

Notebook - UE Rédaction de rapport

Quentin Fouché

1 septembre 2022

1 Description du jeu de données

```
[1]: # (1) Import des packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches
import seaborn as sns
import itertools
from statsmodels.formula.api import ols
import sys
!{sys.executable} -m pip install missingno
import missingno as msno
!{sys.executable} -m pip install upsetplot
from upsetplot import plot, from_indicators
import statsmodels.api as sm
from scipy import stats
import scikit_posthocs as sp

[2]: # (2) Import du jeu de données "Biblio_synthese.csv"
biblio_orig = pd.read_csv(r"C:\Users\quent\Documents\DU Data Analyst 2022\UE_
    ↳n°4 - Rédaction d'un rapport d'analyse\Jeux de données\Biblio_synthese.
    ↳csv", low_memory = False, encoding = "latin-1")
biblio = biblio_orig.copy(deep = True)

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

[4]: # (4) Modification du titre des colonnes
biblio = biblio.rename(columns = {"section": "discipline", "numÃ©ro TEL":
    ↳"numéro TEL", "numÃ©ro citation": "numéro citation"})

[5]: # (5) Renommage des types de référence
```

```

biblio["type"] = biblio["type"].replace(["chapter", "book",
    ↳"article-journal", "paper-conference", "thesis", "report", "webpage",
    ↳"manuscript", "patent", "interview", "motion_picture",
    ↳"personal_communication"], ["chapitre", "livre", "article", "papier de
    ↳conférence", "thèse", "rapport", "page web", "manuscrit", "brevet",
    ↳"interview", "film", "communication personnelle"], regex = True)
biblio.head()

```

```

[5]:  discipline      numéro TEL  numéro citation      type \
0      I  halshs-00005971v1          1  chapitre
1      I  halshs-00005971v1          1    livre
2      I  halshs-00005971v1          1      NaN
3      I  halshs-00005971v1          1  article
4      I  halshs-00005971v1          1    livre

```

```

                                titre
0                                Action
1  Mainstreaming Gender in the European Structura...
2  L'Européanisation saisie par son instrumenta...
3  Eléments pour une sociologie de la traduction Â»
4  [' L'agonie d'un laboratoire', 'La science...

```

```

[6]: # (6) Inspection des variables
print(biblio.info())
print("")
print("-----")
print("")
print("Nombre de valeurs uniques par colonnes:")
print(biblio.nunique())
print("")
print("-----")
print("")
print("Disciplines:")
print(biblio["discipline"].unique())
print("")
print("-----")
print("")
print("Types de publication:")
print(biblio["type"].unique())

```

```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1821132 entries, 0 to 1821131
Data columns (total 5 columns):
#   Column      Dtype
---  -
0   discipline  object
1   numéro TEL  object
2   numéro citation  int64
3   type        object
4   titre       object
dtypes: int64(1), object(4)

```

memory usage: 69.5+ MB
None

Nombre de valeurs uniques par colonnes:

discipline	11
numéro TEL	22889
numéro citation	1
type	12
titre	1437703

dtype: int64

Disciplines:

['I' 'II' 'III' 'IV' 'IX' 'pharmacie' 'V' 'VI' 'VII' 'VIII' 'X']

Types de publication:

['chapitre' 'livre' nan 'article' 'papier de conférence' 'thèse' 'rapport'
'page web' 'manuscrit' 'brevet' 'interview' 'film'
'communication personnelle']

