Notebook - UE Visualisation de données

Quentin Fouché

1 juin 2022

1 Dimension esthétique de la production d'un graphe

1.1 Exploration du jeu de données PhD_v3

[1]:
```
# (1) Import des packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from plotly.subplots import make_subplots
import plotly.graph_objects as go
import sys
!{sys.executable} -m pip install -U kaleido
!{sys.executable} -m pip install bar_chart_race
import bar_chart_race as bcr
!{sys.executable} -m pip install ffmpeg-python
!{sys.executable} -m pip install raceplotly
from raceplotly.plots import barplot
import random
```

[2]:
```
# (2) Import du jeu de données "PhD_v3"
PhD_v3 = pd.read_csv(r"C:\Users\quent\Documents\DU Data Analyst 2022\UE n°2 -␣
↪Visualisation de données\Jeux de données\PhD.v3.csv", low_memory = False,␣
↪encoding='utf-8')
PhD_v3.head()
```

[2]:
```
   Unnamed: 0              Auteur Identifiant auteur  \
0           0      Saeed Al marri                NaN
1           1   Andrea Ramazzotti          174423705
2           2  OLIVIER BODENREIDER                NaN
3           3      Emmanuel Porte                NaN
4           4    Arthur Devriendt                NaN

                                               Titre  \
0  Le credit documentaire et l'onopposabilite des...
1  Application de la PGD a la resolution de probl...
2  Conception d'un outil informatique d'etude des...
3  Socio-histoire des politiques publiques en mat...
```

```
4  LES TECHNOLOGIES DE L'INFORMATION ET DE LA COM...

                          Directeur de these  \
0                        Philippe Delebecque
1  Jean-Claude Grandidier,Marianne Beringhier
2                             Francois Kohler
3                               Gilles Pollet
4                               Gabriel Dupuy

             Directeur de these (nom prenom) Identifiant directeur  \
0                           Delebecque Philippe              29561248
1  Grandidier Jean-Claude,Beringhier Marianne           715,441,511
2                             Kohler Francois              57030758
3                               Pollet Gilles                    na
4                               Dupuy Gabriel                    na

                       Etablissement de soutenance  \
0                                          Paris 1
1  Chasseneuil-du-Poitou, Ecole nationale superie...
2                                          Nancy 1
3                                           Lyon 2
4                                          Paris 1

  Identifiant etablissement  \
0                  27361802
1                  28024400
2                       NaN
3                 02640334X
4                  27361802

                                      Discipline  ...    Year  \
0                                    Driot prive  ...     NaN
1  Mecanique des solides, des materiaux, des stru...  ...     NaN
2                                      Medecine  ...  1993.0
3                                Science politique  ...     NaN
4                                      Geographie  ...     NaN

  Langue de la these Identifiant de la these  Accessible en ligne  \
0                 na                  s69480                  non
1                 na                  s98826                  non
2                 fr            1993NAN19006                  non
3                 na                  s88867                  non
4                 na                  s89663                  non

  Publication dans theses.fr Mise a jour dans theses.fr  \
0                   26-01-12                  26-01-12
1                   22-11-13                  22-11-13
2                   24-05-13                  17-11-12
3                   12-07-13                  12-01-16
4                   13-07-13                  12-07-13
```

```
              Discipline_prÃ©di   Genre  \
0    Droit et Science Politique    male
1  Materiaux, Milieux et Chimie  female
2                      Medecine    male
3    Droit et Science Politique    male
4                           SHS    male

                                  etablissement_rec Langue_rec
0            Université Paris 1 - Panthéon Sorbonne        NaN
1  École nationale supérieure de mécanique et d'a...        NaN
2                            Université de Lorraine   Français
3                       Université Lumière - Lyon 2        NaN
4            Université Paris 1 - Panthéon Sorbonne        NaN

[5 rows x 23 columns]
```

[3]:
```
# (3) Remplacement des caractères "Ã©" par "é"
PhD_v3 = PhD_v3.replace("Ã©", "é", regex = True)
```

[4]:
```
# (4) Identification de la nature des variables
PhD_v3.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 448047 entries, 0 to 448046
Data columns (total 23 columns):
 #   Column                                    Non-Null Count   Dtype
---  ------                                    --------------   -----
 0   Unnamed: 0                                448047 non-null  int64
 1   Auteur                                    448047 non-null  object
 2   Identifiant auteur                        317700 non-null  object
 3   Titre                                     448040 non-null  object
 4   Directeur de these                        448034 non-null  object
 5   Directeur de these (nom prenom)           448034 non-null  object
 6   Identifiant directeur                     448047 non-null  object
 7   Etablissement de soutenance               448046 non-null  object
 8   Identifiant etablissement                 430965 non-null  object
 9   Discipline                                448047 non-null  object
 10  Statut                                    448047 non-null  object
 11  Date de premiere inscription en doctorat  64331 non-null   object
 12  Date de soutenance                        390961 non-null  object
 13  Year                                      390961 non-null  float64
 14  Langue de la these                        448047 non-null  object
 15  Identifiant de la these                   448047 non-null  object
 16  Accessible en ligne                       448047 non-null  object
 17  Publication dans theses.fr                448047 non-null  object
 18  Mise a jour dans theses.fr                447870 non-null  object
 19  Discipline_prÃ©di                         448047 non-null  object
 20  Genre                                     448047 non-null  object
 21  etablissement_rec                         444973 non-null  object
 22  Langue_rec                                383927 non-null  object
dtypes: float64(1), int64(1), object(21)
```

```
memory usage: 78.6+ MB
```

[5]:
```
# (5) Suppression de la colonne "Unnamed: 0"
PhD_v3 = PhD_v3.drop(columns="Unnamed: 0")
```

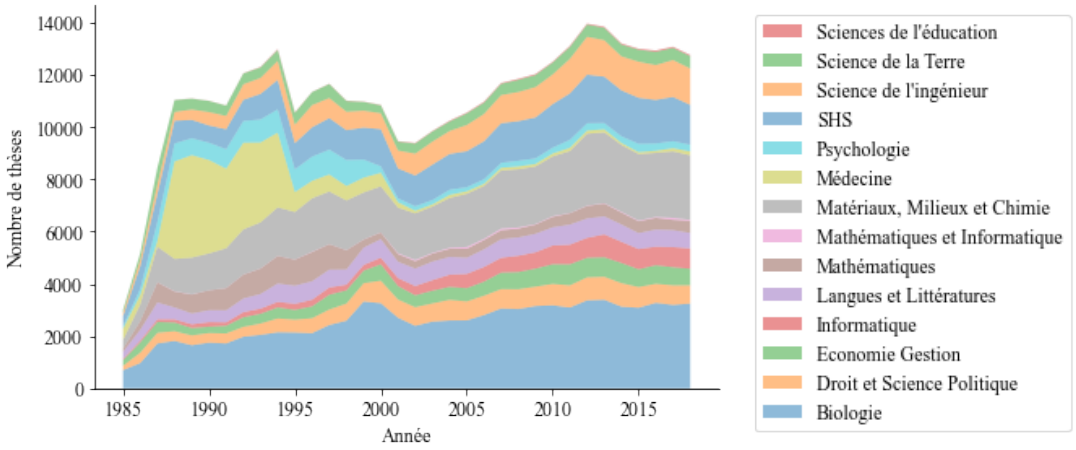
1.2 Exercice 1

[6]:
```
# (1) Sélection des données sur la période 1985-2018
PhD_v3["Date de soutenance"] = pd.to_datetime(PhD_v3["Date de soutenance"],␣
↪format = "%d-%m-%y")
PhD_v3_1985_2018 = PhD_v3[np.logical_and(PhD_v3["Date de soutenance"] >=␣
↪"1985-01-01", PhD_v3["Date de soutenance"] < "2019-01-01")]
```

[7]:
```
# (2) Calcul du nombre de thèses soutenues par discipline et par année
PhD_v3_1985_2018["Discipline_predi"] = PhD_v3_1985_2018["Discipline_prÃ©di"]
PhD_v3_1985_2018.drop(columns="Discipline_prÃ©di")
PhD_v3_1985_2018 = PhD_v3_1985_2018.replace("education", "éducation", regex =␣
↪True)
PhD_v3_1985_2018 = PhD_v3_1985_2018.replace("Medecine", "Médecine", regex =␣
↪True)
PhD_v3_1985_2018 = PhD_v3_1985_2018.replace("Mathematiques", "Mathématiques",␣
↪regex = True)
PhD_v3_1985_2018 = PhD_v3_1985_2018.replace("Materiaux", "Matériaux", regex =␣
↪True)
PhD_v3_1985_2018 = PhD_v3_1985_2018.replace("Litteratures", "Littératures",␣
↪regex = True)
PhD_v3_discipline_annee = PhD_v3_1985_2018.pivot_table(values = "Date de␣
↪soutenance", index = "Discipline_predi", columns = "Year", aggfunc =␣
↪"count", fill_value = 0)
PhD_v3_discipline_annee.reset_index(inplace = True)
Poubelle =␣
↪PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
↪"Poubelle"].index
PhD_v3_discipline_annee = PhD_v3_discipline_annee.
↪drop(PhD_v3_discipline_annee.index[Poubelle])
```

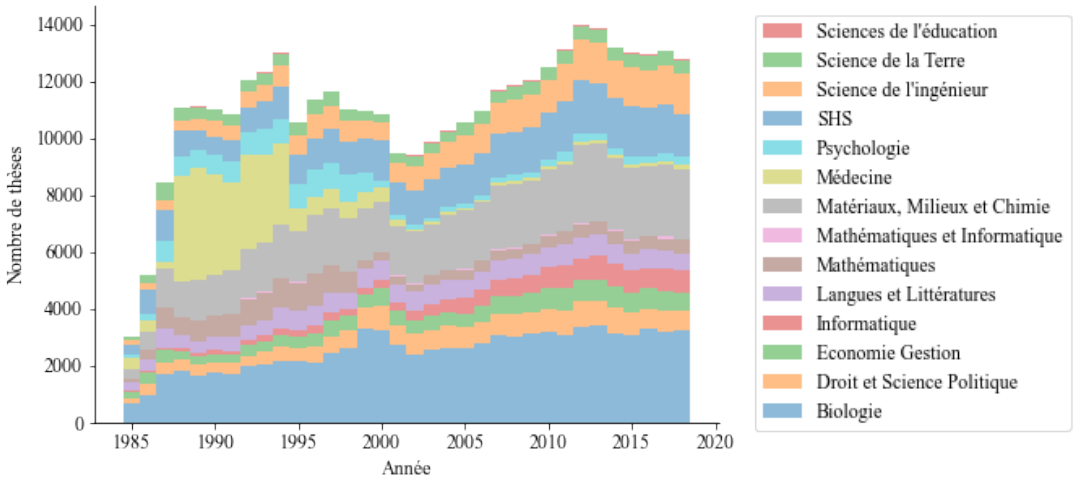
[8]:
```
# (3) Représentation de l'évolution du nombre de thèses soutenues au fil des␣
↪ans en fonction de la discipline
Dict_PhD_v3_discipline_annee = {}
for i in list(PhD_v3_discipline_annee["Discipline_predi"]):
    Dict_PhD_v3_discipline_annee[i] =␣
↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
↪i].values[0][1:])

plt.rc('font', family = 'Times New Roman', size = 12)
fig, ax = plt.subplots(figsize=(17.5/2.54, 11/2.54))
ax.stackplot(list(PhD_v3_discipline_annee.columns)[1:],␣
↪Dict_PhD_v3_discipline_annee.values(), labels =␣
↪list(PhD_v3_discipline_annee["Discipline_predi"]), alpha = 0.5)
handles, labels = ax.get_legend_handles_labels()
```

```
ax.legend(reversed(handles), reversed(labels), bbox_to_anchor=(1, 1.68),␣
↪frameon = False, ncol = 2)
ax.set_xlabel("Année")
ax.set_ylabel("Nombre de thèses")
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['top'].set_visible(False)
plt.show()
fig.savefig("Figure_1.pdf", bbox_inches = 'tight')
```



