is.na(select(., is.numeric))) <
             length(select(., is.numeric)))
  }
```

En rassemblant tout ce que nous avons fait précédemment pour `table11.1a` et ces nouvelles étapes flexibles, nous pouvons créer cette fonction :

```
data_prep_settings <- function(TABLE, new_col, pivot_col){
# make the table a tibble
  dplyr::as_tibble(TABLE) %>%
# remove the last row
    slice(1:(n() - 1)) %>%
# make "nc" values NA etc.
  mutate(across(everything(), ~na_if(.x, "nc"))) %>%
  mutate(across(everything(), ~na_if(.x, "--"))) %>%
  mutate(across(everything(), ~na_if(.x, ""))) %>%
  mutate(across(everything(), ~na_if(.x, "*")))%>%
# rename the first column (names(.)[1]) to be what was specified with the new_col argument
    rename({{new_col}} := names(.)[1]) %>%
# remove the numbers and punctuation marks and carriage returns (\r) and new lines (\n) from the first co
lumn
    mutate({{new_col}} :=
        str_remove_all(string = pull(., {{new_col}}),
                       pattern = "[:digit:]|\r\n|[:punct:]|")) %>%
# replace white spaces with a single space
    mutate({{new_col}} :=
```

```
        str_replace_all(string =pull(., {{new_col}}),
                        pattern = "[:blank:]{1,}",
                        replacement = " ")) %>%
# remove "a" and , from the values for the columns that are not the first column (called new_col)
    mutate(dplyr::across(.cols = -{{new_col}},
                         stringr::str_remove_all, "a|,")) %>%
# make these columns numeric  (all the columns but the first column)
    mutate(across(-{{new_col}}, as.numeric)) %>%
# make a new variable called type with 9 values that are Specialty followed by 11 values of Nonspecialty
    mutate(type = c(rep("Specialty", 9), rep("Nonspecialty", 11))) %>%
# make a new variable called subtype with the following values repeated various times
    mutate(subtype = c("Specialty_total",
                       rep("Outpatient", 5),
                       rep("Inpatient", 3),
                       "Nonspecialty_total",
                       rep("Education", 3),
                       rep("General_medicine", 2),
                       rep("Juvenile_Justice", 2),
                       rep("Child_Welfare", 2),
                       "combination")) %>%
# make a new variable called short_label to use as labels for plots for the data
    mutate(short_label = c("Specialty total", "Outpatient total",
                           "Therapist", "Clinic", "Day program",
                           "In-home Therapist", "Inpatient total",
                           "Hospital", "Residential Center",
                           "Nonspecialty total", "School total",
                           "School Therapist", "School Program",
                           "General Medicine", "Family Dr",
                           "Justice System", "Justice System",
                           "Welfare", "Fostercare",
                           "Specialty Combination")) %>%
# remove rows were all the values are NA -
# the number of `NA` values for a row should be less than the number of columns that could have `NA` valu
es
  filter(rowSums(is.na(select(., is.numeric))) <
           length(select(., is.numeric))) %>%
# make the table into long format so that the year columns are collapsed together
# the new value column is called what was specified with the "pivot_col" argument
  pivot_longer(cols = contains("20"),
               names_to = "Year",
               values_to = pivot_col)%>%
# make the new "Year" variable numeric
  mutate(Year = as.numeric(Year))
}
```

Appliquons la fonction à `table11.1b`.

```
table11.1b <-
  data_prep_settings(TABLE = table11.1b,
                     new_col = "MHS_setting",
                     pivot_col = "Percent")
head(table11.1b)
```

```
# A tibble: 6 × 6
  MHS_setting                     type      subtype    short_label  Year Percent
  <chr>                           <chr>     <chr>      <chr>       <dbl>   <dbl>
1 SPECIALTY MENTAL HEALTH SERVICE Specialty Specialty… Specialty …  2002    11.8
2 SPECIALTY MENTAL HEALTH SERVICE Specialty Specialty… Specialty …  2003    12.4
3 SPECIALTY MENTAL HEALTH SERVICE Specialty Specialty… Specialty …  2004    13.4
4 SPECIALTY MENTAL HEALTH SERVICE Specialty Specialty… Specialty …  2005    13.4
5 SPECIALTY MENTAL HEALTH SERVICE Specialty Specialty… Specialty …  2006    13
6 SPECIALTY MENTAL HEALTH SERVICE Specialty Specialty… Specialty …  2007    12.4
```

Nous disposons à présent de données claires sur les nombres et les pourcentages de jeunes qui ont connu un EDM et qui ont reçu un traitement contre la dépression.
Qu'en est-il des tableaux suivants ?

3.4.4. Cas des tables démographiques

Tous les autres tableaux contiennent des informations démographiques et ont une même structure. Dans ces tableaux, nous avons des groupes d'âge dans notre première colonne, nous ne voulons donc plus supprimer les chiffres ou les signes de ponctuation, nous devons donc modifier un peu notre fonction pour supprimer cette étape.
Nous voulons également

- ajouter le mot `Age` et un trait de soulignement devant le groupe d'âge listé dans les tableaux. Nous pouvons utiliser la fonction `str_replace()` du package `stringr`, parce que maintenant nous voulons seulement remplacer la première instance de `1` par `Age : 1`.
- remplacer le nom de la première colonne par `Demographic` pour toutes les tables.
- créer une nouvelle variable qui listera les sous-groupes.
- manipuler les données seulement jusqu'au moment où nous changerons la forme des données, de façon que nous puissions d'abord vérifier l'apparence des données.

Mettons tout cela ensemble dans une fonction `data_dem_settings()` :

```
data_dem_settings <- function(TABLE){
  # make the table a tibble
  dplyr::as_tibble(TABLE) %>%
  # Remove the last row - keep only the 1st through 2nd to last rows
  slice(1:(n()-1)) %>%
  # change the values from "nc, --" etc to NA
  mutate(across(everything(), ~na_if(.x, "nc"))) %>%
  mutate(across(everything(), ~na_if(.x, "--"))) %>%
  mutate(across(everything(), ~na_if(.x, ""))) %>%
  mutate(across(everything(), ~na_if(.x, "*"))) %>%
  # rename the first column to be "Demographic"
  rename(Demographic := names(.)[1]) %>%
  # replace white spaces form the values of the "Demographic" variable with a single space
  mutate(Demographic := str_replace_all(string = pull(., Demographic),
                                        pattern = "[:blank:]{1,}",
                                        replacement = " ")) %>%
  # replace values where there is a "1" in the "Demographic" variable to be "Age: 1"
  mutate(Demographic = str_replace(string = Demographic,
                                   pattern = "1",
                                   replacement = "Age: 1")) %>%
  # create a new variable called subgroup
  mutate(subgroup = c("Total", rep("Age", 4),
                      rep("Gender", 3), rep("Race/Ethnicity", 9))) %>%
  # remove "a" and commas from the variables that have column names with "20" in them
  mutate(dplyr::across(.cols = contains("20"),
                       stringr::str_remove_all, "a|,")) %>%
  # make the variables with "20" in the names (the year variables) to be numeric
  mutate(across(contains("20"), as.numeric)) %>%
  # remove empty rows - rows were the number of NA values is equal to the number of numeric columns
  filter(rowSums(is.na(select(., is.numeric))) < length(select(., is.numeric)))
  }
```

Maintenant, nous utilisons la fonction `data_dem_settings()` pour structurer le prochain ensemble de tables. Nous allons également ajouter une colonne pour décrire le sujet des données, ce qui sera utile pour fusionner les données plus tard.

```
table11.2a <- data_dem_settings(TABLE = table11.2a)
table11.2a %<>% mutate(data_type = "Major_Depressive_Episode")
head(table11.2a)

table11.2b <- data_dem_settings(TABLE = table11.2b)
table11.2b %<>% mutate(data_type = "Major_Depressive_Episode")
head(table11.2b)

table11.3a <- data_dem_settings(TABLE = table11.3a)
table11.3a %<>% mutate(data_type = "Severe_Major_Depressive_Episode")
head(table11.3a)

table11.3b <- data_dem_settings(TABLE = table11.3b)
table11.3b %<>% mutate(data_type = "Severe_Major_Depressive_Episode")
head(table11.3b)
```

```
table11.4a <- data_dem_settings(TABLE = table11.4a)
table11.4a %<>% mutate(data_type = "Treatment")
head(table11.4a)

table11.4b <- data_dem_settings(TABLE = table11.4b)
table11.4b %<>% mutate(data_type = "Treatment")
head(table11.4b)
```

Bien : tous les tableaux démographiques semblent corrects !
Il est recommandé de vérifier régulièrement vos données pour vous assurer qu'elles sont conformes à vos attentes.

3.5. Vérification de la conformité des données

Maintenant, créons une fonction pour vérifier que nos données sont structurées conformément. Nous avons plusieurs tableaux, ce qui pourrait rendre cette tâche un peu difficile.

Tout d'abord, assurons-nous que nos tables sont des tibbles en utilisant la fonction `is_tibble()` du paquet `tibble`. Nous pouvons utiliser la fonction `case_when()` pour nous donner un message si la valeur de la fonction `is_tibble()` est `TRUE` - c'est le message après le premier `~` et un message différent pour tous les autres cas utilisant `TRUE` suivi à nouveau de `~` et d'un message utile sur les données.

```
data_dem_check <- function(TABLE){
  # check that the data is a tibble
  case_when(is_tibble(TABLE) ~ "Good!",
                        TRUE ~ "Not a tibble")
}
```

Nous allons maintenant essayer cette méthode sur des données dont nous savons avec certitude qu'il s'agit d'un tibble (tableau 11.1a) et sur des données dont nous savons avec certitude qu'il ne s'agit pas d'un tibble.

```
test_that_should_fail <- c(1,2,3)
class(test_that_should_fail)
class(table11.1a)
```

```
data_dem_check(test_that_should_fail)
data_dem_check(table11.1a)
```

On dirait que ça marche.
Créons maintenant d'autres fonctions pour effectuer des contrôles supplémentaires sur les données. Ensuite, vérifions que la légende a été supprimée. Pour ce faire, nous allons nous assurer qu'il n'y a pas de `-- = non disponible'` (car cela faisait partie de la légende) dans la dernière ligne en utilisant `str_detect()` pour chercher et `slice(n())` pour regarder la dernière ligne spécifiquement.

Reprenons tout d'abord l'aspect de la légende :

```
Legend
data_dem_check <- function(TABLE){
  # check that the last row does not contain "--" by..
   #first grabbing only the last row
   #pulling one of the years
  case_when(TABLE %>% slice(n()) %>% pull(`2018`) %>%
  # if it is detected print
            str_detect(pattern = "-- = not available")  ~ "Legend might still be there",
                                                  TRUE ~ "Good!")
}
data_dem_check(table11.4a)
```

Nous allons maintenant les rassembler dans un nouveau tableau :

```
data_dem_check <- function(TABLE){
tibble(tibble_check = case_when(is_tibble(table11.4a) ~ "Good!",
                                                 TRUE ~ "Not a tibble"),
       legend_check = case_when(table11.4a %>% slice(n()) %>%pull(`2004`) %>%
   # if it is detected print
                          str_detect(pattern = "--")  ~ "Legend might still be there",
                                                 TRUE ~ "Good!"))
}
data_dem_check(table11.4a)
```

```
# A tibble: 1 × 2
  tibble_check legend_check
  <chr>        <chr>
1 Good!        Good!
```

Notez ici que nous allons faire en sorte que toutes les vérifications positives aient la même valeur de `Good!`. Cela nous permettra de vérifier plus tard que toutes les vérifications ont réussi.

Maintenant, nous allons écrire une fonction pour vérifier si l'une des valeurs qui étaient `nc`, `*`, `--`, ou si elles ont été converties en `NA`. Nous pouvons vérifier la présence d'une valeur dans un tibble entier en utilisant la fonction de base `any()`.

```
data_dem_check <- function(TABLE){
  case_when(any(str_detect(TABLE, pattern = "nc|\\*|--"))
  # if it is detected, print this:
         ~ "NA not fixed",
  # if not detected, print this:
    TRUE ~ "Good!")
}
data_dem_check(table11.4a)
```

```
[1] "Good!"
```

Nous allons maintenant vérifier que la première variable s'appelle `Demographic`.