2 Étude des données manquantes

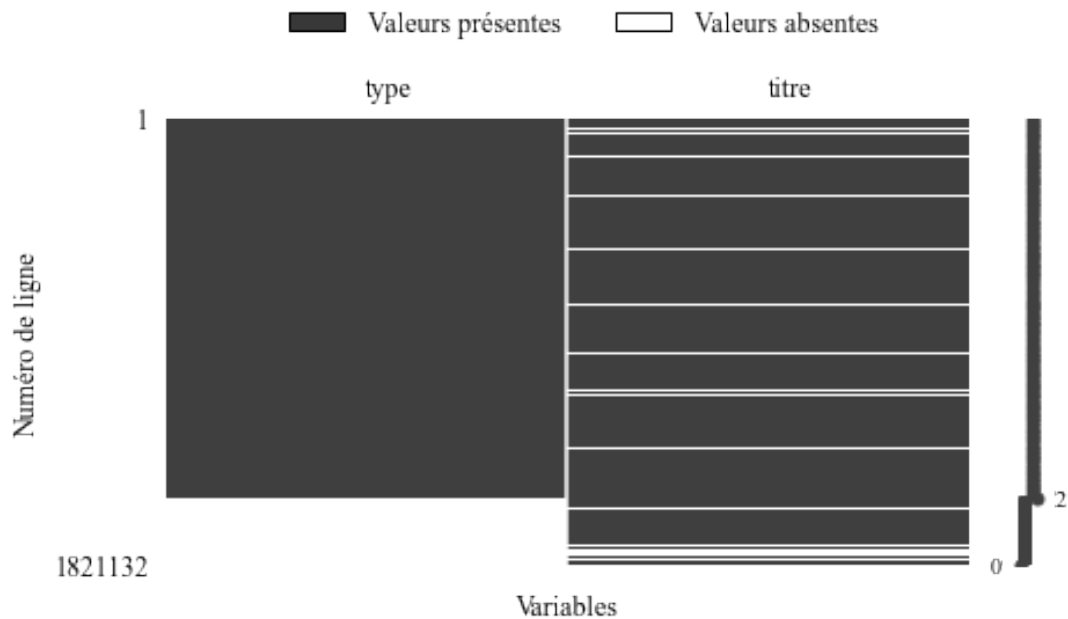
2.1 Répartition des données manquantes

```
[7]: # (1) Calcul du pourcentage de valeurs manquantes par variable
biblio.isna().mean()*100
```

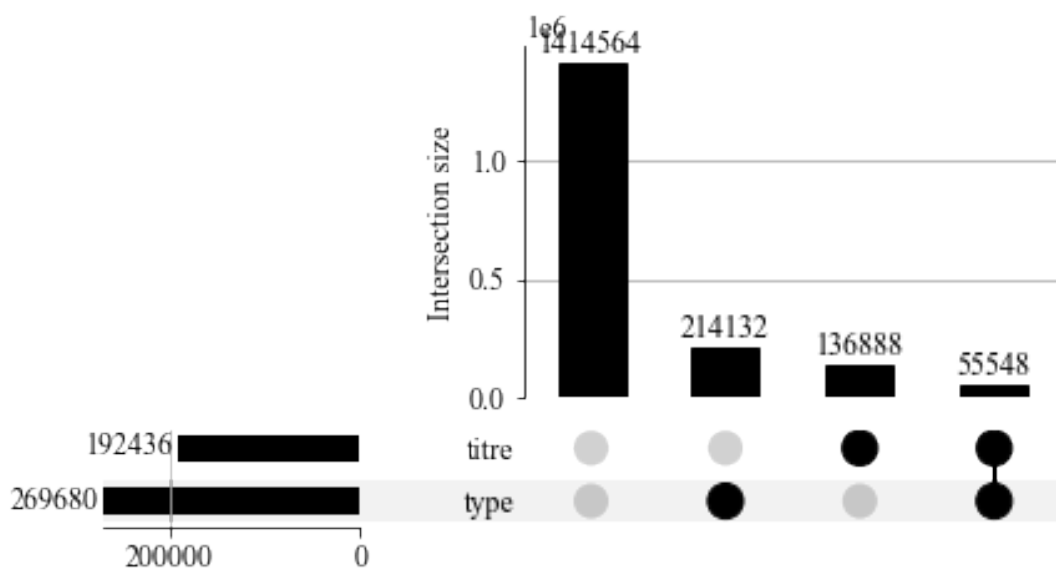
```
[7]: discipline          0.000000
numéro TEL            0.000000
numéro citation       0.000000
type                 14.808372
titre                10.566834
dtype: float64
```

```
[8]: # (2) Représentation de la distribution des valeurs manquantes dans les deux
      ↪ variables qui en contiennent
plt.rc("font", family = "Times New Roman", size = 12)
msno.matrix(biblio[["type", "titre"]].sort_values("type"), figsize = (17.5/2.
      ↪ 54, 9/2.54), fontsize = 12, label_rotation = 0)
plt.ylabel("Numéro de ligne", fontsize = 12)
plt.xlabel("Variables", fontsize = 12, labelpad = 12)
plt.tick_params(axis = "both", labelsize = 12)
gray_patch = mpatches.Patch(facecolor = '#383838', edgecolor = 'black', label_
      ↪ = 'Valeurs présentes')
white_patch = mpatches.Patch(facecolor = 'white', edgecolor = 'black', label_
      ↪ = 'Valeurs absentes')
```

```
plt.legend(handles = [gray_patch, white_patch], fontsize = 12, bbox_to_anchor=(
    0.92, 1.30), frameon = False, ncol = 2)
plt.show()
```



```
[9]: # (3) Représentation de la distribution des valeurs manquantes dans chaque
    ↪ combinaison de variables
plt.rc('font', family = 'Times New Roman', size = 12)
fig = plt.figure(figsize = (17.5/2.54, 9/2.54))
plot(from_indicators(indicators = pd.isna, data = biblio[["type", "titre"]]),
    ↪ show_counts = True, fig = fig, element_size = None)
plt.show()
```



```
[10]: # (4) Calcul du pourcentage de valeurs manquantes dans chaque combinaison des
      ↪deux variables qui en contiennent
print("Pourcentage de valeurs manquantes dans :")
print("    - le titre des références : " + str(round(214132 * 100 / (214132 +
      ↪136888 + 55548)))) + "%")
print("    - le type des références : " + str(round(136888 * 100 / (214132 +
      ↪136888 + 55548)))) + "%")
print("    - le titre et le type : " + str(round(55548 * 100 / (214132 +
      ↪136888 + 55548)))) + "%")
```