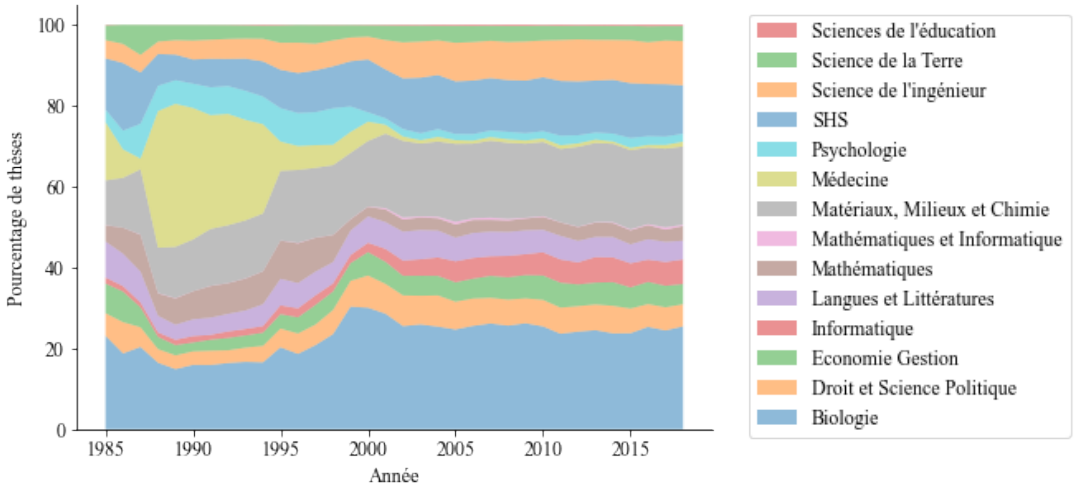
[9]:
```
# (4) Représentation de cette même évolution sous forme d'histogramme empilé
List_discipline = list(PhD_v3_discipline_annee["Discipline_predi"])
List_annees = list(PhD_v3_discipline_annee.columns[1:])
k = np.
↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
↪== "Biologie"][List_annees].values[0]))

plt.rc('font', family = 'Times New Roman', size = 12)
fig, ax = plt.subplots(figsize=(17.5/2.54, 12/2.54))
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['top'].set_visible(False)
ax.set_xlabel("Année")
ax.set_ylabel("Nombre de thèses")
ax.bar(List_annees,␣
↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
↪"Biologie"][List_annees].values[0]), label = "Biologie", width = 1, alpha =␣
↪0.5)
for j in list(range(1,len(List_discipline))):
    ax.bar(List_annees,␣
↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
↪List_discipline[j]][List_annees].values[0]), bottom = k, label =␣
↪List_discipline[j], width = 1, alpha = 0.5)
    k = k + np.
↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
↪== List_discipline[j]][List_annees].values[0]))
```

```
handles, labels = ax.get_legend_handles_labels()
ax.legend(reversed(handles), reversed(labels), bbox_to_anchor=(1.58, 1))
```



[10]:
```
# (5) Calcul du pourcentage de thèses soutenues par discipline et par année
PhD_v3_prop_discipline_annee = PhD_v3_discipline_annee.copy(deep = True)

for i in list(PhD_v3_prop_discipline_annee.columns)[1:]:
    PhD_v3_prop_discipline_annee[i] = (PhD_v3_prop_discipline_annee[i] /␣
 ↪PhD_v3_prop_discipline_annee[i].sum()) * 100
```

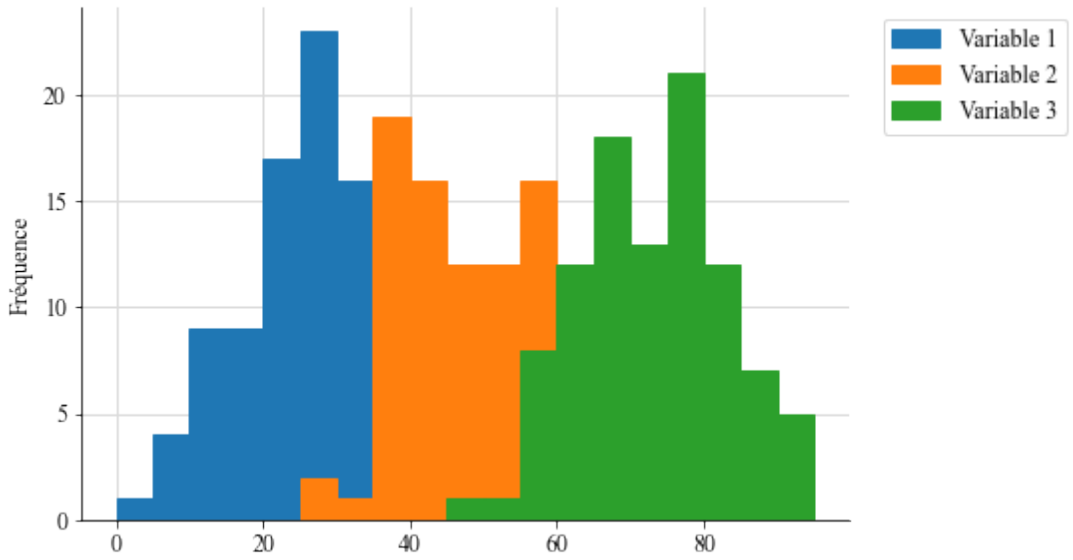
[11]:
```
# (6) Représentation de l'évolution du pourcentage de thèses soutenues au fil␣
 ↪des ans en fonction de la discipline
Dict_PhD_v3_prop_discipline_annee = {}
for i in list(PhD_v3_prop_discipline_annee["Discipline_predi"]):
    Dict_PhD_v3_prop_discipline_annee[i] =␣
 ↪list(PhD_v3_prop_discipline_annee[PhD_v3_prop_discipline_annee["Discipline_predi"]␣
 ↪== i].values[0][1:])

plt.rc('font', family = 'Times New Roman', size = 12)
fig, ax = plt.subplots(figsize=(17.5/2.54, 12/2.54))
ax.stackplot(list(PhD_v3_prop_discipline_annee.columns)[1:],␣
 ↪Dict_PhD_v3_prop_discipline_annee.values(), labels =␣
 ↪list(PhD_v3_prop_discipline_annee["Discipline_predi"]), alpha = 0.5)
handles, labels = ax.get_legend_handles_labels()
ax.legend(reversed(handles), reversed(labels), bbox_to_anchor=(1.58, 1))
ax.set_xlabel("Année")
ax.set_ylabel("Pourcentage de thèses")
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['top'].set_visible(False)
```



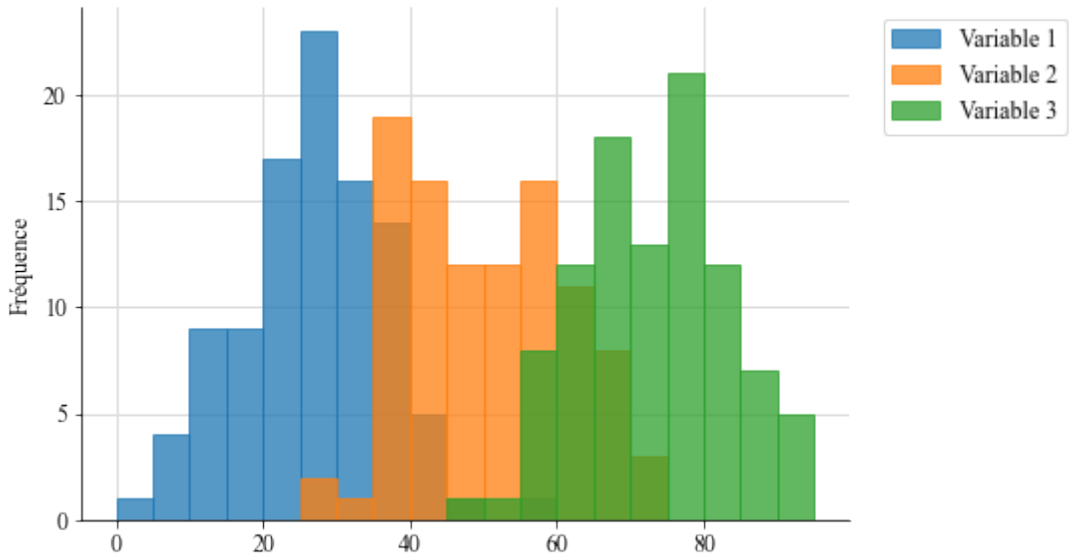
1.3 Exercice 2

[12]:
```
# (1) Création de trois variables comprenant chacune 100 nombres entiers␣
↪aléatoires dont la distribution suit une loi normale
var1 = [int(x) for x in np.random.normal(25, 10, 100)]
var2 = [int(x) for x in np.random.normal(50, 10, 100)]
var3 = [int(x) for x in np.random.normal(75, 10, 100)]
```