```
data_dem_check <- function(TABLE){
  case_when(names(TABLE)[1] == "Demographic" ~ "Good!",
                                        TRUE ~ "check first column")
  }
data_dem_check(table11.4a)
```

```
[1] "Good!"
```

Vérifions maintenant qu'il n'y a pas d'espaces blancs plus grands qu'un espace. Nous pouvons utiliser `[:blank:]{2,}` pour indiquer deux espaces blancs ou plus.

```
data_dem_check <- function(TABLE){
  case_when(any(str_detect(TABLE, pattern = "[:blank:]{2,}"))
         ~ "white spaces not fixed",
    TRUE ~ "Good!")
}
data_dem_check(table11.4a)
```

```
[1] "Good!"
```

Vérifions maintenant que toutes les valeurs d'âge commencent par `Age  :` pour la variable démographique. Nous pouvons utiliser `^` pour regarder le début des chaînes de caractères dans la variable `Demographic`. Aucune ne doit plus commencer par `1`. Nous pouvons donc utiliser `^1` pour vérifier si une chaîne de caractères commence par un `1`.

```
data_dem_check <- function(TABLE){
  case_when(any(str_detect(pull(TABLE,Demographic), pattern = "^1"))
  # if it is detected print
         ~ "Age data not fixed!",
    TRUE ~ "Good!")
}
data_dem_check(table11.4a)
```

```
[1] "Good!"
```

Vérifions maintenant que nous disposons d'une variable appelée sous-groupe

```
data_dem_check <- function(TABLE){
  case_when(any(names(TABLE) == "subgroup")
  # if it is detected print
         ~ "Good",
    TRUE ~ "No subgroup variable!")
}
data_dem_check(table11.4a)
```

```
[1] "Good"
```

Nous allons ensuite vérifier que les variables relatives à l'année ne contiennent pas de "a" ou de ",". Pour ce faire, au lieu de sélectionner les colonnes dont les noms sont des années, nous n'inclurons pas les colonnes qui ne sont pas des années. Nous utiliserons également la fonction `map_df` du paquetage `purrr` pour vérifier la détection des virgules et des "a" pour chaque colonne séparément. Typiquement, ce n'est pas nécessaire car tant que nous ne vérifions pas les virgules, cela devrait fonctionner. Cependant, `str_detect()` va forcer les données à être vectorisées et pour ce faire, il va ajouter des virgules à nos données ! Puisque nous cherchons des virgules, cela nous amènerait à détecter des virgules indépendamment du fait qu'elles soient présentes dans nos données. Les fonctions `map` du package `purrr` nous permettent d'exécuter des fonctions sur plusieurs colonnes de tibbles. La fonction `map_df()` préserve la structure du cadre de données, sinon nous nous retrouvons avec une liste, ce qui serait légèrement plus difficile à travailler. Cela créera un cadre de données de valeurs "VRAI" et "FAUX".

Nous pouvons alors faire la somme de chaque ligne (`FALSE` est évalué comme 0, `TRUE` est évalué comme 1). Ensuite, pour obtenir une valeur unique pour notre fonction `case_when()`, nous allons additionner les sommes des lignes. Nous ne devrions pas avoir de valeurs avec "a" ou "," donc quand nous faisons cette vérification, la somme devrait être égale à 0. Pour envoyer les données dans la fonction `map_df()` et ensuite dans `str_detect()`, nous devons utiliser les notations `~` et `.x`. Ainsi, le `.X` représente les colonnes sélectionnées dans la table qui seront envoyées dans `str_detect`. Le `~` indique la fonction que nous allons utiliser sur chaque colonne.

```
data_dem_check <- function(TABLE){
  case_when(
   TABLE%>% select(-Demographic, -subgroup, -data_type) %>%
     map_df(~str_detect(.x, pattern ="a|,")) %>%
     rowSums(na.rm = TRUE) %>%
     sum() == 0
         ~ "Good!",
    TRUE ~ "There may be commas or the letter a in the year columns!")
}
data_dem_check(table11.4a)
```

Nous allons maintenant vérifier que les variables relatives à l'année sont numériques.

```
data_dem_check <- function(TABLE){
  case_when(sum(map_dbl(TABLE, is.numeric))== sum(str_count(names(TABLE), "20"))
  # if it is detected print
         ~ "Good!",
    TRUE ~ "Variables are not numeric!")
}
data_dem_check(table11.4a)
```

Enfin, nous nous assurerons qu'il n'y a pas de lignes où toutes les colonnes de l'année ont des valeurs `NA`.

```
data_dem_check <- function(TABLE){
  case_when(nrow(TABLE %>%filter(rowSums(is.na(select(., is.numeric))) > length(select(., is.numeric))))
 >0
  # if it is detected print
         ~ "There are empty rows ",
    TRUE ~ "Good!")
}
data_dem_check(table11.4a)
```

Rassemblons maintenant toutes nos fonctions de vérification en une seule grande fonction de vérification des données. Remarquez que si le résultat est bon pour chaque vérification, il en résulte une valeur de "Good !". Nous pouvons alors utiliser la fonction de base `all()` pour vérifier que toutes les valeurs dans le tibble `results` qui est créé pendant notre fonction globale donne une valeur de "Good !".

Nous pouvons utiliser la fonction de base `ifelse` pour donner notre résultat de la même manière que nous avons utilisé `case_when()`. Si toutes les valeurs de chaque vérification sont "Good !" alors nous obtiendrons "Data looks good !", sinon nous verrons tous les résultats des vérifications. Il y a une fonction `if_else()` dans `dplyr` mais elle ne produit que des chaînes de caractères, donc cela ne fonctionnerait pas pour montrer quels contrôles ont échoué quand toutes les valeurs n'étaient pas "Good !".

```
data_dem_check <- function(TABLE){
results <-tibble(tibble_check = case_when(is_tibble(TABLE) ~ "Good!",
                                          TRUE ~ "Not a tibble"),
       legend_check = case_when(TABLE %>% slice(n()) %>%pull(`2018`) %>%
   # if it is detected print
```

```
                          str_detect(pattern = "--")  ~ "Legend might still be there",
                                                 TRUE ~ "Good!"),
       NAs_check = case_when(any(str_detect(TABLE, pattern = "nc"))
                                                      ~ "NA not fixed",
                                                 TRUE ~ "Good!"),
  firstcol_check = case_when(names(TABLE)[1] == "Demographic"
                                                      ~ "Good!",
                                                 TRUE ~ "check first column"),
  white_space_check = case_when(any(str_detect(TABLE, pattern = "[:blank:]{2,}"))
                                                      ~ "white spaces not fixed",
                                                 TRUE ~ "Good!"),
  age_data_check = case_when(any(str_detect(pull(TABLE,Demographic), pattern = "^1"))
                                                      ~ "Age data not fixed!",
                                                 TRUE ~ "Good!"),
  subgroup_check = case_when(any(names(TABLE) == "subgroup")
                                                      ~ "Good!",
                                                 TRUE ~ "No subgroup variable!"),
    a_comma_check = case_when(TABLE%>% select(-Demographic, -subgroup, -data_type) %>%
                                      map_df(~(str_detect(.x, pattern ="a|,"))) %>%
                                           rowSums(na.rm = TRUE) %>%
                                                sum() == 0
                                                       ~ "Good!",
                                                  TRUE ~ "There may be commas or the letter a in the year
columns!"),
numeric_check = case_when(sum(map_dbl(TABLE, is.numeric))== sum(str_count(names(TABLE), "20"))
                                                     ~ "Good!",
                                                TRUE ~ "Variables are not numeric!"),
  empty_row_check = case_when(nrow(TABLE %>%filter(rowSums(is.na(select(., is.numeric))) >
                                                     length(select(., is.numeric)))) >0
                                                     ~ "There are empty rows ",
                                                TRUE ~ "Good!"))

ifelse(all(results == "Good!"),
       "Data looks good!", glimpse(results))
}

data_dem_check(table11.4a)

[1] "Data looks good!"
```

Maintenant, vérifions nos tableaux démographiques. Nous pouvons utiliser la fonction générale `map()` du package `purrr` pour vérifier efficacement toutes nos tables démographiques. Nous allons créer une liste des noms des tableaux et ensuite appliquer la fonction `data_dem_check()` que nous avons écrite à chaque tableau en utilisant `map()`.

```
tables_tocheck <-list(table11.2a, table11.2b, table11.3a, table11.3b, table11.4a, table11.4b)
tables_tocheck %>% map(data_dem_check)

[[1]]
[1] "Data looks good!"

[[6]]
[1] "Data looks good!"
```

Bien : maintenant que nous avons vérifié nos données, rassemblons-les. Combinons les données de comptage (les tableaux "a") et les données de pourcentage (les tableaux "b") en utilisant :

- la fonction `bind_rows()` du package `dplyr` pour concaténer les tableaux.
- la fonction `distinct()` du package `dplyr` pour vérifier que nous avons tous les types de données dans ces tableaux plus grands.

```
counts <- dplyr::bind_rows(table11.2a, table11.3a, table11.4a)
percents <- bind_rows(table11.2b, table11.3b, table11.4b)

counts %>% dplyr::distinct(data_type)

# A tibble: 3 × 1
  data_type
  <chr>
1 Major_Depressive_Episode
2 Severe_Major_Depressive_Episode
3 Treatment
```

```
percents %>% distinct(data_type)

# A tibble: 3 × 1
  data_type
  <chr>
1 Major_Depressive_Episode
2 Severe_Major_Depressive_Episode
3 Treatment
```

Bien : nous allons maintenant reformater les données `counts` et `percents` pour qu'elles soient au format long en utilisant à nouveau `pivot_longer()`.

```
counts %<>%
  pivot_longer(cols = contains("20"),
               names_to = "Year",
               values_to = "Number") %>%
  mutate(Year = as.numeric(Year))

percents %<>%
  pivot_longer(cols = contains("20"),
               names_to = "Year",
               values_to = "Percent") %>%
  mutate(Year = as.numeric(Year))

glimpse(counts)
glimpse(percents)
```

Notez également que certains groupes sont abrégés en AIAN et NHOPI.

```
percents %>%
  distinct(Demographic)%>%
  pull(Demographic)
```

En utilisant les définitions du Census Bureau :

- AIAN désigne les Indiens d'Amérique et les autochtones de l'Alaska.
- NHOPI désigne Native Hawaiian or Other Pacific Islander (Indien d'Hawaï ou autre insulaire du Pacifique)

Mettons à jour nos données pour refléter ces définitions en utilisant la fonction `str_replace()`.

```
percents %<>% mutate(Demographic = str_replace(string = Demographic,
                                               pattern = "AIAN",
                                           replacement = "American Indian and Alaska Native"))

percents %<>% mutate(Demographic =  str_replace(string = Demographic,
                                               pattern = "NHOPI",
                                           replacement = "Native Hawaiian or Other Pacific Islander"))

counts %<>% mutate(Demographic = str_replace(string = Demographic,
                                               pattern = "AIAN",
                                           replacement = "American Indian and Alaska Native"))

counts %<>% mutate(Demographic =  str_replace(string = Demographic,
                                               pattern = "NHOPI",
                                           replacement = "Native Hawaiian or Other Pacific Islander"))
```

Vérifions que cela a fonctionné.

```
percents %>%
  distinct(Demographic)%>%
  pull(Demographic)

counts %>%
  distinct(Demographic)%>%
  pull(Demographic)
```

Nous avons fini de manipuler les données et nous sommes prêts à procéder à notre analyse.
Sauvegardons d'abord nos données dans un fichier rda et dans un fichier csv dans un répertoire appelé *data>wrangled*.