Pourcentage de valeurs manquantes dans :

- le titre des références : 53%
- le type des références : 34%
- le titre et le type : 14%

2.2 Corrélations entre les données manquantes

```
[11]: # (1) Représentation de la proportion de données manquantes communes par
      ↪paire de variables
plt.rc('font', family = 'Times New Roman', size = 12)
msno.heatmap(biblio, figsize = (17.5/2.54, 9/2.54), fontsize = 12,
      ↪label_rotation = 0)
plt.show()
```



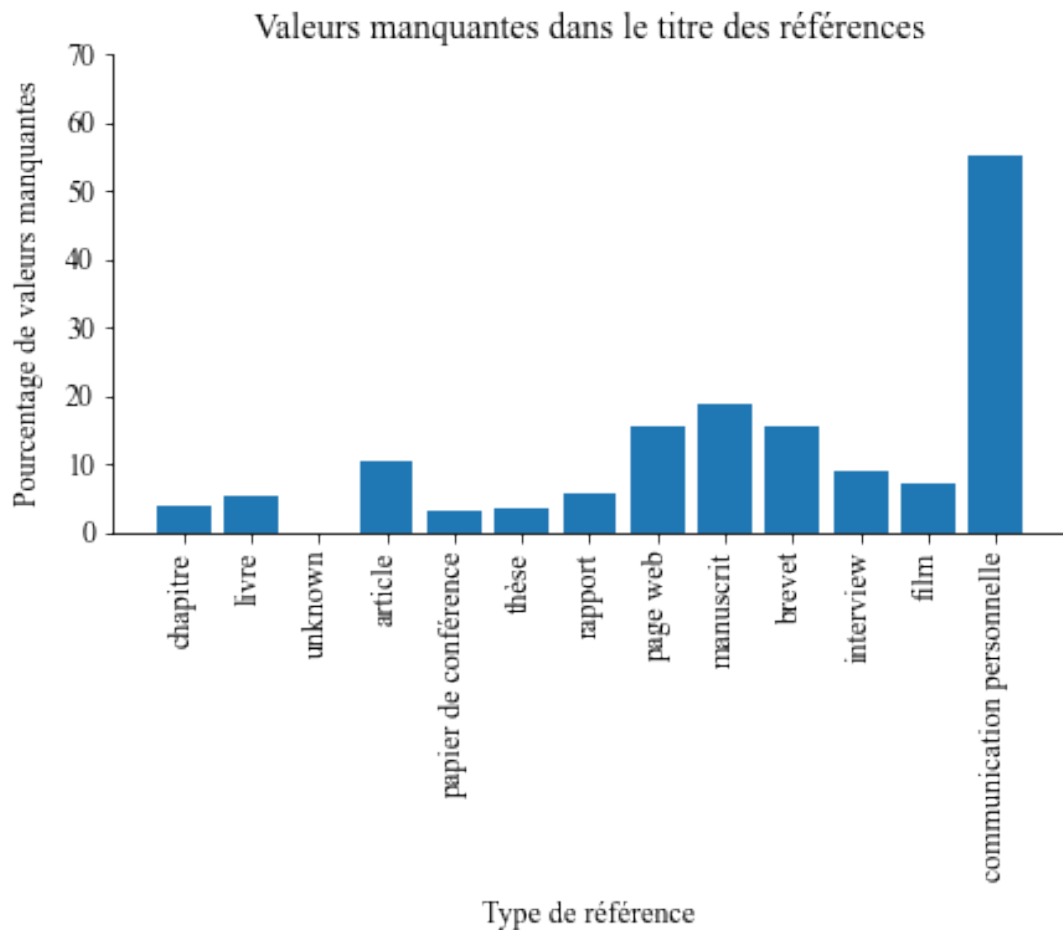
```
[12]: # (2) Relation entre le type d'une référence et la probabilité que son titre
      ↪soit manquant
prop_nan_titre_par_type = []
for i in biblio["type"].unique():
```

```

prop_nan_titre_par_type.append((biblio[biblio["type"] == i].isna().
↳mean()*100)["titre"])
list_type_ref = biblio["type"].unique()
list_type_ref[2] = "unknown"

plt.rc("font", family = "Times New Roman", size = 12)
fig, ax = plt.subplots(figsize = (17.5/2.54, 9/2.54))
ax.bar(x = list_type_ref, height = prop_nan_titre_par_type)
ax.spines[["right", "top"]].set_visible(False)
ax.set_title("Valeurs manquantes dans le titre des références")
ax.set_xlabel("Type de référence", labelpad = 8)
ax.tick_params(axis = "x", rotation = 90)
ax.set_ylabel("Pourcentage de valeurs manquantes", labelpad = 8)
ax.set_ylim(0,70)
plt.show()