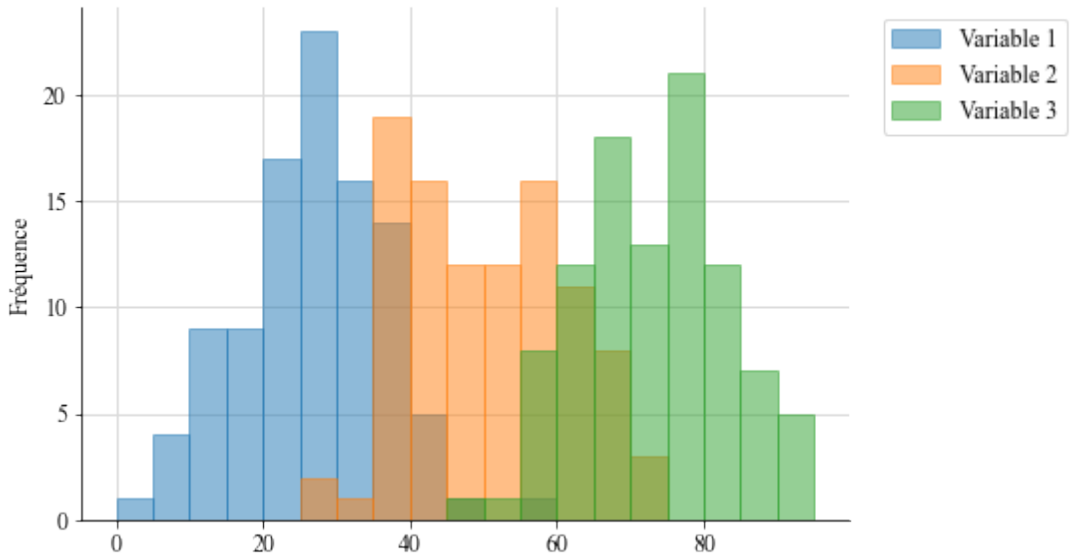
[13]:
```
# (2) Représentation de la distribution de ces trois variables, sans␣
↪transparence
plt.rc('font', family = 'Times New Roman', size = 12)
fig, ax = plt.subplots(figsize=(17.5/2.54, 12/2.54))
ax.hist(var1, label = "Variable 1", alpha = 1, edgecolor = "tab:blue", bins =␣
↪list(range(0,99,5)))
ax.hist(var2, label = "Variable 2", alpha = 1, edgecolor = "tab:orange", bins␣
↪= list(range(0,99,5)))
ax.hist(var3, label = "Variable 3", alpha = 1, edgecolor = "tab:green", bins␣
↪= list(range(0,99,5)))
ax.grid(True, color = "gainsboro", linestyle = '-', linewidth = 1)
ax.set_axisbelow(True)
ax.legend(bbox_to_anchor=(1.30, 1))
ax.set_ylabel("Fréquence")
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['top'].set_visible(False)
```



[14]:
```
# (3) Représentation de la distribution de ces trois variables avec une␣
 ↪transparence de 75%
plt.rc('font', family = 'Times New Roman', size = 12)
fig, ax = plt.subplots(figsize=(17.5/2.54, 12/2.54))
ax.hist(var1, label = "Variable 1", alpha = 0.75, edgecolor = "tab:blue",␣
 ↪bins = list(range(0,99,5)))
ax.hist(var2, label = "Variable 2", alpha = 0.75, edgecolor = "tab:orange",␣
 ↪bins = list(range(0,99,5)))
ax.hist(var3, label = "Variable 3", alpha = 0.75, edgecolor = "tab:green",␣
 ↪bins = list(range(0,99,5)))
ax.grid(True, color = "gainsboro", linestyle = '-', linewidth = 1)
ax.set_axisbelow(True)
ax.legend(bbox_to_anchor=(1.30, 1))
ax.set_ylabel("Fréquence")
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['top'].set_visible(False)
```



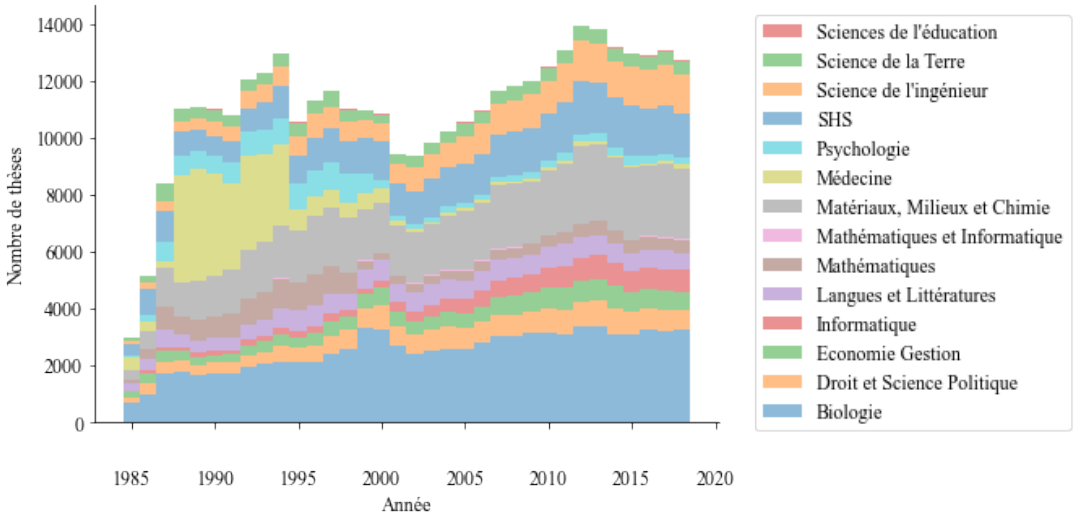
[15]:
```
# (4) Représentation de la distribution de ces trois variables avec une␣
 ↪transparence de 50%
plt.rc('font', family = 'Times New Roman', size = 12)
fig, ax = plt.subplots(figsize=(17.5/2.54, 12/2.54))
ax.hist(var1, label = "Variable 1", alpha = 0.5, edgecolor = "tab:blue", bins␣
 ↪= list(range(0,99,5)))
ax.hist(var2, label = "Variable 2", alpha = 0.5, edgecolor = "tab:orange",␣
 ↪bins = list(range(0,99,5)))
ax.hist(var3, label = "Variable 3", alpha = 0.5, edgecolor = "tab:green",␣
 ↪bins = list(range(0,99,5)))
ax.grid(True, color = "gainsboro", linestyle = '-', linewidth = 1)
ax.set_axisbelow(True)
ax.legend(bbox_to_anchor=(1.30, 1))
ax.set_ylabel("Fréquence")
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['top'].set_visible(False)
```



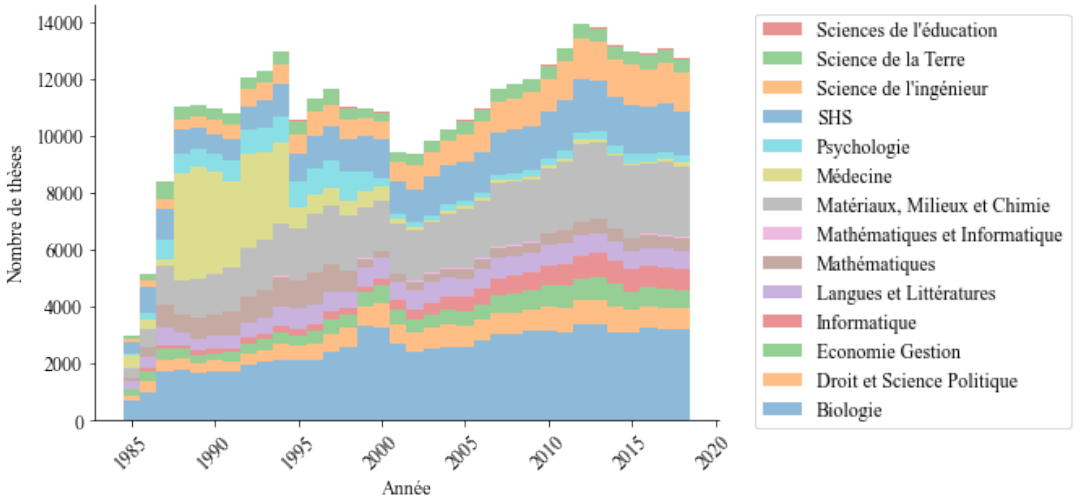
1.4 Exercice 3

Dans cet exercice et les suivants, le graphique utilisé est l'histogramme empilé de l'exercice 1 (évolution du nombre de thèses soutenues au fil des ans en fonction de la discipline).

[16]:
```
# (1) Augmentation de la distance entre l'axe des abscisses et ses labels
k = np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== "Biologie"][List_annees].values[0]))
plt.rc('font', family = 'Times New Roman', size = 12)
fig, ax = plt.subplots(figsize=(17.5/2.54, 12/2.54))
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['top'].set_visible(False)
ax.set_xlabel("Année")
ax.set_ylabel("Nombre de thèses")
ax.tick_params(axis = "x", pad = 25)
ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪"Biologie"][List_annees].values[0]), label = "Biologie", width = 1, alpha =␣
 ↪0.5)
for j in list(range(1,len(List_discipline))):
    ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪List_discipline[j]][List_annees].values[0]), bottom = k, label =␣
 ↪List_discipline[j], width = 1, alpha = 0.5)
    k = k + np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== List_discipline[j]][List_annees].values[0]))
handles, labels = ax.get_legend_handles_labels()
ax.legend(reversed(handles), reversed(labels), bbox_to_anchor=(1.58, 1))
```

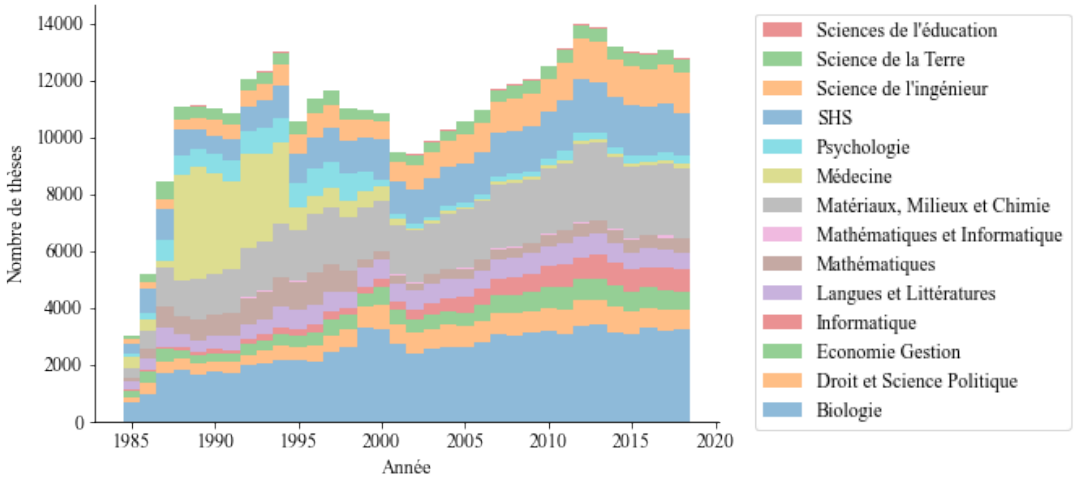


[17]:
```
# (2) Inclinaison des labels de l'axe des abscisses de 45°
k = np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== "Biologie"][List_annees].values[0]))
plt.rc('font', family = 'Times New Roman', size = 12)
fig, ax = plt.subplots(figsize=(17.5/2.54, 12/2.54))
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['top'].set_visible(False)
ax.set_xlabel("Année")
ax.set_ylabel("Nombre de thèses")
ax.tick_params(axis = "x", rotation = 45)
ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪"Biologie"][List_annees].values[0]), label = "Biologie", width = 1, alpha =␣
 ↪0.5)
for j in list(range(1,len(List_discipline))):
    ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪List_discipline[j]][List_annees].values[0]), bottom = k, label =␣
 ↪List_discipline[j], width = 1, alpha = 0.5)
    k = k + np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== List_discipline[j]][List_annees].values[0]))
handles, labels = ax.get_legend_handles_labels()
ax.legend(reversed(handles), reversed(labels), bbox_to_anchor=(1.58, 1))
```