```
# Ensure the directory exists
fs::dir_create(here("data", "wrangled"))
save(percents, counts, table11.1a, table11.1b,
     file = here::here("data", "wrangled", "wrangled_data.rda"))
```

```
readr::write_csv(percents, path = here::here("data", "wrangled", "percents.csv"))
readr::write_csv(counts, path = here::here("data", "wrangled", "counts.csv"))
readr::write_csv(table11.1a, path = here::here("data", "wrangled", "table11.1a.csv"))
readr::write_csv(table11.1b, path = here::here("data", "wrangled", "table11.1b.csv"))
```

4. RESULTATS : VISUALISATION DES DONNEES

4.1. (Re-)chargement des données

Si vous reprenez le TP, vous pouvez charger vos données structurées d'un des trois manieres suivantes :

```
load(file = here::here("data", "wrangled", "wrangled_data.rda"))
```

Vous pouvez autrement charger les données structurées :

```
library(OCSdata)
wrangled_rda("ocs-bp-youth-mental-health", outpath = getwd())
load(here::here("OCSdata", "data", "wrangled", "wrangled_data.rda"))
```

Vous pouvez autrement telecharger les données structurées ici ou ici. Placez ce fichier dans votre répertoire de travail actuel dans un sous-répertoire appelé "data>wrangled"

```
load(here::here("data", "wrangled", "wrangled_data.rda"))
```

Avant de commencer les analyses, rappelons les questions principales de l'étude

1. Comment les taux de dépression chez les jeunes Américains ont-ils évolué depuis 2004, selon les données de la NSDUH ? Comment les taux ont-ils varié entre les différents sous-groupes de jeunes (âge, genre, origine ethnique) ?
2. Les services de santé mentale semblent-ils atteindre davantage de jeunes ? Là encore, comment les taux diffèrent-ils entre les différents sous-groupes de jeunes (âge, genre, origine ethnique) ?

4.2. Evolution temporelle des EDM

Pour examiner le taux d'EDM chez les jeunes au fil du temps dans différents groupes démographiques, nous utiliserons le jeu de données `percents` que nous avons manipulé dans la section ci-dessus.

Nous utilisons la fonction `ggplot()` pour spécifier les données que nous souhaitons tracer sur chaque axe. Nous indiquerons également que nous souhaitons utiliser la variable `Demographic` de notre jeu de données pour regrouper nos données et les colorer. Il s'agit de la première couche du graphique.
Ensuite, nous

- utiliserons le signe plus `+`, pour les couches suivantes,
- utiliserons la fonction `geom_line()` du package `ggplot2` pour spécifier que nous souhaitons un graphique linéaire.
- ajouterons des étiquettes pour le titre et le sous-titre en utilisant la fonction `labs()` du package `ggplot2`.
- déplacerons notre légende en bas en utilisant la fonction `theme()` qui aide à contrôler les différents détails du graphique.

```
percents %>%
  filter(data_type == "Major_Depressive_Episode") %>%
  ggplot2::ggplot(aes(x = Year, y = Percent,
                      color = Demographic)) +
      geom_line(size = 1) +
      labs(title = "Major Depressive Episode among Persons Aged 12 to 17",
           subtitle = "By Demographic Characteristics, Percentages, 2004-2018") +
  theme(legend.position = "bottom")
```

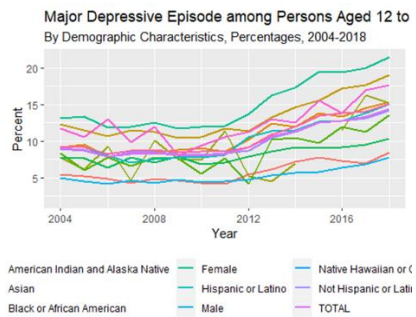


Figure 12. Evolution des EDM chez les jeunes américains

Ce graphique est très difficile à lire en raison du grand nombre de groupes.

Examinons maintenant uniquement le total dans le temps. Nous pouvons le faire en filtrant d'abord nos données pour les valeurs `TOTAL`. Il serait également intéressant d'inclure chaque année dans l'axe des x. Nous pouvons le faire en

utilisant la fonction `scale_x_continuous()` qui nous donne un plus grand contrôle sur la façon dont l'axe des x est affiché.

Enfin, nous allons supprimer la légende puisque nous n'aurons qu'un seul groupe en utilisant `legend.position = "none"` et nous pouvons changer l'angle du texte de l'axe des x en utilisant `axis.text.x = element_text(angle = 90)` dans la fonction `theme()`.

Nous allons également rendre la ligne plus épaisse en utilisant l'argument `size =` pour la fonction `geom_line()`.

La fonction `theme_classic()` modifie l'esthétique du graphique. Voir ici pour une liste d'options.

```
MDE_total <- percents %>%
  filter(data_type == "Major_Depressive_Episode",
         Demographic == "TOTAL") %>%
  ggplot(aes(x = Year, y = Percent, color = Demographic)) +
  geom_line(aes(color = Demographic), size = 1.5) +
  scale_x_continuous(breaks = seq(2004, 2018, by = 1),
                     labels = seq(2004, 2018, by = 1),
                     limits = c(2004, 2018)) +
  labs(title = "Percent of Persons Aged 12 to 17 Reporting Having a \n Major Depressive E
pisode in the Past Year ") +
  theme_classic() +
  theme(axis.text.x = element_text(angle = 90),
        legend.position = "none")

MDE_total
```

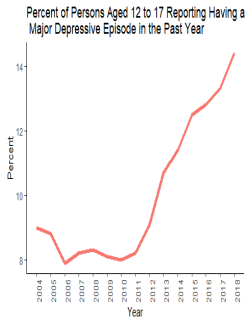


Figure 13. Evolution des jeunes (12-17 ans) déclarants un EDM

Nous pouvons voir qu'il y a une forte augmentation après 2011 environ.

Ajoutons une couleur de fond différente pour mettre en évidence les années depuis 2011. Pour ce, ajoutons un calque `geom_rect()` avant de tracer la ligne en spécifiant l'emplacement du rectangle sur notre tracé.

Ajoutons une facette en utilisant la fonction `facet_wrap()` pour ajouter une bande de texte en haut du graphe pour en dire plus sur ce qu'il contient.

Utilisons les paramètres `strip.background` et `strip.text` de la fonction `theme()` pour spécifier l'aspect du texte en haut du graphe.

Nous voulons

- changer la valeur `TOTAL` de la variable `Demographic` en `"Pourcentage de répondants avec MDE"` pour que le texte dans la bande au-dessus du graphique montre ceci à la place (fonction `recode()` du package `dplyr`).
- changer la couleur de la ligne (fonction `scale_color_manual()` du package `ggplot2`).

```
MDE_total <- percents %>%
  filter(data_type == "Major_Depressive_Episode", Demographic == "TOTAL") %>%
  mutate(Demographic = recode(Demographic,
            "TOTAL" = "Percent of respondents with MDE"))%>%
  ggplot(aes(x = Year, y = Percent, group = Demographic)) +
    facet_wrap( ~ Demographic)+
    geom_rect(xmin = 2011, xmax = Inf,  ymin = -Inf, ymax = Inf, fill = "light gray") +
    geom_line(aes(color = Demographic), size = 1.5) +
    scale_x_continuous(breaks = seq(2004, 2018, by = 1),
                       labels = seq(2004, 2018, by = 1),
                       limits = c(2004, 2018)) +
    labs(title = "The Rate of Youths Aged 12 to 17 Reporting Having a \n MDE is Increasi
ng")+
    theme_classic() +
    theme(axis.text.x = element_text(angle = 90),
          legend.position = "none",
          strip.background = element_rect(fill ="black"),
          strip.text = element_text(face = "bold", size = 14, color = "white")) +
    scale_color_manual(values = c("blue"))

MDE_total
```

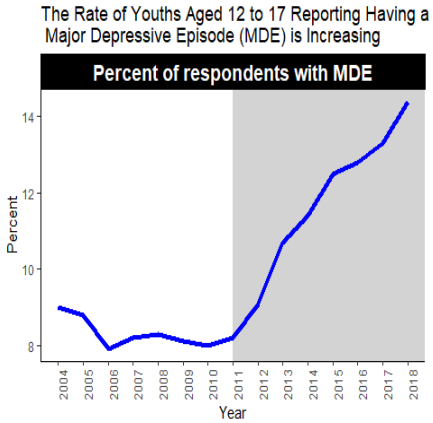


Figure 14. Evolution en pourcentages des jeunes (12-17 ans) déclarants un EDM

Sauvegardons ce plot. Utilisons

- la fonction `save()` pour le sauvegarder dans un répertoire "plots" en tant que fichier rda
- la fonction `png()` pour sauvegarder un png pour les collaborateurs
- la fonction `dev.off()` pour fermer le dispositif graphique que nous utiliserons pour créer la version png du plot

```
fs::dir_create(here("plots"))
save(MDE_total,                file                =here::here("plots",                "MDE_total.rda"))
png(here::here("plots",                                                                "MDE_total.png"))
```

```
MDE_total
dev.off()
```

Question 3 : que pensez-vous qu'il se passera si nous utilisons le symbole `+` pour ajouter la fonction `geom_rect()` avec `MDE_total` comme cela ? Est-ce que c'est ce que vous aviez prévu ? Pourquoi ou pourquoi pas ?
Votre réponse : … … …

```
MDE_total +
  geom_rect(xmin = 2011, xmax = Inf, ymin = -Inf, ymax = Inf, fill = "light gray")
```

Créons un thème pour nos futurs `ggplots` similaires comme suit :

```
ocs_theme <- function() {
  theme_classic() +
  theme(axis.text.x = element_text(angle = 90),
        strip.background = element_rect(fill = "black"),
        strip.text = element_text(face = "bold", size = 14, color = "white"))
  }
```

Vous remarquerez que nous n'avons pas utilisé `legend.position = "none"` afin que ce thème soit flexible pour les plots pour lesquels nous voulons tracer une légende.

Regardons maintenant les différences entre les groupes.

Pour s'assurer que notre graphique n'est pas trop envahissant, limitons-nous aux sous-groupes d'âge et de genre. Ainsi, nous filtrerons les données concernant les totaux et les différents groupes ethniques pour l'instant. Nous utiliserons également la fonction `facet_wrap()` pour créer des sous-graphes basés sur les catégories démographiques, que nous avons placées dans une variable appelée `subgroups` plus tôt lorsque nous avons structuré les données.

```
MDE_age_gender <-percents %>%
  filter(data_type == "Major_Depressive_Episode",
         subgroup != "Race/Ethnicity",
         Demographic != "TOTAL") %>%
  ggplot(aes(x = Year, y = Percent, color = Demographic)) +
    geom_line(aes(color = Demographic), size = 1) +
    scale_x_continuous(breaks = seq(2004, 2018, by = 1),
                       labels = seq(2004, 2018, by = 1),
                       limits = c(2004, 2018)) +
    labs(title = "Major Depressive Episode among Persons Aged 12 to 17",
         subtitle = "By Demographic Characteristics, Percentages, 2004-2018") +
    facet_wrap(~ subgroup) +
    ocs_theme()

MDE_age_gender
```

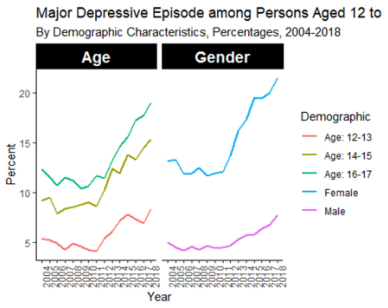


Figure 15. Evolution des EDM selon l'âge et le genre.

Bien : il est maintenant beaucoup plus facile de voir comment chaque groupe a évolué dans le temps.

Nous pouvons également ajouter des étiquettes directement aux lignes en utilisant le paquet `directlabels`. Il existe plusieurs méthodes pour ce faire. Voir ici pour plus d'informations sur les options d'ajout d'étiquettes avec ce package. Nous utilisons

- la méthode `"far.from.others.borders"` pour que nos étiquettes ne se chevauchent pas.
- la fonction `dl.trans()` du paquet `directlabels` pour déplacer les étiquettes légèrement vers le haut (`y = y +0.35`) et vers la gauche (`x = x -0.1`).
- La fonction `dl.move()` du package `directlabels` pour déplacer l'une des étiquettes à un endroit particulier.
- les arguments `cex` et le style de la police avec l'argument `fontface` pour modifier la taille des étiquettes

Note : les fonctions `dl.move` sont configurées pour le rendu du R Markdown - donc si vous visualisez l'étude de cas à partir de RStudio, les étiquettes se chevaucheront.