```



```

[13]: # (3) Relation entre la discipline d'une thèse et la probabilité que le titre_
↳des références citées soit manquant
prop_nan_titre_par_discipline = []
for i in biblio["discipline"].unique():

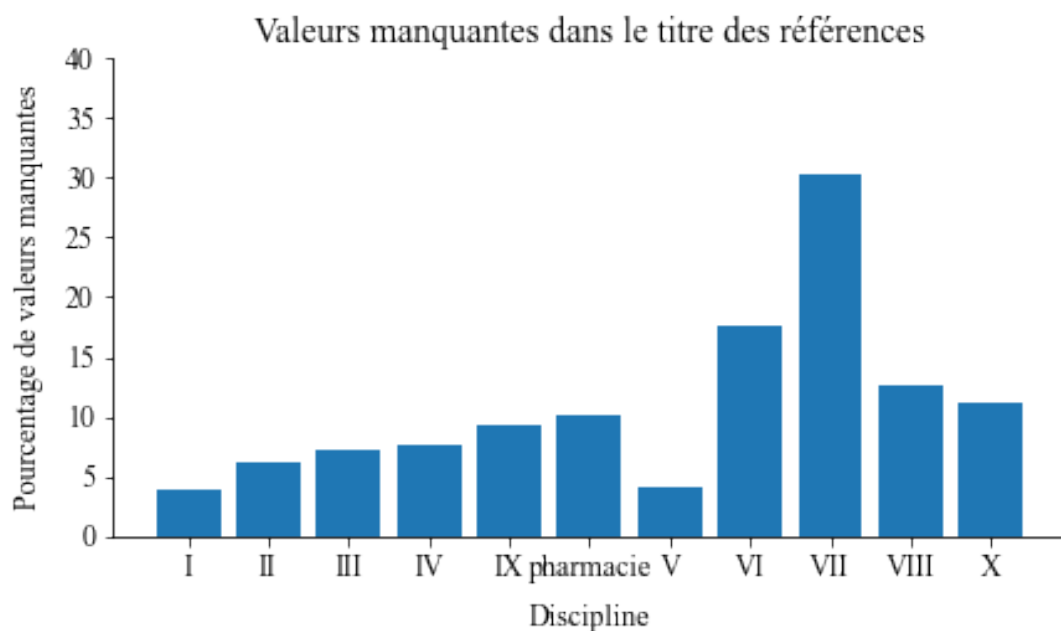
```

```

prop_nan_titre_par_discipline.append((biblio[biblio["discipline"] == i].
↳isna().mean()*100)["titre"])

plt.rc("font", family = "Times New Roman", size = 12)
fig, ax = plt.subplots(figsize = (17.5/2.54, 9/2.54))
ax.bar(x = list(biblio["discipline"].unique()), height =
↳prop_nan_titre_par_discipline)
ax.spines[["right", "top"]].set_visible(False)
ax.set_title("Valeurs manquantes dans le titre des références")
ax.set_xlabel("Discipline", labelpad = 8)
ax.set_ylabel("Pourcentage de valeurs manquantes", labelpad = 8)
ax.set_ylim(0,40)
plt.show()