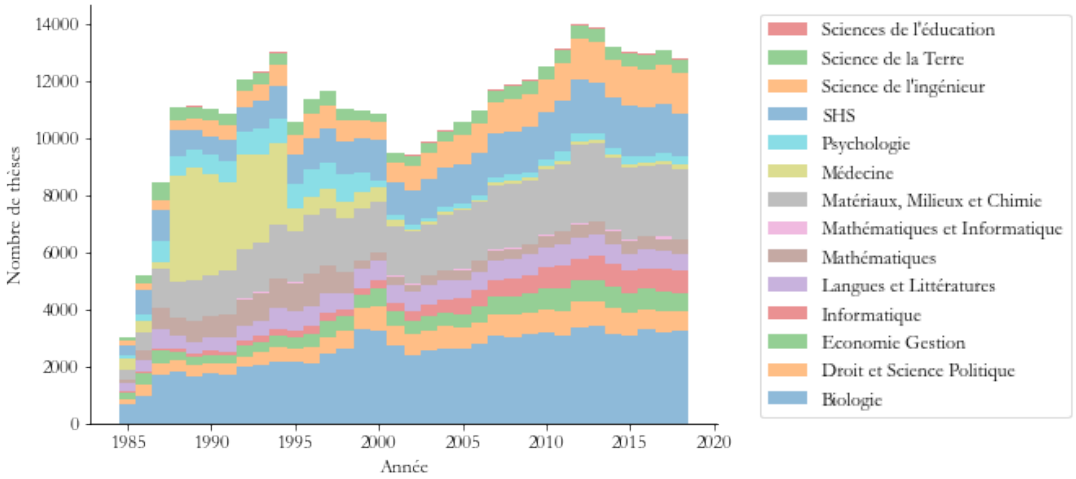


1.5 Exercice 4

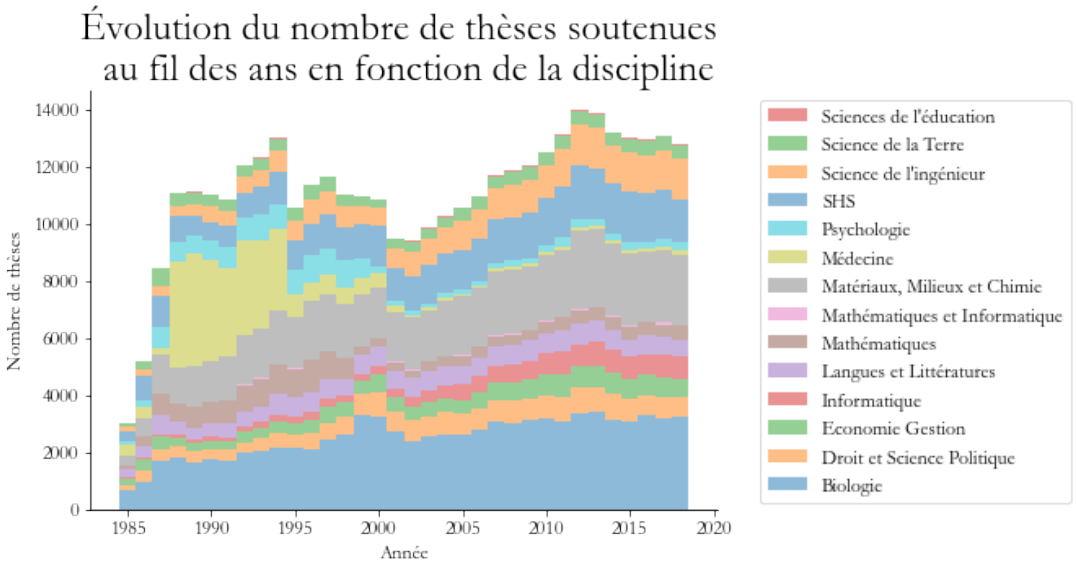
[18]:
```
# (1) Affichage de la police en Times New Roman
k = np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== "Biologie"][List_annees].values[0]))
plt.rc('font', family = 'Times New Roman', size = 12)
fig, ax = plt.subplots(figsize=(17.5/2.54, 12/2.54))
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['top'].set_visible(False)
ax.set_xlabel("Année")
ax.set_ylabel("Nombre de thèses")
ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪"Biologie"][List_annees].values[0]), label = "Biologie", width = 1, alpha =␣
 ↪0.5)
for j in list(range(1,len(List_discipline))):
    ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪List_discipline[j]][List_annees].values[0]), bottom = k, label =␣
 ↪List_discipline[j], width = 1, alpha = 0.5)
    k = k + np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== List_discipline[j]][List_annees].values[0]))
handles, labels = ax.get_legend_handles_labels()
ax.legend(reversed(handles), reversed(labels), bbox_to_anchor=(1.58, 1))
```



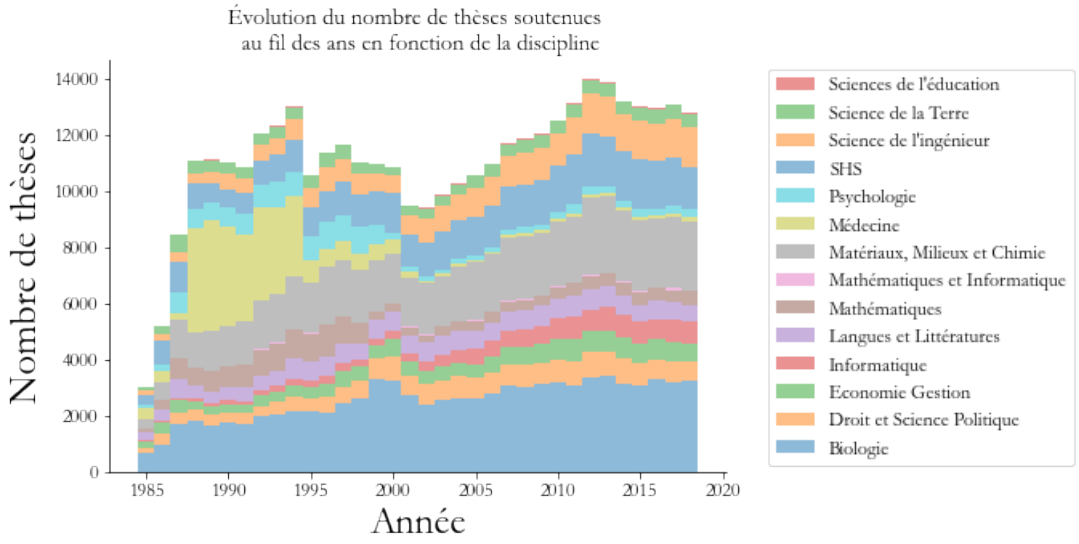
[19]:
```
# (2) Changement de la police en Garamond
k = np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== "Biologie"][List_annees].values[0]))
plt.rc('font', family = 'Garamond', size = 12)
fig, ax = plt.subplots(figsize=(17.5/2.54, 12/2.54))
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['top'].set_visible(False)
ax.set_xlabel("Année")
ax.set_ylabel("Nombre de thèses")
ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪"Biologie"][List_annees].values[0]), label = "Biologie", width = 1, alpha =␣
 ↪0.5)
for j in list(range(1,len(List_discipline))):
    ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪List_discipline[j]][List_annees].values[0]), bottom = k, label =␣
 ↪List_discipline[j], width = 1, alpha = 0.5)
    k = k + np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== List_discipline[j]][List_annees].values[0]))
handles, labels = ax.get_legend_handles_labels()
ax.legend(reversed(handles), reversed(labels), bbox_to_anchor=(1.58, 1))
```



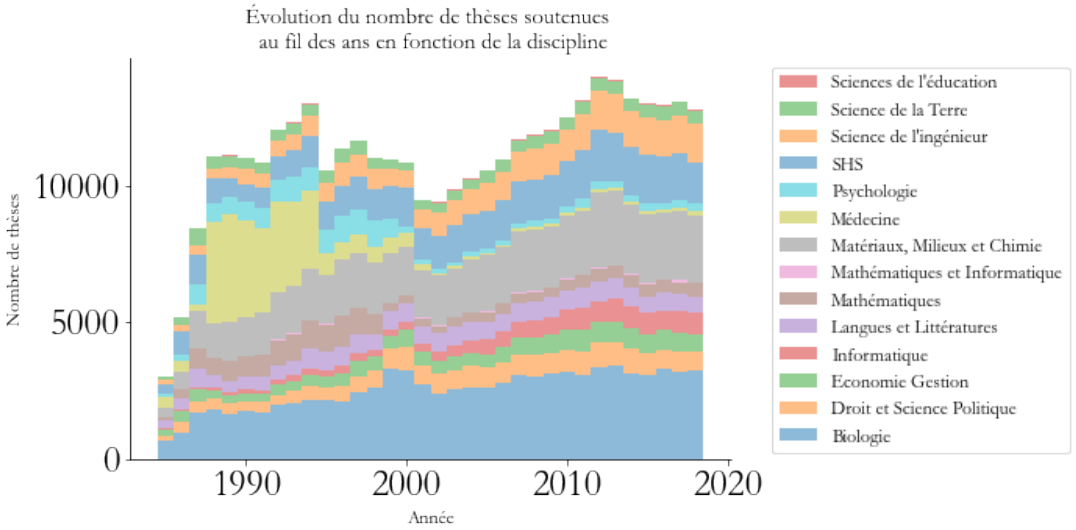
[20]:
```
# (3) Ajout d'un titre avec une police deux fois plus grosse que celle des␣
 ↪axes
k = np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== "Biologie"][List_annees].values[0]))
plt.rc('font', family = 'Garamond', size = 12)
fig, ax = plt.subplots(figsize=(17.5/2.54, 12/2.54))
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['top'].set_visible(False)
ax.set_xlabel("Année")
ax.set_ylabel("Nombre de thèses")
ax.set_title("Évolution du nombre de thèses soutenues \n au fil des ans en␣
 ↪fonction de la discipline", size = 24)
ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪"Biologie"][List_annees].values[0]), label = "Biologie", width = 1, alpha =␣
 ↪0.5)
for j in list(range(1,len(List_discipline))):
    ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪List_discipline[j]][List_annees].values[0]), bottom = k, label =␣
 ↪List_discipline[j], width = 1, alpha = 0.5)
    k = k + np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== List_discipline[j]][List_annees].values[0]))
handles, labels = ax.get_legend_handles_labels()
ax.legend(reversed(handles), reversed(labels), bbox_to_anchor=(1.58, 1))
```