```
MDE_age_gender <- directlabels::direct.label(
  MDE_age_gender,
  list(dl.trans(y = y + 0.38, x = x -0.1),
       "far.from.others.borders",
       cex = .8,
       fontface=c("bold"),
       dl.move("Age: 14-15", x = 2007, y = 9.7))
  )
MDE_age_gender
```

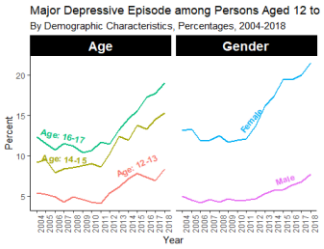


Figure 16. Evolution des EDM selon l'âge et le genre (avec labels)

Afin choisissons des couleurs judicieuses les différents groupes d'âge par ordre d'âge en fonction de l'intensité de la nuance de couleur. afin de pouvoir colorer les groupes `Mâle` et `Femme` de manière cohérente dans nos différents tracés futurs. Il serait intéressant d'intervertir les couleurs pour les hommes et les femmes pour éviter la confusion et aider à l'interprétation.

Cette fois, nous pouvons utiliser les fonctions `show_col()` et `hue_pal()` du paquet `scales` pour voir quel est le [code hexadécimal (appelé hex)] pour ces couleurs.

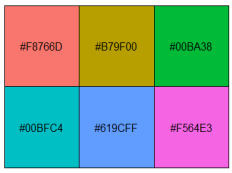


Figure 17. code hexadécimal des couleurs utilisées.

Récupérons les couleurs que nous avons précédemment utilisées

```
scales::show_col(scales::hue_pal()(6))
```

Faisons en sorte que les groupes d'âge soient de différentes nuances de vert. Nous pouvons obtenir des nuances supplémentaires en utilisant la même fonction, mais en spécifiant davantage de couleurs pour décider si nous voulons une couleur différente.

```
scales::show_col(scales::hue_pal()(30))
```

Figure 18. Autres couleurs et leur code hexadécimal

Pour les différents groupes d'âge, nous conservons quelques nuances de couleurs allant de l'or au vert. Pour les couleurs masculines et féminines, nous choisissons respectivement bleu et rose

```
age_col_light <- c("#B79F00")
age_col<- c("#6BB100")
age_col_dark<- c("#00BD5F")
Female_col <-c("#F564E3")
Male_col <- c("#619CFF")
```

Maintenant, changeons les couleurs en utilisant la fonction `scale_color_manual()` et en listant les couleurs dans l'ordre où elles apparaissent dans les données.

```
MDE_age_gender <- MDE_age_gender  +
  scale_color_manual(values = c(age_col_light,
                                age_col,
                                age_col_dark,
                                Female_col,
                                Male_col))
MDE_age_gend
```

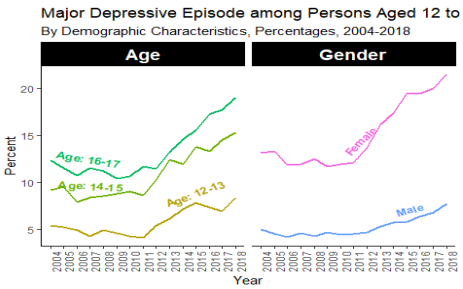


Figure 19. Evolution des EDM selon l'âge et le genre (avec labels et couleurs adéquates)

Cela semble plus clair maintenant.

Nous constatons

- que la majorité des personnes ayant déclaré avoir vécu un EDM au cours de l'année écoulée se situent dans une tranche d'âge plus élevée (16-17 ans par rapport à 12-13 ans).
- que la tendance est à la hausse pour les trois tranches d'âge depuis 2011 environ.
- une augmentation pour les deux genres depuis 2011 environ. Cette augmentation est plus marquée chez les femmes.
- que les femmes ont un pourcentage beaucoup plus élevé que les hommes pour toutes les années.

Question 4 : réaliser le même graphique avec un arrière-plan ombré différent pour les années d'augmentation, comme nous l'avons fait pour le graphique total. Nommer ce graphique `MDE_age_gender`

```
#votre code
```

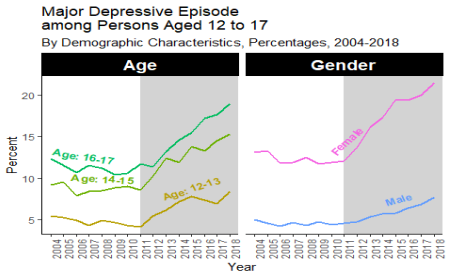


Figure 20. Evolution des EDM selon l'âge et le genre (avec labels, couleurs adéquates, et ombrages de périodes)

Sauvegardons ce plot.

```
save(MDE_age_gender, file =here::here("plots", "MDE_age_gender.rda"))
png(here::here("plots", "MDE_age_gender.png"))
MDE_age_gender
dev.off()
```

Dans la partie analyse de données, nous testerons formellement (via un test statistique) si le genre est indépendant des différences de taux d'EDM dans le temps.

Voyons ensuite comment le taux d'EDM a évolué dans le temps pour différents groupes ethniques.

Étant donné le grand nombre de groupes, nous ne voudrons probablement pas étiqueter directement les lignes cette fois-ci, mais plutôt dans la légende qui sera automatiquement créée. Nous utiliserons la fonction `fct_reorder()` du package `forcats` pour ordonner les groupes ethniques dans la légende en fonction de la dernière valeur (en utilisant `tail()`) de la variable `Percent`.
Nous allons également colorer manuellement nos lignes en nous basant sur une palette de couleurs appelée viridis (discernable pour les daltoniens) en utilisant la fonction `scale_color_viridis_d()` (destinée à colorer des valeurs discrètes) du package `ggplot2`.

```
MDE_race <- percents %>%
  filter(data_type == "Major_Depressive_Episode",
         subgroup == "Race/Ethnicity") %>%
  mutate(Demographic = forcats::fct_reorder(Demographic, Percent,
                                            tail, n = 1, .desc = TRUE)) %>%
  ggplot(aes(x = Year, y = Percent, group = Demographic)) +
    geom_rect(xmin=2011,xmax=Inf,ymin=-Inf,ymax = Inf, fill = "light gray") +
    geom_line(aes(color = Demographic), size = 1) +
    facet_wrap( ~ subgroup) +
    scale_x_continuous(breaks = seq(2004, 2018, by = 1),
                       labels = seq(2004, 2018, by = 1),
                       limits = c(2004, 2018)) +
    scale_color_viridis_d() +
    labs(title = "Major Depressive Episode\namong Persons Aged 12 to 17",
         subtitle = "By Demographic Characteristics, Percentages, 2004-2018") +
    ocs_theme()

MDE_race
```

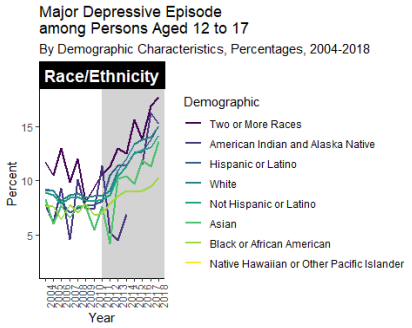


Figure 21. Evolution des EDM selon l'origine ethnique

Malheureusement, il n'y a qu'une seule valeur pour le groupe `Native Hawaiian or Other Pacific Islander`, donc comme il s'agit d'un graphique linéaire, nous n'avons pas assez de points (2 au minimum) pour créer une ligne, donc supprimons ce groupe du graphique pour supprimer le groupe de la légende.

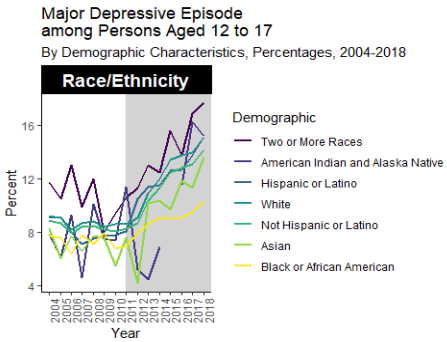


Figure 22. Evolution du pourcentage d'EDM selon l'origine ethnique (sans les NHOPI)

```
MDE_race <- percents %>%
  filter(data_type == "Major_Depressive_Episode",
         subgroup == "Race/Ethnicity",
         Demographic != "Native Hawaiian or Other Pacific Islander") %>%
```

```
  mutate(Demographic = fct_reorder(Demographic, Percent,
                                   tail, n = 1, .desc = TRUE)) %>%
  ggplot(aes(x = Year, y = Percent, group = Demographic)) +
    geom_rect(xmin = 2011, xmax = Inf,
              ymin = -Inf, ymax = Inf,
              fill = "light gray") +
    geom_line(aes(color = Demographic), size = 1) +
    facet_wrap( ~ subgroup) +
    scale_x_continuous(breaks = seq(2004, 2018, by = 1),
                       labels = seq(2004, 2018, by = 1),
                       limits = c(2004, 2018)) +
    scale_color_viridis_d() +
    labs(title = "Major Depressive Episode\namong Persons Aged 12 to 17",
         subtitle = "By Demographic Characteristics, Percentages, 2004-2018") +
    ocs_theme()

MDE_race
```

Nous constatons que

- le groupe d'individus ayant déclaré être de deux ethnies ou plus présente les pourcentages les plus élevés d'EDM au cours de l'année écoulée.
- Le groupe des personnes ayant déclaré être noires ou afro-américaines présente les pourcentages les plus faibles.
- cependant, la plupart des groupes ethniques sont assez similaires et que nous observons une augmentation pour la plupart des groupes depuis 2011 environ.

Gardez à l'esprit les limites énumérées dans la discussions Limitations lorsque vous examinez ces résultats. Il est possible que le groupe des personnes ayant déclaré être noires ou afro-américaines soit moins susceptible de déclarer des symptômes de dépression.

Sauvegardons également ce graphique :

```
save(MDE_race, file =here::here("plots", "MDE_race.rda"))
png(here::here("plots", "MDE_race.png"))
MDE_race
dev.off()
```

4.3. EDM sévère (avec déficience grave)

Voyons maintenant comment le taux global de jeunes déclarant avoir eu un EDM sévère a évolué au fil du temps. Voir la section Présentation des données pour savoir comment la déficience grave a été définie.

Question : réaliser les deux plots suivants par vous-même avant de le révéler. Cette fois-ci, nous supprimons la légende en utilisant la fonction `theme()`.

```
#votre code
```

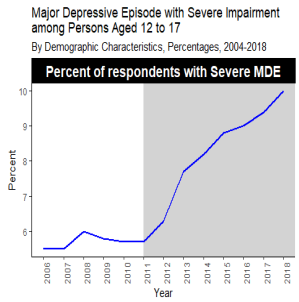


Figure 23. Evolution du pourcentage d'EDM sévère

Examinons ensuite les groupes d'âge et les différences entre les genres :

```
MDES_age_gender <-
  percents %>%
  filter(data_type == "Severe_Major_Depressive_Episode",
         subgroup != "Race/Ethnicity",
         Demographic != "TOTAL") %>%
  ggplot(aes(x = Year, y = Percent, group = Demographic)) +
    geom_rect(xmin = 2011, xmax = Inf, ymin = -Inf, ymax = Inf, fill = "light gray") +
    geom_line(aes(color = Demographic), size = 1)+
    scale_x_continuous(breaks = seq(2006, 2018, by = 1), labels = seq(2006, 2018, by = 1),
                       limits = c(2006, 2018)) +
    labs(title = "Major Depressive Episode with Severe Impairment\namong Persons Aged 12 to 17",
         subtitle = "By Demographic Characteristics, Percentages, 2004-2018") +
    facet_wrap( ~ subgroup) +
    ocs_theme()

MDES_age_gender  <- direct.label(
```

```
MDES_age_gender,
list(dl.trans(y = y +0.39, x = x -0.1),
     "far.from.others.borders",cex = .8, fontface = "bold",dl.move("Age: 14-15", x= 2016.5, y = 11))) +
scale_color_manual(values = c(age_col_light, age_col, age_col_dark, Female_col, Male_col))
```

Nous constatons

- que la majorité des personnes ayant déclaré avoir connu un EDM avec atteinte sévère se situent dans une tranche d'âge plus élevée.
- cependant, il semble y avoir un changement plus spectaculaire dans le groupe d'âge moyen entre 2011 et 2012.
- une très forte augmentation des données pour les femmes après 2011, encore une fois beaucoup plus forte que l'augmentation observée pour les hommes au fil du temps.

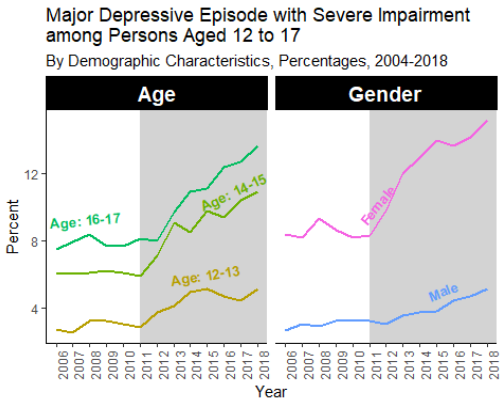


Figure 24. Evolution du pourcentage d'EDM sévère par âge et genre

Examinons maintenant les différents groupes ethniques.

Question : réaliser la figure ci-contre. Nommer la `Race_MDES`.