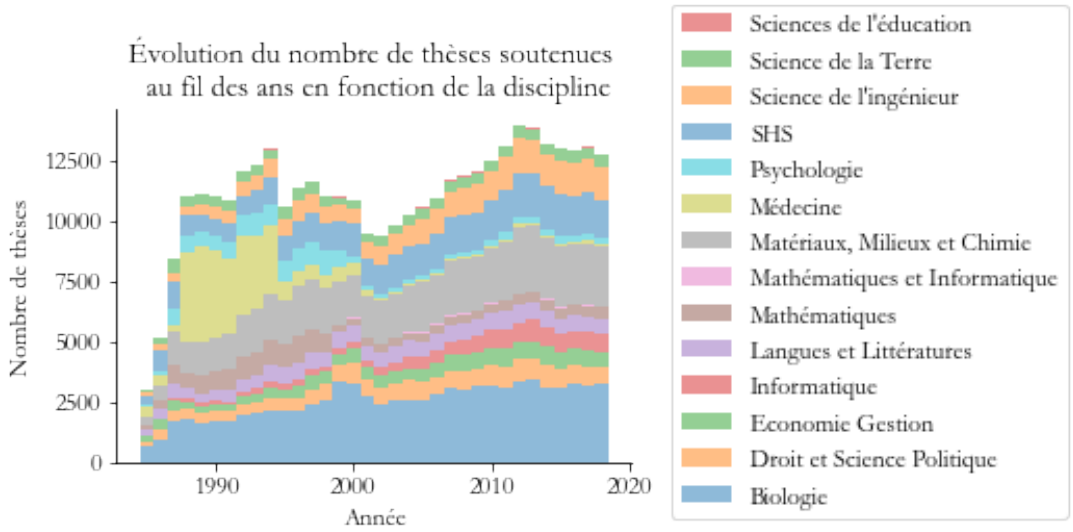
[21]:
```
# (4) Doublement de la taille de la police des titres des axes
k = np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== "Biologie"][List_annees].values[0]))
plt.rc('font', family = 'Garamond', size = 12)
fig, ax = plt.subplots(figsize=(17.5/2.54, 12/2.54))
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['top'].set_visible(False)
ax.set_xlabel("Année", size = 24)
ax.set_ylabel("Nombre de thèses", size = 24)
ax.set_title("Évolution du nombre de thèses soutenues \n au fil des ans en␣
 ↪fonction de la discipline")
ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪"Biologie"][List_annees].values[0]), label = "Biologie", width = 1, alpha =␣
 ↪0.5)
for j in list(range(1,len(List_discipline))):
    ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪List_discipline[j]][List_annees].values[0]), bottom = k, label =␣
 ↪List_discipline[j], width = 1, alpha = 0.5)
    k = k + np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== List_discipline[j]][List_annees].values[0]))
handles, labels = ax.get_legend_handles_labels()
ax.legend(reversed(handles), reversed(labels), bbox_to_anchor=(1.58, 1))
```



[22]:
```
# (5) Augmentation de la taille de la police uniquement pour les labels des␣
 ↪axes
k = np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== "Biologie"][List_annees].values[0]))
plt.rc('font', family = 'Garamond', size = 12)
fig, ax = plt.subplots(figsize=(17.5/2.54, 12/2.54))
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['top'].set_visible(False)
ax.set_xlabel("Année")
ax.set_ylabel("Nombre de thèses")
ax.set_title("Évolution du nombre de thèses soutenues \n au fil des ans en␣
 ↪fonction de la discipline")
ax.tick_params(axis = "both", labelsize = 24)
ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪"Biologie"][List_annees].values[0]), label = "Biologie", width = 1, alpha =␣
 ↪0.5)
for j in list(range(1,len(List_discipline))):
    ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪List_discipline[j]][List_annees].values[0]), bottom = k, label =␣
 ↪List_discipline[j], width = 1, alpha = 0.5)
    k = k + np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== List_discipline[j]][List_annees].values[0]))
handles, labels = ax.get_legend_handles_labels()
ax.legend(reversed(handles), reversed(labels), bbox_to_anchor=(1.58, 1))
```

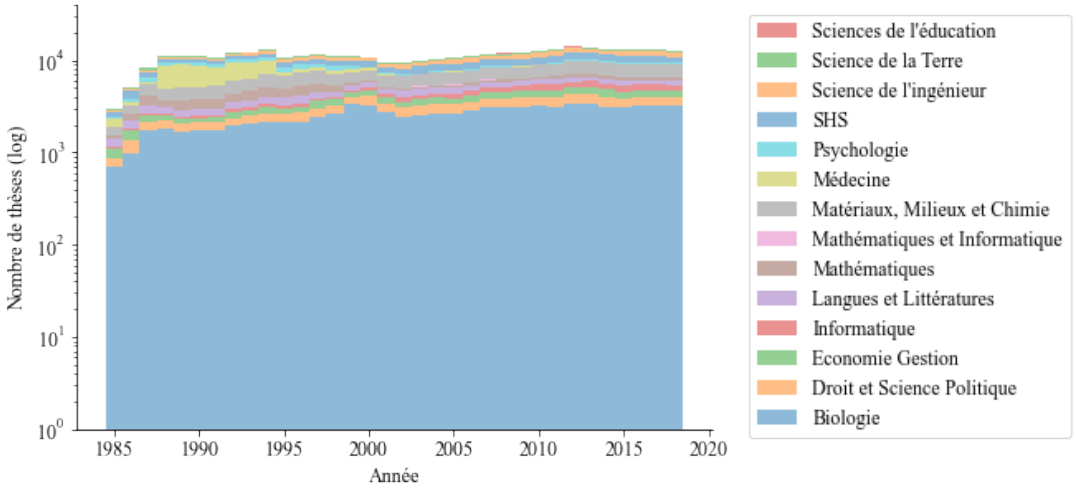


[23]:
```
# (6) Augmentation des marges (i.e. "écrasement" du graphique vers le centre)
k = np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== "Biologie"][List_annees].values[0]))
plt.rc('font', family = 'Garamond', size = 12)
fig, ax = plt.subplots(figsize=(17.5/2.54, 12/2.54))
plt.subplots_adjust(left = 0, bottom = 0, right = 0.5, top = 0.5)
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['top'].set_visible(False)
ax.set_xlabel("Année")
ax.set_ylabel("Nombre de thèses")
ax.set_title("Évolution du nombre de thèses soutenues \n au fil des ans en␣
 ↪fonction de la discipline")
ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪"Biologie"][List_annees].values[0]), label = "Biologie", width = 1, alpha =␣
 ↪0.5)
for j in list(range(1,len(List_discipline))):
    ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪List_discipline[j]][List_annees].values[0]), bottom = k, label =␣
 ↪List_discipline[j], width = 1, alpha = 0.5)
    k = k + np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== List_discipline[j]][List_annees].values[0]))
handles, labels = ax.get_legend_handles_labels()
ax.legend(reversed(handles), reversed(labels), bbox_to_anchor=(1.05, 1.33))
```



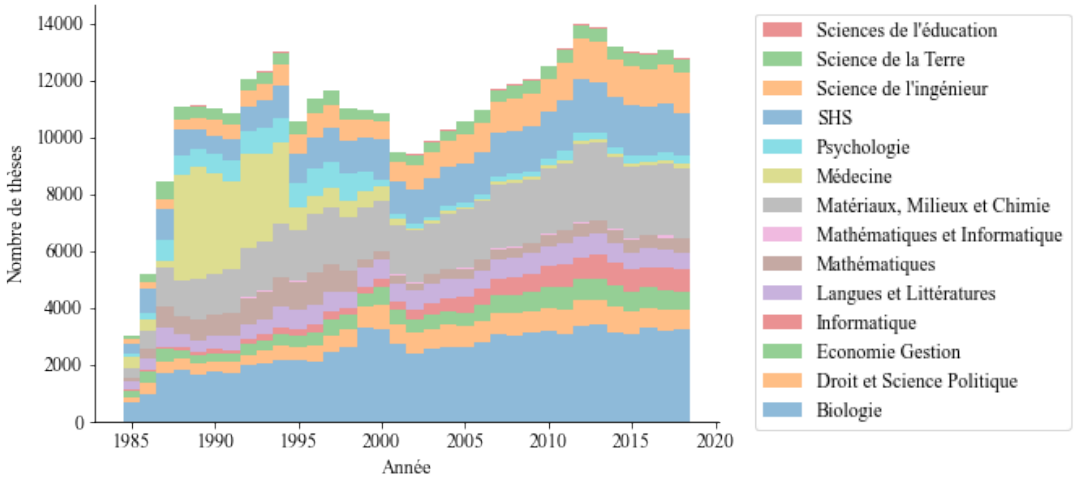
1.6 Exercice 5

[24]:
```
# (1) Conversion de l'axe des ordonnées en logarithme décimal
k = np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== "Biologie"][List_annees].values[0]))
plt.rc('font', family = 'Times New Roman', size = 12)
fig, ax = plt.subplots(figsize=(17.5/2.54, 12/2.54))
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['top'].set_visible(False)
ax.set_xlabel("Année")
ax.set_ylabel("Nombre de thèses (log)")
plt.yscale('log')
ax.set(ylim=(1, 40000))
ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪"Biologie"][List_annees].values[0]), label = "Biologie", width = 1, alpha =␣
 ↪0.5)
for j in list(range(1,len(List_discipline))):
    ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪List_discipline[j]][List_annees].values[0]), bottom = k, label =␣
 ↪List_discipline[j], width = 1, alpha = 0.5)
    k = k + np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== List_discipline[j]][List_annees].values[0]))
handles, labels = ax.get_legend_handles_labels()
ax.legend(reversed(handles), reversed(labels), bbox_to_anchor=(1.58, 1))
```