```
#votre code
```

Les tendances sont les mêmes que pour le graphique précédent sur les groupes ethniques. Le taux est le plus élevé pour les personnes appartenant à deux races ou plus et le plus bas pour les Noirs ou les Afro-Américains. Les données concernant le groupe des "Indiens d'Amérique et des autochtones de l'Alaska" sont rares, et il n'est donc pas certain que leur taux soit le plus bas la dernière année.

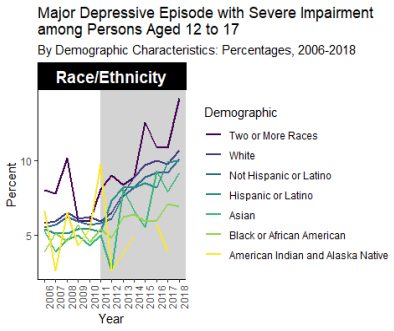


Figure 25. Evolution du pourcentage d'EDM sévère

4.4. Traitement des EDM

Examinons maintenant les personnes qui ont déclaré avoir eu un EDM et qui ont reçu un traitement pour la dépression. Tout d'abord, examinons l'ensemble en utilisant le groupe `Démographique == "TOTAL"`. Nous allons supprimer la légende pour ce graphique.

Question : réaliser la figure ci-contre. Nommer le plot `Treat_total`.

```
#votre code
```

Cela montre qu'

- environ 40 % des jeunes qui ont déclaré un EDM ont également reçu un traitement. Ainsi, la majorité des jeunes ayant déclaré un EDM ne reçoivent pas de traitement.
- une augmentation globale du taux de jeunes recevant un traitement depuis 2011, à l'instar de la tendance observée pour le nombre d'EDM, mais les données relatives au traitement sont plus variables d'une année à l'autre.

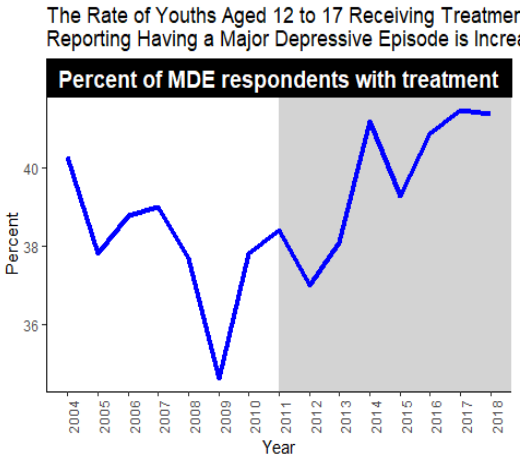


Figure 26. Evolution du pourcentage de traitement d'EDM

Ensuite, nous examinons les différences entre les hommes et les femmes et les différents groupes d'âge. Commençons par enregistrer ce graphique :

```
save(Treat_total, file =here::here("plots", "Treat_total.rda"))
png(here::here("plots", "Treat_total.png"))
Treat_total
dev.off()
```

Question : réaliser la figure ci-contre. Nommer le plot `treat`.

```
#votre code
```

Il semble y avoir une tendance à la hausse, mais elle est loin d'être aussi importante que celle que nous avons observée pour l'augmentation des EDM. D'une manière générale, les données semblent également varier beaucoup plus.

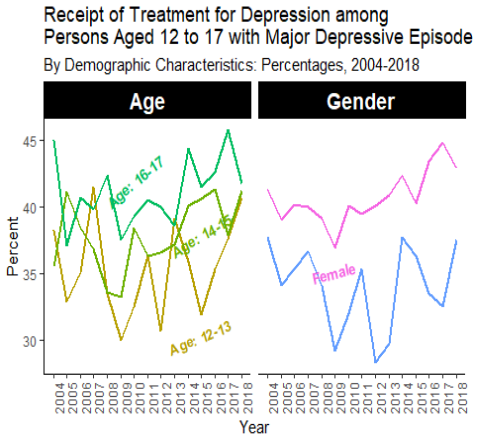


Figure 27. Evolution du pourcentage de traitement pour un EDM sévère par âge et genre

Question : Créez un graphique similaire pour les différents groupes ethniques. Nommer le plot `Race_treat`.

```
#votre code
```

Il semble que les jeunes qui se déclarent blancs soient ceux qui reçoivent le plus de soins de la part des services de santé mentale.

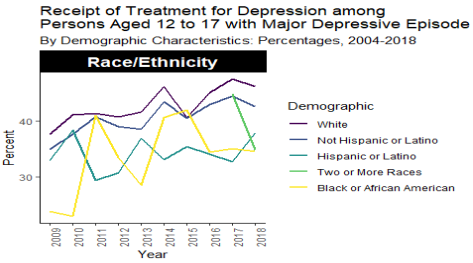


Figure 28. Evolution du pourcentage de traitement pour un EDM sévère par origine ethnique

4.5. Traitement des EDM

Nous allons également examiner où les jeunes reçoivent un traitement en utilisant les valeurs du `tableau11.1b` qui contient les valeurs en pourcentage pour les chiffres présentés dans le `tableau11.1a`.

Nous pouvons utiliser la fonction `str_detect()` du package `stringr` pour filtrer les valeurs de la variable `short_label` qui contient le mot `total`.

```
plotMHS <- table11.1b %>%
  filter(stringr::str_detect(short_label, "total") ) %>%
  ggplot(aes(x = Year, y = Percent, group = MHS_setting, color = short_label)) +
    geom_line(size = 1) +
    facet_wrap( ~ type) +
    scale_x_continuous(breaks = seq(2009, 2018, by = 1),
                                labels = seq(2009, 2018, by = 1),
                                limits = c(2009, 2018)) +
    labs(title = "Settings Where Mental Health Services Were Received\namong Per
sons Aged 12 to 17",
         subtitle = "Percentages, 2002-2018") +
    ocs_theme()

plotMHS <- direct.label(
  plotMHS,
  list(dl.trans(y = y +0.35, x = x -0.1),
       "far.from.others.borders",
       cex = .8, dl.move("Outpatient total", x = 2015, y =11))
  )

plotMHS
```

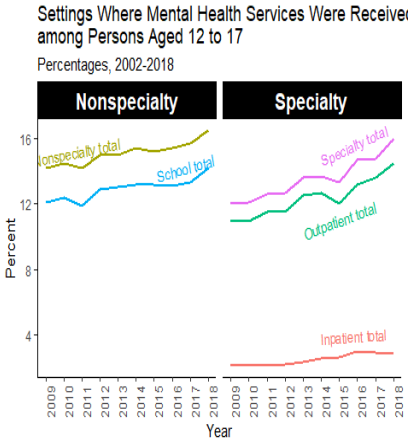


Figure 29. Evolutions des environnements de traitements des EDM (sans les sous-catégories de services de santé mentale labels)

Nous constatons que les jeunes semblent recevoir des soins dans des établissements non spécialisés à un taux légèrement supérieur à celui des établissements spécialisés.

Un établissement

- non spécialisé fournit des traitements de santé généraux et d'autres services, comme un hôpital classique ou une école.

- spécialisé se concentre sur le traitement de la santé mentale. Les services ambulatoires sont ceux dans lesquels le patient ne passe même pas une nuit à l'hôpital ou dans l'établissement de traitement, tandis que les services hospitaliers sont ceux dans lesquels le patient passe au moins une nuit dans l'établissement de soins.

Toutefois, les taux semblent être très similaires et les différences relatives semblent être constantes dans le temps.

Examinons maintenant les sous-catégories de services de santé mentale. Pour ce faire, nous allons filtrer les valeurs de la variable `short_label` qui ne contiennent pas le mot "total" en utilisant un `!` devant l'instruction `str_dectect`.

```
plotMHSS <- table11.1b %>%
  filter(!str_detect(short_label, "total")) %>%
  ggplot(aes(x = Year, y = Percent,
         group = MHS_setting, color = short_label)) +
    geom_line(size = 1) +
    facet_wrap( ~ type) +
    scale_x_continuous(breaks = seq(2002, 2019, by = 1),
                       labels = seq(2002, 2019, by = 1),
                       limits = c(2002, 2019)) +
    labs(title = "Settings Where Mental Health Services Were Received\namong Pe
rsons Aged 12 to 17",
         subtitle = "Percentages, 2002-2018") +
    ocs_theme()

plotMHSS <- direct.label(
  plotMHSS,
  list(dl.trans(y = y +0.3),
       "far.from.others.borders",
       dl.move("School Therapist", 2010, 10),
       dl.move("Fostercare", 2010, 1),
       dl.move("Therapist", x=2009, y = 10.5))
  )

plotMHSS
```

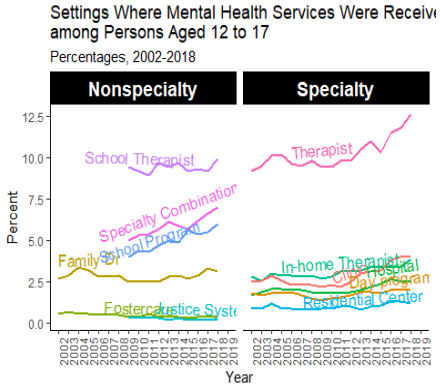


Figure 30. Evolutions des environnements de traitements des EDM (avec les sous-catégories de services de santé mentale)

Il semble que la plupart des jeunes reçoivent des soins d'un thérapeute ou d'un thérapeute scolaire.

Nous savons maintenant

- comment se comparent les taux des différents sous-groupes en ce qui concerne le fait d'avoir eu un EDM au cours de l'année écoulée, d'avoir eu un EDM avec une déficience grave et d'avoir reçu un traitement après un EDM.
- également où les jeunes sont généralement traités.

Mais comment se comparent les taux d'EDM au cours de l'année écoulée, d'EDM avec déficience grave et de traitement au sein de chaque sous-groupe (par exemple, uniquement les femmes) ?
C'est ce que nous vérifions dans la section suivante.

4.6. Résultats globaux par groupe

Dans le graphique suivant, nous filtrons d'abord les variables `Male`, `Femme` et `Total`, puis nous les regrouperons par la variable `Démographique`. Nous utiliserons * différents types de lignes pour indiquer les différentes valeurs des résultats en utilisant la fonction `scale_linetype_manual()`. * le paquet `ggthemes` et le paquet `scales` afin de voir toutes les options actuelles pour les différents types de lignes.

```
ggthemes::show_linetypes(scales::linetype_pal()(12), labels = TRUE)
```

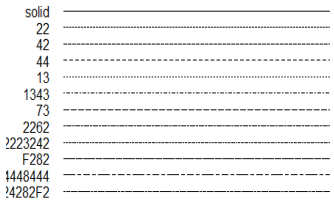


Figure 31. 12 types différents de lignes

- Nous pouvons maintenant utiliser
- les étiquettes pour les différents types de lignes dans la fonction `scale_linetype_manual()` pour spécifier des types de lignes spécifiques.
- la fonction `guides()` du package `ggplot2` pour supprimer la légende spécifiquement pour la couleur, et non pour le type de ligne en utilisant `guides(color = FALSE).

```
gender_outcomes <-
  percents %>%
  filter(Demographic %in% c("Male", "Female", "TOTAL")) %>%
  ggplot(aes(x = Year, y = Percent, color = Demographic)) +
    geom_line(aes(linetype = data_type), size = 1) +
```

```
    scale_linetype_manual(values = c("solid", "2262", "13")) +
    scale_color_manual(values = c(Female_col, Male_col, "black")) +
    scale_x_continuous(breaks = seq(2004, 2018, by = 1),
                       labels = seq(2004, 2018, by = 1),
                       limits = c(2004, 2018)) +
    labs(title = "Major Depressive Episodes and Treatment Among Persons Aged 12 to 17",
         subtitle = "By Demographic Characteristics, Percentages, 2004-2018") +
    facet_wrap( ~ Demographic, strip.position = "top") +
  ocs_theme() +
  theme(legend.title = element_blank(),
        legend.position = "bottom") +
    guides(color = FALSE)

gender_outcomes
```

On constate

- qu'une grande partie des personnes qui connaissent un EMD ont un épisode avec une déficience sévère pour chaque groupe.
- que les femmes ont un taux plus élevé de ces deux types d'EMD et de traitement. Bien que les femmes aient plus du double du taux d'EMD, elles reçoivent un taux de traitement de la dépression relativement similaire à celui des hommes.

Cela suggère que les femmes sont plus susceptibles que les hommes de déclarer des symptômes de dépression dans les enquêtes, ou qu'elles ne reçoivent pas autant de soins malgré un besoin plus important.

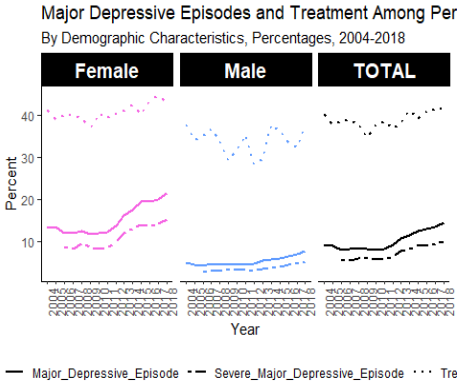


Figure 32. Evolution des EDM : homme, femmes et ensemble

Question : réaliser un graphique similaire pour différents groupes d'âge.