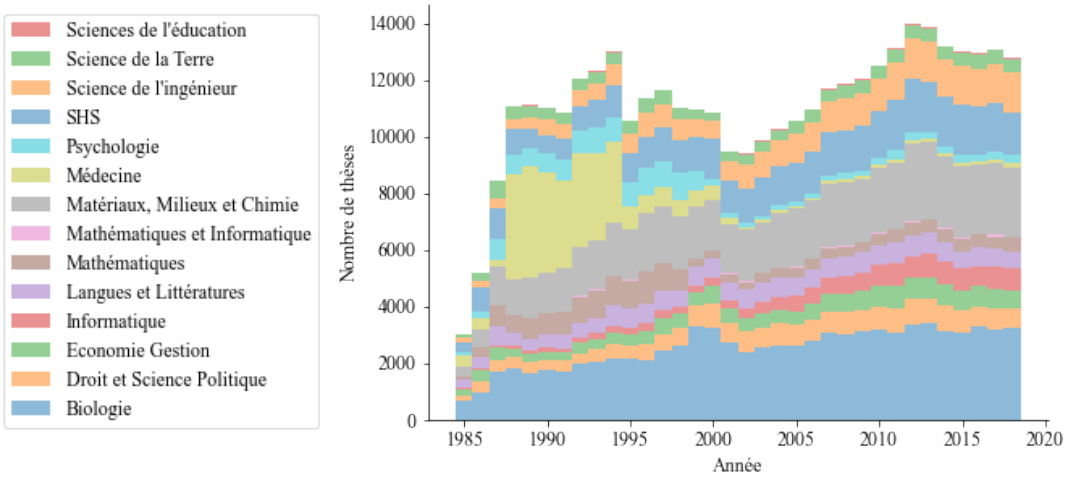


1.7 Exercice 6

[25]:
```
# (1) Affichage de la légende à droite du graphique
k = np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== "Biologie"][List_annees].values[0]))
plt.rc('font', family = 'Times New Roman', size = 12)
fig, ax = plt.subplots(figsize=(17.5/2.54, 12/2.54))
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['top'].set_visible(False)
ax.set_xlabel("Année")
ax.set_ylabel("Nombre de thèses")
ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪"Biologie"][List_annees].values[0]), label = "Biologie", width = 1, alpha =␣
 ↪0.5)
for j in list(range(1,len(List_discipline))):
    ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪List_discipline[j]][List_annees].values[0]), bottom = k, label =␣
 ↪List_discipline[j], width = 1, alpha = 0.5)
    k = k + np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== List_discipline[j]][List_annees].values[0]))
handles, labels = ax.get_legend_handles_labels()
ax.legend(reversed(handles), reversed(labels), bbox_to_anchor=(1.58, 1))
```

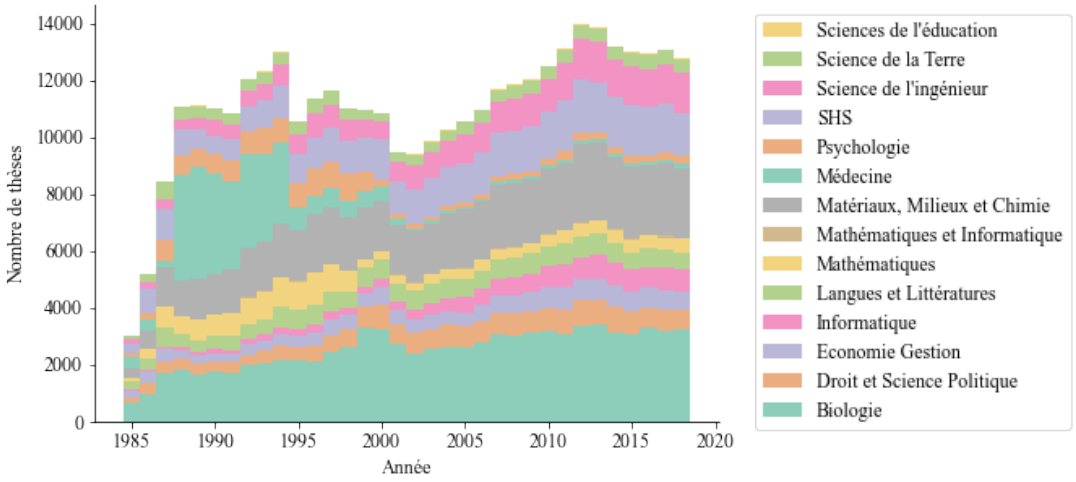


[26]:
```
# (2) Affichage de la légende à gauche du graphique
k = np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== "Biologie"][List_annees].values[0]))
plt.rc('font', family = 'Times New Roman', size = 12)
fig, ax = plt.subplots(figsize=(17.5/2.54, 12/2.54))
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['top'].set_visible(False)
ax.set_xlabel("Année")
ax.set_ylabel("Nombre de thèses")
ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪"Biologie"][List_annees].values[0]), label = "Biologie", width = 1, alpha =␣
 ↪0.5)
for j in list(range(1,len(List_discipline))):
    ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪List_discipline[j]][List_annees].values[0]), bottom = k, label =␣
 ↪List_discipline[j], width = 1, alpha = 0.5)
    k = k + np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== List_discipline[j]][List_annees].values[0]))
handles, labels = ax.get_legend_handles_labels()
ax.legend(reversed(handles), reversed(labels), bbox_to_anchor=(-0.16, 1))
```



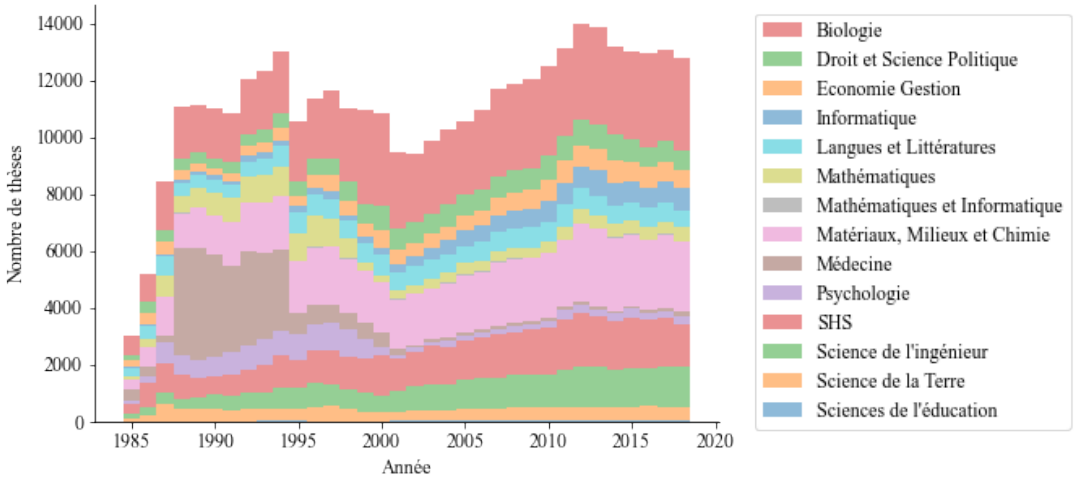
1.8 Exercice 7

[27]:
```
# (1) Modification de la palette de couleurs (sélection de la palette "Dark2"
 ↪du package seaborn)
k = np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]
 ↪== "Biologie"][List_annees].values[0]))
sns.set_palette("Dark2")
plt.rc('font', family = 'Times New Roman', size = 12)
fig, ax = plt.subplots(figsize=(17.5/2.54, 12/2.54))
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['top'].set_visible(False)
ax.set_xlabel("Année")
ax.set_ylabel("Nombre de thèses")
ax.bar(List_annees,
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==
 ↪"Biologie"][List_annees].values[0]), label = "Biologie", width = 1, alpha =
 ↪0.5)
for j in list(range(1,len(List_discipline))):
    ax.bar(List_annees,
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==
 ↪List_discipline[j]][List_annees].values[0]), bottom = k, label =
 ↪List_discipline[j], width = 1, alpha = 0.5)
    k = k + np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]
 ↪== List_discipline[j]][List_annees].values[0]))
handles, labels = ax.get_legend_handles_labels()
ax.legend(reversed(handles), reversed(labels), bbox_to_anchor=(1.58, 1))
```



1.9 Exercice 8

[28]:
```
# (1) Inversion de l'ordre de représentation des disciplines dans le graphique
List_discipline = list(PhD_v3_discipline_annee["Discipline_predi"])
List_discipline.reverse()
k = np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== "Sciences de l'éducation"][List_annees].values[0]))
sns.set_palette("tab10")
plt.rc('font', family = 'Times New Roman', size = 12)
fig, ax = plt.subplots(figsize=(17.5/2.54, 12/2.54))
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['top'].set_visible(False)
ax.set_xlabel("Année")
ax.set_ylabel("Nombre de thèses")
ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪"Sciences de l'éducation"][List_annees].values[0]), label = "Sciences de␣
 ↪l'éducation", width = 1, alpha = 0.5)
for j in list(range(1,len(List_discipline))):
    ax.bar(List_annees,␣
 ↪list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"] ==␣
 ↪List_discipline[j]][List_annees].values[0]), bottom = k, label =␣
 ↪List_discipline[j], width = 1, alpha = 0.5)
    k = k + np.
 ↪array(list(PhD_v3_discipline_annee[PhD_v3_discipline_annee["Discipline_predi"]␣
 ↪== List_discipline[j]][List_annees].values[0]))
handles, labels = ax.get_legend_handles_labels()
ax.legend(reversed(handles), reversed(labels), bbox_to_anchor=(1.58, 1))
```