```
#votre code
```

Tous les groupes d'âge présentent un ratio similaire d'EDM sévères pour ceux qui ont connu un épisode.

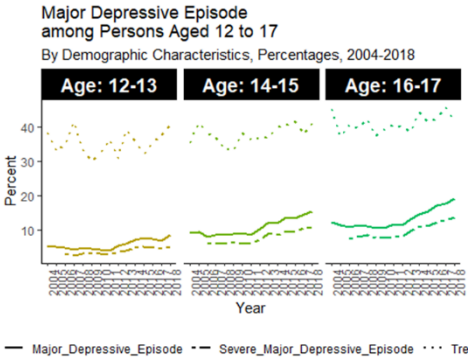


Figure 33. Evolution des EDM par tranche d'âge

Question : réaliser un graphique similaire pour différents groupes ethniques

```
#votre code
```

Tous les groupes ethniques présentent également un taux similaire d'épisodes graves par rapport au taux général d'épisodes. Le taux de traitement est assez similaire par rapport au pourcentage de jeunes ayant déclaré avoir des symptômes dans chaque groupe.

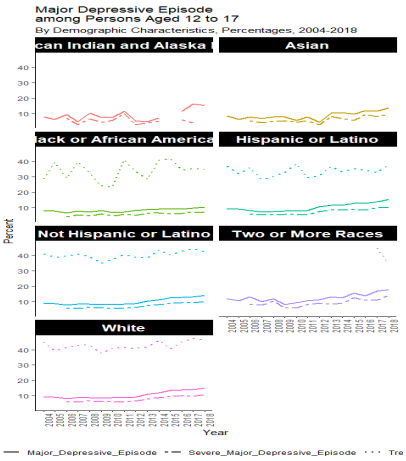


Figure 34. Evolution des EMD selon les groupes ethniques

5. RESULTATS : ANALYSE DE DONNEES

5.1. (Re-)chargement des données

Si vous reprenez le TP, vous pouvez charger vos données structurées d'un des trois manières suivantes :

```
load(file = here::here("data", "wrangled", "wrangled_data.rda"))
```

Méthode 2 :

```
library(OCSdata)
wrangled_rda("ocs-bp-youth-mental-health", outpath = getwd())
load(here::here("OCSdata", "data", "wrangled", "wrangled_data.rda"))
```

Télécharger un fichier RDA des données structurées ici ou ici. Placez-ce fichier dans votre répertoire de travail actuel dans un sous-répertoire appelé "data>wrangled".

```
load(here::here("data", "wrangled", "wrangled_data.rda"))
```

Rappelons les principales questions :

- Comment les taux de dépression chez les jeunes Américains ont-ils évolué depuis 2004, selon les données de la NSDUH ? Comment les taux ont-ils varié entre les différents sous-groupes de jeunes (âge, genre, origine ethnique) ?
- Les services de santé mentale semblent-ils atteindre davantage de jeunes ? Là encore, comment les taux diffèrent-ils entre les différents sous-groupes de jeunes (âge, genre, origine ethnique) ?

Nous nous intéressons

- à la manière dont la fréquence des EDM de 2018 chez les jeunes diffère de celle observée en 2004.
- aux différences entre les différents sous-groupes.

Pour cela, nous utiliserons le test du χ^2 de Pearson d'indépendance (détails sur ce test dans la section méthode).

5.2. Lien entre genre et EMD

Nous commencerons par examiner les différences entre les hommes et les femmes au fil du temps. Cela nous permettra de comparer si les fréquences relatives des EDM diffèrent de ce que nous attendrions par hasard si les variables années et genre étaient indépendantes. Puisque nous disposons de chiffres pour les deux genres : hommes et femmes, et pour les deux années qui nous intéressent, 2004 et 2018, nous pouvons effectuer un test du χ^2.

Question : *le taux d'EDM déclarés au cours des deux années (2004 et 2018) est-il associé au genre ?*

L'hypothèse nulle associées est

H0 = "la proportion d'EMD déclarés par les hommes ou les femmes est la même pour chaque année"
="genre et évolution de l'EMD sont indépendants "

Créons maintenant un tableau de contingence avec nos propres données.
Il est essentiel que nous utilisions les données de comptage, et non les données de pourcentage pour notre analyse, car le test χ^2 **exige** des comptages. Nous allons filtrer les données de comptage pour les données `Major_Depressive_Episode`, ainsi que pour les données `Male` et `Female` de `2004` et `2018`.
Le code suivant filtre les données dont nous avons besoin et effectue les manipulations nécessaires pour que les unités d'observation soient appropriées.

```
chi_squared_11.2a <- counts %>%
  filter(data_type == "Major_Depressive_Episode") %>%
  filter(Year %in% c(2004, 2018)) %>%
  filter(Demographic %in% c("Male", "Female")) %>%
  mutate(Number = Number * 1000) # because the numbers are in thousands
```

L'objet résultant contient toutes les valeurs dont nous avons besoin pour notre tableau de contingence.

```
chi_squared_11.2a

# A tibble: 4 × 5
  Demographic subgroup data_type                Year  Number
  <chr>       <chr>    <chr>                   <dbl>   <dbl>
1 Male        Gender   Major_Depressive_Episode  2004  637000
2 Male        Gender   Major_Depressive_Episode  2018  946000
3 Female      Gender   Major_Depressive_Episode  2004 1588000
4 Female      Gender   Major_Depressive_Episode  2018 2537000
```

Un tableau de contingence peut maintenant être produit à partir de ces données (qui sont actuellement en format long) en transformant les données en format long et en réutilisant certaines valeurs comme noms de lignes. Pour reformater les données en format long, nous pouvons utiliser la fonction `pivot_wider()` du package `tidyr`dont les principaux arguments sont :

- `names_from` - c'est la variable d'où proviendront les noms des nouvelles colonnes
- `values_from` - c'est la variable d'où proviendront les valeurs des nouvelles colonnes.
- `names_prefix` - si nous voulons ajouter un préfixe aux nouvelles colonnes, nous pouvons le faire en utilisant cet argument

Dans notre cas, nous voulons

- répartir les données relatives à l'année dans deux colonnes. Les noms proviendront donc de la variable `Year` et les valeurs proviendront de la variable `Number`.
- ajouter le mot `Year` aux nouvelles colonnes.
- supprimer les variables `subgroup` et `data_type` et ne garder que les variables `Demographic`, `Year`, et `Number`. Pour ce faire, nous pouvons utiliser la fonction `select()`.

Utilisons les fonction `pivot_wider()` et `select()`, pour créer le tableau de contingence

```
chi_squared_11.2a %<>%
  select(Demographic, Year, Number) %>%
  tidyr::pivot_wider(names_from = Year,
                     names_prefix = "Year",
                     values_from = Number)
chi_squared_11.2a
```

A ce stade, nous avons trois colonnes, mais la première colonne n'a besoin que des étiquettes `Male` et `Female` et nous voulons la traiter comme des noms de lignes. Pour convertir une colonne en noms de lignes, utilisons la fonction `column_to_rownames()` du package `tibble` pour faire de la variable `Demographic` le niveau des noms de lignes.

```
chi_squared_11.2a %<>%
  tibble::column_to_rownames("Demographic")
chi_squared_11.2a
```

```
       Year2004 Year2018
Male     637000   946000
Female  1588000  2537000
```

Note : un tableau de contingence devrait normalement avoir aussi des totaux pour tous les groupes, mais ce n'est pas nécessaire pour la fonction `stats::chisq.test()`.

Le test du chi-2 pour l'indépendance peut être effectué en utilisant la fonction `stats::chisq.test()`.

```
stats::chisq.test(chi_squared_11.2a)
```

```

    Pearson's Chi-squared test with Yates' continuity correction

data:  chi_squared_11.2a
X-squared = 1461.2, df = 1, p-value < 2.2e-16
```

La *p*-value est très petite, on rejette H0, ce qui suggère qu'il y a une dépendance entre le genre et le nombre d'EDM à travers le temps (2004 comparé à 2018).

Maintenant que nous voyons qu'il y a probablement une dépendance, nous voulons décrire la **taille** de l'association entre les variables. Pour cela, utilisons la fonction prop_test() du package `rstatix`.
Dans le cas présent, H0 est que l'année et le genre sont indépendants. Si l'année et le genre sont indépendants, nous nous attendons à ce que les hommes aient la même proportion d'épisodes pour chacune des deux années, ce qui est exactement l'hypothèse nulle d'un test comparant deux proportions. Il est utile de considérer notre test comme une comparaison de deux proportions, car cela peut nous donner un intervalle de confiance (pour en savoir plus sur les différences entre les proportions. Voir ici pour plus d'informations.

```
prop_test(chi_squared_11.2a, detailed = TRUE, correct = TRUE) %>%
glimpse()
```

```
Rows: 1
Columns: 13
$ n           <dbl> 5708000
$ n1          <dbl> 1583000
$ n2          <dbl> 4125000
$ estimate1   <dbl> 0.4024005
$ estimate2   <dbl> 0.3849697
$ statistic   <dbl> 1461.23
$ p           <dbl> 1.040008e-319
$ df          <dbl> 1
$ conf.low    <dbl> 0.01653368
$ conf.high   <dbl> 0.01832793
$ method      <chr> "Prop test"
$ alternative <chr> "two.sided"
$ p.signif    <chr> "****"
```

Ici :

- n est le total pour les hommes et les femmes des deux années.
- $n1$ est le total n pour les hommes des deux années.
- $n2$ est le n total pour les femmes des deux années.
- *statistic* est équivalente à la statistique du χ^2
- estimate1 est la proportion des rapports masculins donnés en 2004 (sur le nombre total d'hommes rapportant un épisode en 2004 et 2018)
- estimate2 est l'équivalent pour les femmes.

Ainsi estimate1 pour les hommes est : $\frac{\text{Males2004}}{\text{MalesBothYears}} = \frac{637000}{637000+946000} = \frac{637000}{n_1} = 0.40$

Ainsi estimate2 pour les femmes est : $\frac{\text{Females2004}}{\text{FemalesBothYears}} = \frac{1588000}{2537000+1588000} = \frac{1588000}{n_2} = 0.38$

Ainsi, sur l'ensemble des déclarations de EMD présentés par les hommes au cours de ces deux années, 40 % l'ont été en 2004. Pour les femmes, 38% des déclarations de EMD ont été faites en 2004.

L'intervalle de confiance donne une gamme de valeurs plausibles pour la différence réelle de ces proportions dans la population. Il nous donne une idée de la différence entre les hommes et les femmes en ce qui concerne la proportion de signalements effectués en 2004. D'après notre intervalle de confiance, nous sommes sûrs à 95 % que la véritable différence dans la proportion de rapports reçus en 2004 entre les hommes et les femmes (sur le total de chacun) se situe entre 1,65 % et 1,83 %. Il ne s'agit pas d'un changement très important. Mais nous Cependant 0 n'est pas dans cet intervalle de confiance. Ainsi, nous sommes certains que la différence n'est pas égale à 0, ce qui suggère qu'il existe effectivement une différence entre les proportions (ce qui revient à vérifier si la valeur p est inférieure à 0,05). Pour plus d'informations sur la relation entre les intervalles de confiance et les valeurs p, voir ce document.

Vous remarquerez peut-être que les proportions estimées par `prop_test()` ne correspondent pas tout à fait à l'hypothèse nulle énoncée plus tôt, selon laquelle la proportion d'EMD déclarés par les hommes est la même pour chaque année. Au lieu de cela, nous avons comparé la proportion d'épisodes signalés en 2004 entre les hommes et les femmes. Nous pouvons obtenir des proportions qui correspondent à l'hypothèse nulle énoncée précédemment en transposant le tableau de contingence que nous utilisons de manière à ce que les colonnes soient composées d'hommes et de femmes et que les années figurent sur les lignes. Dans ce cas, le résultat de notre test sera le même, puisqu'il teste l'indépendance de l'année et du genre, mais les proportions estimées seront la proportion d'hommes (sur les hommes + les femmes) en 2004 et la proportion d'hommes (sur les hommes + les femmes) en 2018.

Nous pouvons utiliser la fonction de base `t()` pour transposer notre tableau de contingence.