2 Production de graphes animés et interactifs

2.1 Exercice 9

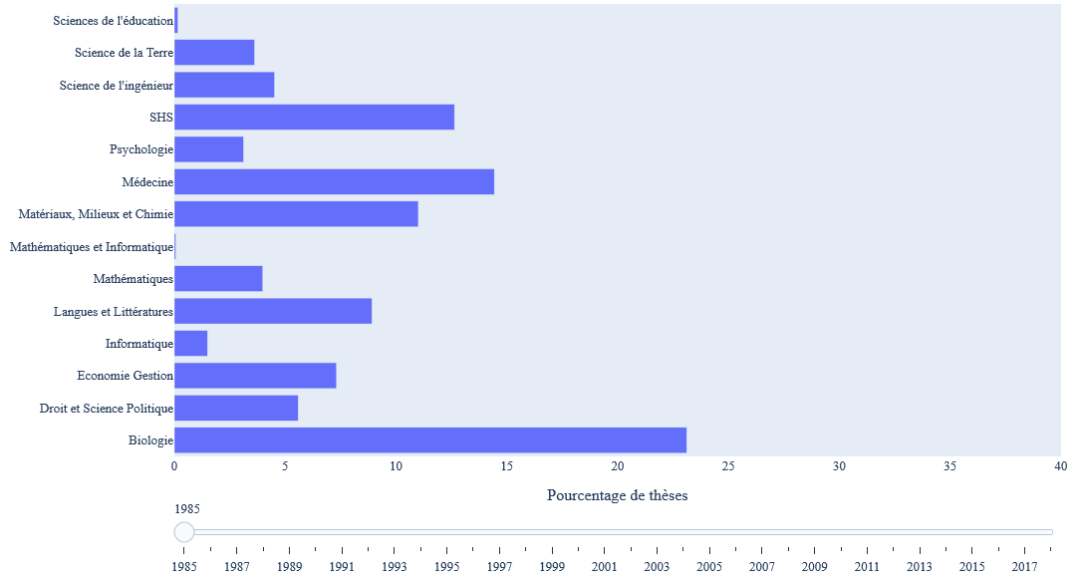
[29]:
```
# (1) Calcul du pourcentage de thèses rédigées en anglais en fonction de la␣
 ↪discipline et de l'année, entre 1985 et 2018
PhD_v3_1985_2018_ang = PhD_v3_1985_2018[PhD_v3_1985_2018["Langue_rec"] ==␣
 ↪"Anglais"]
PhD_v3_discipline_ang = PhD_v3_1985_2018_ang.pivot_table(values = "Date de␣
 ↪soutenance", index = "Discipline_predi", columns = "Year", aggfunc =␣
 ↪"count", fill_value = 0)
for i in list(PhD_v3_discipline_ang.columns)[1:]:
    PhD_v3_discipline_ang[i] = (PhD_v3_discipline_ang[i] /␣
 ↪PhD_v3_discipline_ang[i].sum()) * 100
PhD_v3_discipline_ang.reset_index(inplace = True)
PhD_v3_discipline_ang = PhD_v3_discipline_ang.transpose()
PhD_v3_discipline_ang.columns = list(PhD_v3_discipline_ang.iloc[0,:])
PhD_v3_discipline_ang = PhD_v3_discipline_ang.drop(PhD_v3_discipline_ang.
 ↪index[0])
my_list = []
for i in range(0,len(PhD_v3_discipline_ang.index)): my_list.
 ↪append(int(PhD_v3_discipline_ang.index[i]))
PhD_v3_discipline_ang.index = pd.to_datetime(my_list, format = "%Y")
PhD_v3_discipline_ang = PhD_v3_discipline_ang.astype("int64")
```

[30]:
```
# (2) Représentation sous forme d'histogramme animé avec le package␣
 ↪bar_chart_race, montrant le pourcentage de thèses soutenues au fil des ans␣
 ↪en fonction de la discipline ; seules les 6 disciplines dont le pourcentage␣
 ↪de thèses soutenues est le plus élevé sont représentées par année ; le␣
 ↪graphique est sauvegardé sous format mp4, pour une conversion ultérieure en␣
 ↪gif
```

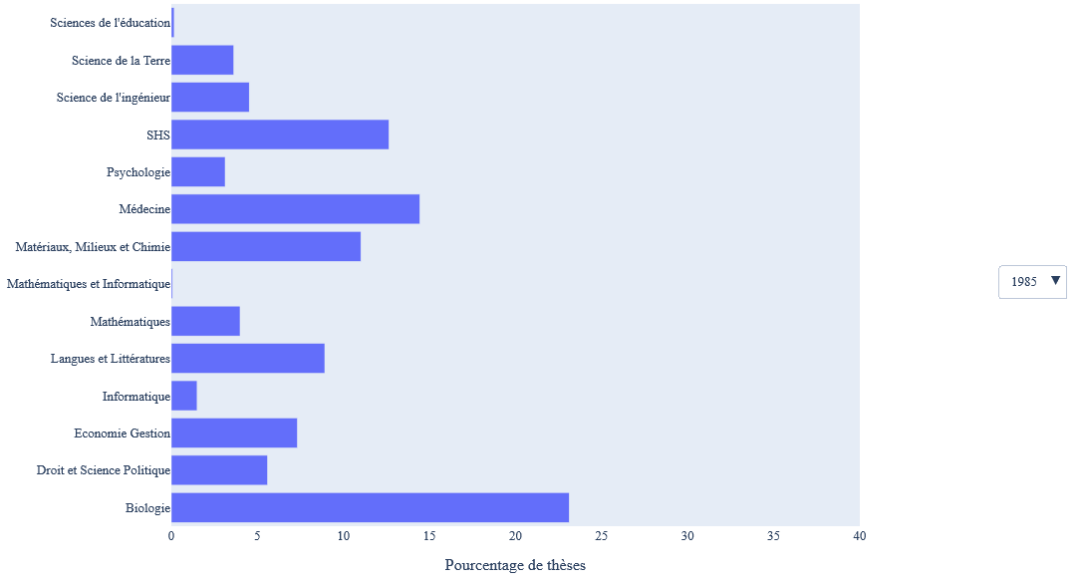
```
bcr.bar_chart_race(PhD_v3_discipline_ang, r"C:\Users\quent\Documents\DU Data␣
↪Analyst 2022\UE n°2 - Visualisation de données\Devoirs\bar_chart_race.mp4",␣
↪n_bars = 6, fixed_order = False, fixed_max = True, steps_per_period = 20,␣
↪period_length = 1000, figsize = (12/2.54, 8/2.54), dpi = 300, title =␣
↪"Pourcentage de thèses rédigées en anglais par discipline", shared_fontdict␣
↪= {"family" : "Times New Roman", "weight" : "normal", "size" : 12},␣
↪bar_label_size = 6, tick_label_size = 6, title_size = 10, period_fmt = "%Y")
```

2.2 Exercice 10

[31]:
```
# (1) Utilisation d'un slider pour afficher différents histogrammes en␣
↪fonction du temps, chaque histrogramme représentant le pourcentage de␣
↪thèses soutenues par discipline sur une année donnée
List_discipline = list(PhD_v3_discipline_annee["Discipline_predi"])
nb_tot_theses = []
for i in PhD_v3_discipline_annee.columns[1:]:
    nb_tot_theses.append(np.sum(PhD_v3_discipline_annee[i].values))

fig = go.Figure()
for i in range(1985,2019):
    fig.add_trace(go.Bar(x = PhD_v3_prop_discipline_annee[i], y =␣
↪List_discipline, name = i, orientation = "h"))
for i in range(1,len(range(1985,2019))):
    fig.data[i].visible = False
my_steps = []
for i in range(1985,2019):
    my_list = [False] * len(range(1985,2019))
    my_list[i-1985] = True
    my_steps.append({"label": str(i), "method": "update", "args": [{"visible":
↪ my_list}, {"title": {"text": "N = " + str(nb_tot_theses[i-1985]), "font":␣
↪{"family": "Times New Roman", "size": 13}, "x": 0.945, "y": 0.95}}]})
sliders = [{"steps": my_steps}]
fig.update_layout({"xaxis": {"range": [0, 40], "title": {"text": "Pourcentage␣
↪de thèses"}}})
fig.update_layout({"sliders": sliders}, margin = dict(l = 20, r = 20, t = 20,␣
↪b = 20), width = 550, height = 600, font = {"family": "Times New Roman",␣
↪"size": 13})
fig["layout"]["sliders"][0]["pad"] = dict(t = 40)
fig.show()
fig.write_image("Figure_2.pdf")
fig.write_html("Figure_2.html")
```



[32]:
```
# (2) Même représentation mais en remplaçant le slider par un bouton␣
 ↪"selector"
fig = go.Figure()
for i in range(1985,2019):
    fig.add_trace(go.Bar(x = PhD_v3_prop_discipline_annee[i], y =␣
 ↪List_discipline, name = i, orientation = "h"))
for i in range(1,len(range(1985,2019))):
    fig.data[i].visible = False
dropdown_buttons = []
for i in range(1985,2019):
    my_list = [False] * len(range(1985,2019))
    my_list[i-1985] = True
    dropdown_buttons.append({"label": str(i), "method": "update", "args":␣
 ↪[{"visible": my_list}, {"title": {"text": "N = " +␣
 ↪str(nb_tot_theses[i-1985]), "font": {"family": "Times New Roman", "size":␣
 ↪13}, "x": 0.81, "y": 0.95}}]})
fig.update_layout({'updatemenus': [{'type': "dropdown", 'direction': "down",␣
 ↪'x': 1.3, 'y': 0.5, 'showactive': True, 'active': 0, 'buttons':␣
 ↪dropdown_buttons}]})
fig.update_layout({"xaxis": {"range": [0, 40], "title": {"text": "Pourcentage␣
 ↪de thèses"}}})
fig.update_layout(margin = dict(l = 20, r = 20, t = 20, b = 20), width = 550,␣
 ↪height = 600, font = {"family": "Times New Roman", "size": 13})
fig.write_html("Figure_2B.html")
fig.show()
```



3 Visualisation de données spatialisées

3.1 Exercice 11

[33]:
```
# (1) Import du jeu de données "df_russia_2022_final"
df_russia_2022_final = pd.read_csv(r"C:\Users\quent\Documents\DU Data Analyst␣
↪2022\UE n°2 - Visualisation de données\Jeux de données\df_russia_2022_final.
↪csv", low_memory = False, encoding='utf-8')
df_russia_2022_final.head()
```

[33]:

	callsign	number	icao24	registration	typecode	origin	destination	\
0	AZS4001	NaN	152ad6	RA-76502	IL76	LTAC	BIKF	
1	SDM6453	NaN	155c3b	NaN	NaN	ULLI	NaN	
2	LLM90	YC90	155c01	NaN	NaN	UUDD	NaN	
3	RWZ1284	NaN	155c6b	NaN	NaN	NaN	NaN	
4	IAE410	NaN	155bca	RA-89034	SU95	UIII	NaN	

	firstseen	lastseen	\
0	2022-01-31 21:35:37+00:00	2022-02-01 05:17:56+00:00	
1	2022-01-31 22:13:08+00:00	2022-02-01 00:14:18+00:00	
2	2022-01-31 22:59:23+00:00	2022-02-01 01:08:41+00:00	
3	2022-01-31 23:30:57+00:00	2022-02-01 00:03:54+00:00	
4	2022-01-31 23:38:18+00:00	2022-02-01 00:02:13+00:00	

	day	latitude_1	longitude_1	altitude_1	\
0	2022-02-01 00:00:00+00:00	40.159359	33.023155	914.4	
1	2022-02-01 00:00:00+00:00	59.793320	30.262299	304.8	
2	2022-02-01 00:00:00+00:00	55.365280	37.971497	1219.2	
3	2022-02-01 00:00:00+00:00	57.897577	66.597198	9753.6	
4	2022-02-01 00:00:00+00:00	52.265315	104.401664	304.8	