```
t(chi_squared_11.2a)
t(chi_squared_11.2a) %>%
  prop_test( detailed = TRUE, correct = TRUE) %>%
  glimpse()
```

```
          Male  Female
Year2004 637000 1588000
Year2018 946000 2537000
```

```
Rows: 1
Columns: 13
$ n           <dbl> 5708000
$ n1          <dbl> 2225000
$ n2          <dbl> 3483000
$ estimate1   <dbl> 0.2862921
$ estimate2   <dbl> 0.2716049
$ statistic   <dbl> 1461.23
$ p           <dbl> 1.040008e-319
$ df          <dbl> 1
$ conf.low    <dbl> 0.0139312
$ conf.high   <dbl> 0.01544319
$ method      <chr> "Prop test"
$ alternative <chr> "two.sided"
$ p.signif    <chr> "****"
```

Ici

- n est à nouveau le total pour les hommes et les femmes des deux années.
- $n1$ est le total n pour les hommes et les femmes en 2004.
- $n2$ est le n total pour les hommes et les femmes en 2018.
- *statistic* est équivalente à la statistique du χ^2
- estimate1 est la proportion des rapports masculins donnés en 2004 (sur le nombre total d'hommes et de femmes ayant rapporté un épisode en 2004)
- estimate2 est l'équivalent pour 2018.

Ainsi estimate1 est : $\frac{\text{Males2004}}{\text{MalesAndFemales2004}} = \frac{637000}{637000+1588000} = \frac{637000}{n_1} = 0.29$

Ainsi estimate2 est : $\frac{\text{Males2018}}{\text{MalesAndFemales2018}} = \frac{946000}{2537000+946000} = \frac{637000}{n_2} = 0.27$

Nous pouvons maintenant interpréter notre intervalle de confiance de la manière suivante : nous sommes sûrs à 95 % que la différence dans la proportion d'hommes déclarant un EDM au cours des deux années est comprise entre 1,39 % et 1,54 %. Sur l'ensemble des signalements, il y a environ 1,5 % de plus d'hommes en 2004 qu'en 2018.

Nous pouvons également jeter un coup d'œil à notre graphique pour faciliter l'interprétation. Cette fois-ci, nous montrerons le même graphique que précédemment, mais pour les nombres au lieu des pourcentages.

```
MDE_age_gender_counts <-
  counts %>%
  filter(data_type == "Major_Depressive_Episode",
         subgroup != "Race/Ethnicity",
         Demographic != "TOTAL") %>%
  ggplot(aes(x = Year, y = Number, group = Demographic)) +
    geom_rect(xmin = 2011, xmax = Inf, ymin = -Inf, ymax = Inf,
              fill = "light gray") +
    geom_line(aes(color = Demographic), size =1) +
    scale_x_continuous(breaks = seq(2004, 2018, by=1),
                       labels = seq(2004, 2018, by=1),
                       limits = c(2004, 2018)) +
    labs(title = "Major Depressive Episode\namong Persons Aged 12 to 17",
         subtitle = "By Demographic Characteristics, Percentages, 2004-2018") +
    facet_wrap(~ subgroup) +
    ocs_theme()

MDE_age_gender_counts <- direct.label(
  MDE_age_gender_counts,
  list(dl.trans(y = y +0.38, x = x -0.2), "far.from.others.borders",
       cex = .8, fontface = "bold",
       dl.move("Age: 14-15", x = 2008, y =10))) +
  scale_color_manual(values = c(age_col_light, age_col, age_col_dark,
                                Female_col, Male_col))
MDE_age_gender_counts
```

Nous pouvons voir que la ligne bleue relative à la somme des lignes roses et bleues en 2004 (environ 29%) est assez similaire à celle de 2018 (environ 27%). Il peut être difficile de voir des proportions et surtout une différence proportionnelle de ~ 1,5 % !

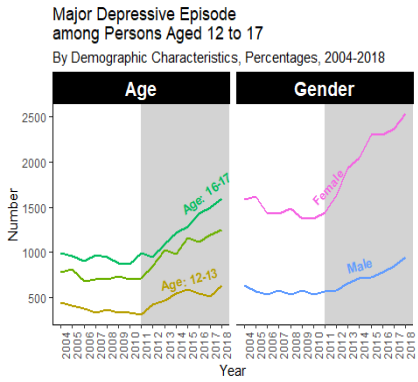


Figure 35. EDM par genre et âge.

5.3. Lien entre genre et de l'EDM sévères

Question : *le taux d'EDM sévères déclarés au cours des deux années (2004 et 2018) est-il associé au genre ?*

L'hypothèse nulle associées est

H0 = "la proportion d'EMD sévères déclarés par les hommes ou les femmes est la même pour chaque année"
="genre et évolution de l'EMD sévères sont indépendants "

```
chi_squared_11.3a <- counts %>%
  filter(data_type == "Severe_Major_Depressive_Episode",
         Year %in% c(2006, 2018),
         Demographic %in% c("Male","Female")) %>%
  mutate(Number = Number * 1000) %>%
  select(-data_type, -subgroup) %>%
  pivot_wider(names_from = Year,
              names_prefix = "Year",
              values_from = Number) %>%
  column_to_rownames("Demographic")

chi_squared_11.3a

       Year2006 Year2018
Male     335000   628000
Female  1023000  1795000

chisq.test(chi_squared_11.3a)


    Pearson's Chi-squared test with Yates' continuity correction

data:  chi_squared_11.3a
X-squared = 715.87, df = 1, p-value < 2.2e-16
```

Il semble que le genre ait une influence sur le taux d'EDM graves au fil des ans.

```
t(chi_squared_11.3a)

           Male  Female
Year2006 335000 1023000
Year2018 628000 1795000

t(chi_squared_11.3a) %>%
  prop_test( detailed = TRUE, correct = TRUE) %>%
  glimpse()

Rows: 1
Columns: 13
$ n           <dbl> 3781000
$ n1          <dbl> 1358000
$ n2          <dbl> 2423000
$ estimate1   <dbl> 0.2466863
$ estimate2   <dbl> 0.2591828
$ statistic   <dbl> 715.8654
$ p           <dbl> 1.06e-157
$ df          <dbl> 1
$ conf.low    <dbl> -0.01340819
$ conf.high   <dbl> -0.01158486
```

```
$ method      <chr> "Prop test"
$ alternative <chr> "two.sided"
$ p.signif    <chr> "****"
```

La différence entre les proportions des deux années est probablement comprise entre 0,013 et 0,012 ou -1,3 % et -1,2 %. Cette fois-ci, la proportion de Males déclarés par rapport au nombre total de déclarations chaque année était plus importante en 2018 (estimate2 = 26 %) qu'en 2006 (estimate1 = 24,7 %). Là encore, la différence est assez faible et l'intervalle n'inclut pas 0, ce qui suggère qu'il existe effectivement une association entre le genre et le nombre d'EDM graves au fil du temps (2006 par rapport à 2018).

5.4. Lien entre Genre et Traitement

Question : *L'évolution de l'utilisation d'un traitement lié à l'EMD (2004 et 2018) est-il associé au genre ?*

Question : Apres avoir determiné H0 destiné à répondre à la question ci-dessus, effectuer un test du Chi-2 pour valider ou invalider H0. Si dependance, étudier la taille de cette dependance.
L'hypothèse nulle associées est
H0 = "……..”
="……..”

```
#votre code
```

Le genre semble également avoir une influence sur le taux de traitement des jeunes au cours des deux années. Maintenant que nous voyons qu'il y a probablement une dépendance, décrivons la taille de l'indépendance entre les variables.

```
#votre code
```

Les valeurs de notre intervalle de confiance suggèrent qu'il y a une petite différence (environ 2 % de différence dans la proportion d'hommes recevant un traitement pour l'EMD au cours des deux années) et la fourchette ne dépasse pas 0, ce qui suggère qu'il y a effectivement une différence dans les proportions.
Dans ce cas, nous constatons

- en 2004, que les hommes ne représentaient ……% de tous les jeunes déclarant avoir reçu un traitement
- en 2018, que les hommes ne représentaient que ……..% des jeunes déclarant avoir reçu un traitement.

6. DISCUSSIONS

6.1. Que s'est-il passé à partir de 2011 ?

Bien qu'il soit très intrigant de constater une augmentation autour de 2011, nous n'entrerons pas dans les détails ici pour en expliquer les raisons. Toutefois, nous revoyons vers quelques articles qui ont étudié l'augmentation des taux de dépression.
article 1 : les populations modernes sont de plus en plus suralimentées, mal nourries, sédentaires, privées de soleil, de sommeil et socialement isolées. Ces changements de mode de vie contribuent à une mauvaise santé physique, et influent sur l'incidence et le traitement de la dépression.
article 2 : l'utilisation des smartphones et des médias sociaux peut avoir conduit à une augmentation des taux de dépression.

6.2. Limitations

Il convient de garder à l'esprit certaines considérations importantes concernant l'analyse des données :
Les données que nous utiliserons proviennent d'une enquête et sont donc des valeurs provenant d'un échantillon qui estime celui de la population réelle. Dans notre analyse statistique, nous utilisons ces valeurs d'échantillon comme s'il s'agissait d'estimations de la population (parce que c'est tout ce à quoi nous avons accès). Par conséquent, nos résultats ne sont pas nécessairement représentatifs des différences au sein de la population. En outre, le mécanisme d'échantillonnage utilisé pour l'enquête peut introduire un biais de sélection dans les cas où les méthodes d'échantillonnage ne produisent pas un échantillon représentatif.

Les données sont collectées auprès de participants humains, ce qui présente un *potentiel* de biais d'information, car il est *potentiel* que les participants de la base de sondage puissent, pour diverses raisons, fournir des informations inexactes.

Bien que le genre et le genre ne soient pas réellement binaires, les données utilisées dans cette analyse ne contiennent malheureusement que des informations sur les groupes d'individus qui se sont déclarés comme étant de genre masculin ou féminin.

7. CONCLUSION

7.1. Graphique de conclusion

Réalisons un graphique qui résume nos principaux résultats. Celui sera constitué de 4 graphiques que nous jugeons importants.

La fonction `ggdraw()` du package `cowplot` permet d'ajouter des labels et d'autres éléments de tracé sur des plots existants. Ainsi, pour ajouter un titre à notre graphique global que nous ajouterons à un graphique combiné de nos graphiques existants, nous pouvons utiliser `ggdraw()` pour commencer et ensuite la fonction `draw_label()` pour ajouter du texte.

```
title_plots <-
  ggdraw() +
  draw_label(
    "Self-Reported Depression Among American Youths",
    fontface = 'bold',
    size = 18,
    x = 0,
    hjust = -0.01
  )
```

L'argument `x = 0` place le titre à l'extrême gauche de la zone de tracé. Ainsi, si nous utilisons une valeur de -0.01, le titre sera légèrement éloigné du bord gauche de la zone de tracé.

Question : Que se passe-t-il si nous modifions la valeur `hjust` ?
L'argument `hjust` déplace l'étiquette dans le sens ……

Nous pouvons également créer un sous-titre de la même manière. Ici, nous créons un sous-titre appelé `forward`, que nous utiliserons plus tard.

```
forward <- ggdraw() +
  draw_label(
    "The percentage of youths (age 12-17) experiencing major depressive episodes (MDE) has\nincreased sin
ce 2011. Of these youths, the percentage receiving treatment for depression has also\n increased but rema
ins limited to less than 42%.",
    size = 16,
    x = 0,
    hjust = -0.01
  )
```

Ensuite, nous allons modifier certains de nos graphiques existants en utilisant la fonction `theme()` comme nous l'avons fait précédemment pour supprimer le titre de l'axe x, pour changer la couleur du texte de l'axe et la taille et la couleur du titre, ainsi que pour changer les titres des graphiques.

Commençons par charger les graphiques que nous avons l'intention d'utiliser :

```
load(file = here::here("plots", "MDE_total.rda"))
load(file = here::here("plots", "Treat_total.rda"))
load(file = here::here("plots", "MDE_age_gender.rda"))
load(file = here::here("plots", "MDE_race.rda"))
```

Na gardons que les graphiques eux-mêmes (c'est-à-dire enlevons titres et sous-titre)

```
MDE_total_for_mp <- MDE_total +
  theme(plot.title = element_blank(),
        plot.subtitle = element_blank(),
        axis.text = element_text(color = "black"))

treat_for_mp <-
  Treat_total +
  theme(plot.title = element_blank(),
```

```
        plot.subtitle = element_blank(),
        axis.text = element_text(color = "black"))

MDE_age_gender_for_mp <-
  MDE_age_gender +
  theme(plot.title= element_text(size = 14, color = "black"),
        plot.subtitle = element_blank(),
        axis.text = element_text(color = "black")) +
  labs(title = "Older youths and females report MDE at the highest rates\nand show the steepest increase"
)
```

Pour le dernier plot, nous voulons également récupérer la légende et la sauvegarder en tant qu'objet séparé afin de pouvoir l'ajouter à notre grille de tracé d'une manière qui ne réduise pas la taille de notre tracé pour l'adapter à la légende (fonction `get_legend()` du package `cowplot`). Nous pouvons aussi spécifier comment la légende doit être justifiée en utilisant `theme(legend.justification =)` qui prend un certain nombre d'options dont `"center"`, `"left"`, et `"right"`.
Cependant, au préalable, nous voulons aussi changer la façon dont la légende est affichée. Utilisons la fonction `guides()` du package `ggplot2` pour modifier la légende et spécifier que la légende soit affichée sur 2 colonnes comme `guides(color = guide_legend(ncol = 2))`. Nous devons spécifier que nous modifions la légende pour la couleur.

```
MDE_race_for_mp_leg <- MDE_race +
  theme(plot.title= element_text(size = 14, color = "black"),
        plot.subtitle = element_blank(),
        axis.text = element_text(color = "black"),
        legend.position = "right",
        legend.title = element_blank(),
        legend.text = element_text(size = 14)) +
  labs(title = "All racial/ethnic groups show similar\nincreases since 2011") +
  guides(color = guide_legend(ncol = 2))

legend <- get_legend(MDE_race_for_mp_leg +
          theme(legend.justification = "right"))
```

Question : supprimer la légende de ce graphique. Nommer ce graphique `MDE_race_for_mp.`

```
#votre code
```

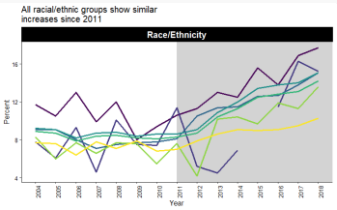


Figure 36. Evolution du pourcentage des EMD par origine ethnique (sans légende)

Nous sommes maintenant prêts à assembler nos graphiques en utilisant la fonction `plot_grid()` du package `cowplot`. Il est utile :

- de commencer par créer des lignes en combinant les tracés que nous voulons afficher les uns à côté des autres
- puis de les combiner avec le titre et le sous-titre, appelé `forward`.

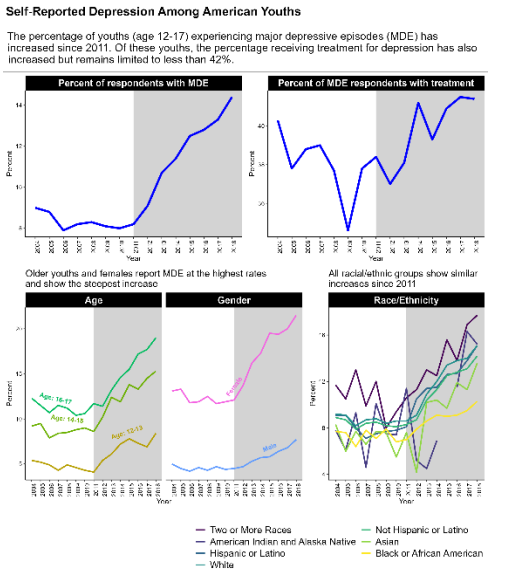
Nous pouvons utiliser l'argument `rel_widths` (largeur relative des colonnes) pour modifier la largeur d'affichage de chaque graphique. Par exemple, dans une grille à deux colonnes, `rel_widths = c(2, 1)` rendra la première colonne deux fois plus large que la seconde.

```
row_1 <- plot_grid(MDE_total_for_mp,
                   treat_for_mp,
                   nrow = 1)

row_2 <- plot_grid(MDE_age_gender_for_mp,
                   MDE_race_for_mp,
                   nrow = 1,
                   rel_widths = c(1, 0.6))
```

Nous pouvons maintenant tout combiner en utilisant à nouveau la fonction `plot_grid()`. Maintenant que nous avons des lignes, nous pouvons tout combiner en une seule colonne et modifier facilement les hauteurs relatives en utilisant la fonction `rel_heights()` afin que notre titre, sous-titre et légende soient tous relativement courts par rapport aux tracés. Nous allons faire en sorte que la première ligne soit deux fois moins haute que la seconde.

Enfin, nous utiliserons la fonction `png()` du paquet `grDevices`. Nous utiliserons la fonction `here()` du package `here`, pour spécifier que nous voulons le sauvegarder dans le répertoire `img` et l'appeler `mainplot.png`. Nous pouvons également utiliser cette fonction pour spécifier la résolution avec `res` et, ce faisant, nous devons sauvegarder l'image avec des spécifications de taille pour l'agrandir.

```
summary_plot = plot_grid(title_plots, forward,
          row_1, row_2,
          legend,
          ncol = 1, rel_heights = c(0.1,0.2,.8, 1, 0.3))

ggsave("summary_plot.png", plot = summary_plot, width = 10, height = 12, dpi = 100)
```



Ça a l'air pas mal du tout ! Mais le graphique est un peu chargé, nous allons donc supprimer le graphique ethnicité afin de simplifier notre graphique.

Cette fois, nous devons recréer notre graphique `MDE_age_genre` car nous voulons séparer nos graphiques pour qu'ils ressemblent davantage aux graphiques du total MDE et du traitement. Nous allons donc créer deux graphiques distincts.

Nous voulons également recoder le texte de la bande au-dessus du graphique et modifier le graphique de manière qu'il n'y ait pas de lignes de grille comme dans la première rangée de graphiques.

Question : Coder les 2 graphiques ci-dessous. Utiliser le `ocs_theme()` pour ces graphiques.

```
#votre code
```

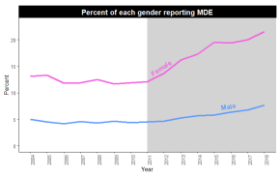


Figure 37. Evolution du pourcentage de EDM par genre

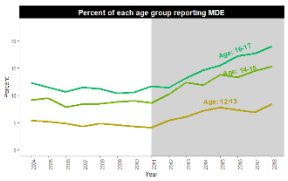
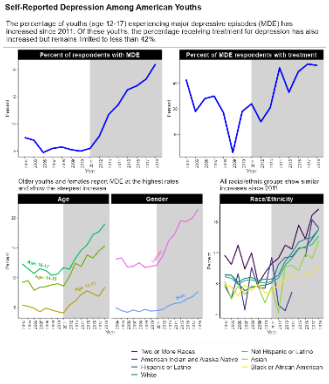


Figure 38. Evolution du pourcentage de EDM par age

Nous allons également créer un autre sous-titre pour les graphiques sur le genre et l'origine ethnique. Cette fois, nous ajouterons du texte en gras à notre texte en utilisant la fonction de base `expression()` et la fonction de base `paste()` (concaténation de chaînes de caractères ensemble).

```
label <- expression(paste(
    bold("Older "), "youths and ",bold("females "),
 "report MDE at the highest rate and also show the steepest increase."), sep = "")

forward2 <- ggdraw() + draw_label(label, size = 16, x = 0,     hjust = -0.01  )
row_2 <- plot_grid(MDE_age, MDE_gender, nrow = 1)

summary_plot_final  = plot_grid(title_plots,forward, row_1, forward2, row_2,
          ncol = 1,
          rel_heights = c(0.1, 0.2, 1, 0.1, 1))
ggsave("summary_plot_final.png", plot = summary_plot, width = 10, height = 12, dpi = 100)
```



7.2. Paragraphe de conclusion

Dans cette étude de cas, nous avons évalué les mesures autodéclarées des symptômes de la dépression chez les jeunes âgés de 12 à 17 ans aux États-Unis, ainsi que le taux de jeunes recevant un traitement pour la dépression. Nous avons appris à récupérer des données directement sur le web à l'aide du package `rvest` et nous avons appris à effectuer et à interpréter un test du χ^2 à l'aide de la fonction `chisq.test()` du package `stats`.

L'analyse et la représentation graphique de nos données montrent clairement que les taux de dépression semblent augmenter, en particulier depuis 2011. Cependant, il est possible que les répondants aient eu des taux similaires les années précédentes, mais qu'ils se sentent maintenant moins stigmatisés lorsqu'ils répondent aux symptômes de la dépression en remplissant l'enquête. L'enquête a toujours été anonyme, mais reporting bias peut parfois amener les individus à exagérer ou à minimiser leurs symptômes parce qu'ils pensent que les chercheurs veulent que leur réponse soit, ou par honte ou embarras, entre autres raisons. Toutefois, les données suggèrent que les jeunes pourraient présenter davantage de symptômes de dépression et que le taux d'augmentation est assez élevé.

Près d'un quart des personnes de genre féminin âgées de 12 à 17 ans ont déclaré présenter des symptômes de dépression. Cela justifie des recherches plus approfondies pour déterminer s'il s'agit d'un phénomène dû à un plus grand nombre de déclarations ou si les jeunes Américaines sont réellement plus déprimées. En outre, si c'est le cas, il est essentiel de déterminer pourquoi les jeunes femmes sont plus déprimées. Une limitation importante est que les données n'incluent pas les intersections de sous-groupes, tels que les taux d'EDM chez les femmes de diverses origines ethniques.

Compte tenu de la très forte augmentation du nombre de femmes, il convient d'étudier plus avant quelles femmes sont particulièrement vulnérables et pourquoi.

8. REFERENCES

Cette étude de cas est une partie d'une étude de cas publique réalisée par le Bloomberg American Health Initiative. Details supplémentaires dans les liens suivants Qier Meng et Michael Breshock

Wright, Carrie and Ontiveros, Michael and Jager, Leah and Taub, Margaret and Hicks, Stephanie C. (2020). https://github.com/opencasestudies/ocs-bp-youth-mental-health. Mental Health of American Youth.

Twenge JM, Cooper AB, Joiner TE, Duffy ME, Binau SG. Age, period, and cohort trends in mood disorder indicators and suicide-related outcomes in a nationally representative dataset, 2005-2017. J Abnorm Psychol.128,3 (2019):185-199. doi:10.1037/abn0000410

Olfson, M., Blanco, C., Wang, S., Laje, G. & Correll, C. U. National Trends in the Mental Health Care of Children, Adolescents, and Adults by Office-Based Physicians. JAMA Psychiatry. 71, 81 (2014):81-90. doi: 10.1001/jamapsychiatry.2013.3074.

Hedegaard H, Curtin SC, Warner M. Suicide rates in the United States continue to increase. NCHS Data Brief, no 309. Hyattsville, MD: National Center for Health Statistics. 2018.

9. ANNEXES

9.1. Annexe 1 : A vous !

Extraire les tableaux 11.5A et 11.5B du site web, qui contiennent des données sur l'obtention d'un traitement chez les jeunes ayant déclaré avoir eu un épisode grave. Créer des graphiques et d'effectuer des tests du χ^2 pour évaluer la comparaison des groupes au fil du temps.

9.2. Annexe 2 : session info

Pour obtenir des détails sur l'exécution du fichier source *Rmd* et des packages utilisés.

```
sessionInfo()
```

Pour obtenir des détails sur le temps d'exécution de ce fichier *Rmd.*

```
rmarkdown::perf_timer_stop("render")
pts = rmarkdown::perf_timer_summary()
cat("About", round(pts$time[1]/1000 + 5), "-", round(pts$time[1]/1000 + 15),"seconds")
```