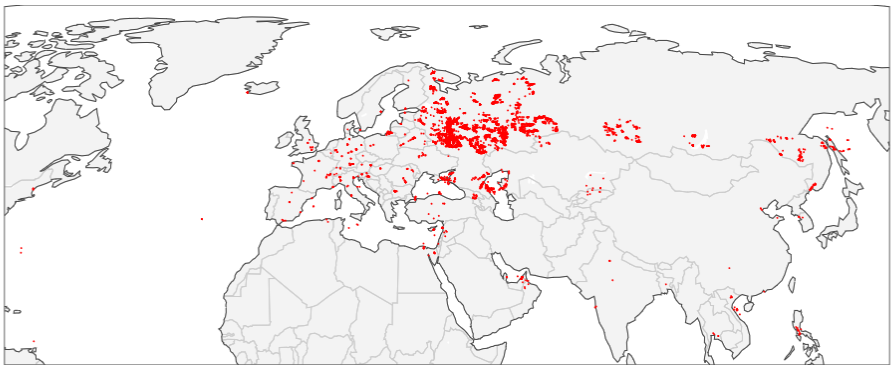
```
   latitude_2  longitude_2  altitude_2
0   63.985248   -22.635654         NaN
1   53.155151    52.842904    10668.00
2   64.820892    61.743823    10668.00
3   55.501347    62.074900     2560.32
4   54.335586   105.956310    10668.00
```

[34]:
```
# (2) Identification de la nature des variables
df_russia_2022_final.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7369 entries, 0 to 7368
Data columns (total 16 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   callsign       7369 non-null   object 
 1   number         628 non-null    object 
 2   icao24         7369 non-null   object 
 3   registration   2842 non-null   object 
 4   typecode       2842 non-null   object 
 5   origin         3556 non-null   object 
 6   destination    3958 non-null   object 
 7   firstseen      7369 non-null   object 
 8   lastseen       7369 non-null   object 
 9   day            7369 non-null   object 
 10  latitude_1     7369 non-null   float64
 11  longitude_1    7369 non-null   float64
 12  altitude_1     7369 non-null   float64
 13  latitude_2     7369 non-null   float64
 14  longitude_2    7369 non-null   float64
 15  altitude_2     7352 non-null   float64
dtypes: float64(6), object(10)
memory usage: 921.2+ KB
```

[35]:
```
# (3) Affichage d'une carte du monde (projection de Robinson)
fig = go.Figure(go.Scattergeo())
fig.update_layout(height = 300, margin = {"r":0,"t":0,"l":0,"b":0}, geo =␣
↪dict(showland = True, showcountries = True, landcolor = "rgb(243, 243,␣
↪243)", countrycolor = "rgb(204, 204, 204)", projection_type = "natural␣
↪earth"))
fig.show()
```

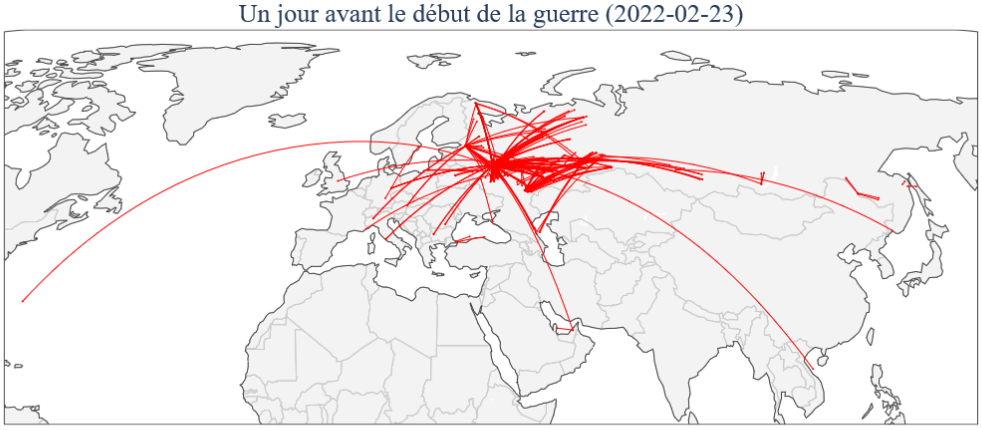
[36]:
```
# (4) Affichage des coordonnées de départ et d'arrivée des vols toute date␣
↪confondue, en centrant la carte sur Moscou
df_russia_2022_final["longitude_1"] = df_russia_2022_final["longitude_1 "]
fig = go.Figure()
fig.add_trace(go.Scattergeo(lon = df_russia_2022_final["longitude_1"], lat =␣
↪df_russia_2022_final["latitude_1"], mode = 'markers', marker = dict(size =␣
↪2, color = 'rgb(255, 0, 0)', line = dict(width = 3, color = 'rgba(68, 68,␣
↪68, 0)'))))
fig.add_trace(go.Scattergeo(lon = df_russia_2022_final["longitude_2"], lat =␣
↪df_russia_2022_final["latitude_2"], mode = 'markers', marker = dict(size =␣
↪2, color = 'rgb(255, 0, 0)', line = dict(width = 3, color = 'rgba(68, 68,␣
↪68, 0)'))))
fig.update_layout(height = 300, margin = {"r":0,"t":0,"l":0,"b":0},␣
↪showlegend = False, geo = dict(showland = True, showcountries = True,␣
↪landcolor = "rgb(243, 243, 243)", countrycolor = "rgb(204, 204, 204)",␣
↪projection_type = "natural earth", lataxis_range = [55-50, 55+28],␣
↪lonaxis_range = [37-105, 37+105]))
fig.show()
```



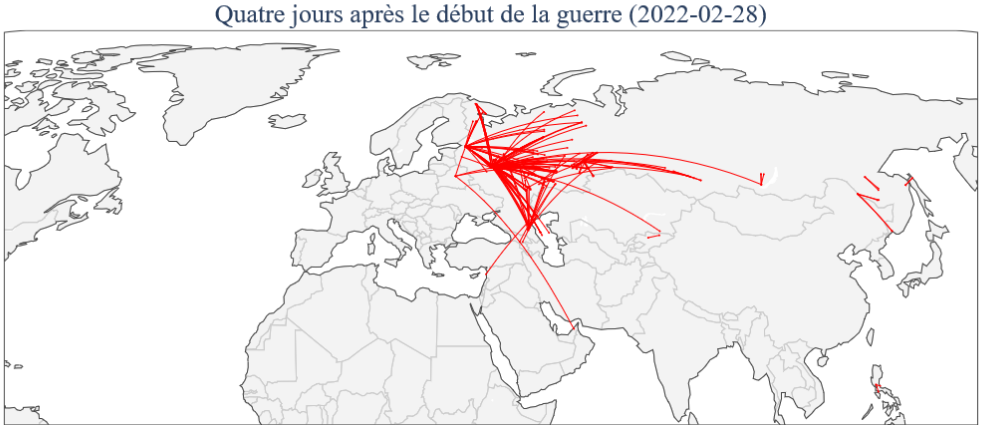
[37]:
```
# (5) Affichage des vols le 23/02/2022 et le 28/02/2022, i.e. 1 jour avant et␣
↪4 jours après le début de la guerre en Ukraine
df_russia_2022_final["firstseen"] = pd.
↪to_datetime(df_russia_2022_final["firstseen"], infer_datetime_format = True)
df_russia_2022_final["firstseen_day"] = df_russia_2022_final["firstseen"].dt.
↪strftime("%Y-%m-%d")
days = df_russia_2022_final["firstseen_day"].unique()
days_23_28 = [days[23], days[28]]
fig = make_subplots(rows = 2, cols = 1, specs = [[{"type": "scattergeo"}],␣
↪[{"type": "scattergeo"}]], subplot_titles = ["Un jour avant le début de la␣
↪guerre (2022-02-23)", "Quatre jours après le début de la guerre␣
↪(2022-02-28)"], vertical_spacing = 0.09)
for i in range(2):
```

```
    fig.add_trace(go.Scattergeo(lon =␣
↪df_russia_2022_final[df_russia_2022_final["firstseen_day"] ==␣
↪days_23_28[i]]["longitude_1"], lat =␣
↪df_russia_2022_final[df_russia_2022_final["firstseen_day"] ==␣
↪days_23_28[i]]["latitude_1"], mode = 'markers', marker = dict(size = 2,␣
↪color = 'rgb(255, 0, 0)', line = dict(width = 3, color = 'rgba(68, 68, 68,␣
↪0)'))), row = 1+i, col = 1)
    fig.add_trace(go.Scattergeo(lon =␣
↪df_russia_2022_final[df_russia_2022_final["firstseen_day"] ==␣
↪days_23_28[i]]["longitude_2"], lat =␣
↪df_russia_2022_final[df_russia_2022_final["firstseen_day"] ==␣
↪days_23_28[i]]["latitude_2"], mode = 'markers', marker = dict(size = 2,␣
↪color = 'rgb(255, 0, 0)', line = dict(width = 3, color = 'rgba(68, 68, 68,␣
↪0)'))), row = 1+i, col = 1)
    for j in list(df_russia_2022_final[df_russia_2022_final["firstseen_day"]␣
↪== days_23_28[i]].index):
        fig.add_trace(go.Scattergeo(lon =␣
↪[df_russia_2022_final[df_russia_2022_final["firstseen_day"] ==␣
↪days_23_28[i]]["longitude_1"][j],␣
↪df_russia_2022_final[df_russia_2022_final["firstseen_day"] ==␣
↪days_23_28[i]]["longitude_2"][j]], lat =␣
↪[df_russia_2022_final[df_russia_2022_final["firstseen_day"] ==␣
↪days_23_28[i]]["latitude_1"][j],␣
↪df_russia_2022_final[df_russia_2022_final["firstseen_day"] ==␣
↪days_23_28[i]]["latitude_2"][j]], mode = 'lines', line = dict(width = 1,␣
↪color = 'red')), row = 1+i, col = 1)
fig.update_layout(height = 500, margin = {"r":0,"t":40,"l":0,"b":10},␣
↪showlegend = False, font = {"family": "Times New Roman", "size": 12})
fig.update_geos(showland = True, showcountries = True, landcolor = "rgb(243,␣
↪243, 243)", countrycolor = "rgb(204, 204, 204)", projection_type = "natural␣
↪earth", lataxis_range = [55-50, 55+28], lonaxis_range = [37-105, 37+105])
fig.show()
fig.write_image("Figure_3.pdf")
```

Un jour avant le début de la guerre (2022-02-23)



Quatre jours après le début de la guerre (2022-02-28)



3.2 Exercice 12

[38]:
```
# (1) Export des données en format csv pour une visualisation sur le site␣
 ↪kepler.gl
df_russia_2022_final_23 =␣
 ↪df_russia_2022_final[df_russia_2022_final["firstseen_day"] == days_23_28[0]]
df_russia_2022_final_23.to_csv('df_russia_2022_final_23.csv')
df_russia_2022_final_28 =␣
 ↪df_russia_2022_final[df_russia_2022_final["firstseen_day"] == days_23_28[1]]
df_russia_2022_final_28.to_csv('df_russia_2022_final_28.csv')
```