```



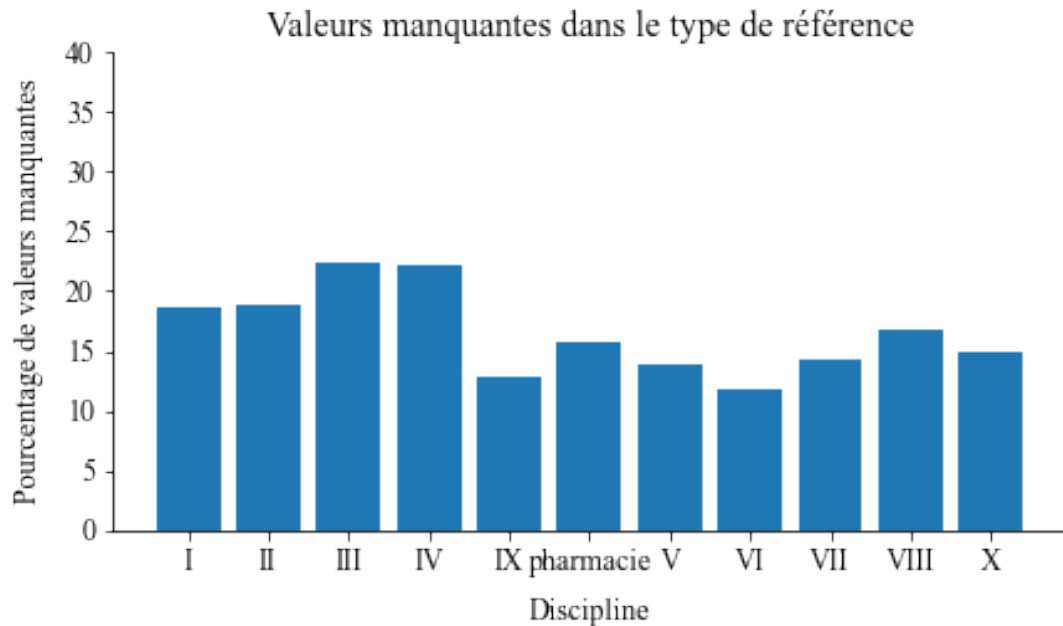
```

[14]: # (4) Relation entre la discipline d'une thèse et la probabilité que le type
↳de référence cité soit manquant
prop_nan_type_par_discipline = []
for i in biblio["discipline"].unique():
    prop_nan_type_par_discipline.append((biblio[biblio["discipline"] == i].
↳isna().mean()*100)["type"])

plt.rc("font", family = "Times New Roman", size = 12)
fig, ax = plt.subplots(figsize = (17.5/2.54, 9/2.54))
ax.bar(x = list(biblio["discipline"].unique()), height =
↳prop_nan_type_par_discipline)
ax.spines[["right", "top"]].set_visible(False)
ax.set_title("Valeurs manquantes dans le type de référence")
ax.set_xlabel("Discipline", labelpad = 8)
ax.set_ylabel("Pourcentage de valeurs manquantes", labelpad = 8)

```

```
ax.set_ylim(0,40)
plt.show()
```



- ```
[15]: # (5) Calcul du pourcentage de valeurs manquantes par thèse dans le titre et
 → dans le type des références
prop_nan_titre_par_these = []
for i in biblio["numéro TEL"].unique():
 prop_nan_titre_par_these.append((biblio[biblio["numéro TEL"] == i].isna().
 → mean()*100)["titre"])
prop_nan_type_par_these = []
for i in biblio["numéro TEL"].unique():
 prop_nan_type_par_these.append((biblio[biblio["numéro TEL"] == i].isna().
 → mean()*100)["type"])

[16]: # (6) Stockage des données de pourcentage de valeurs manquantes par thèse
 → dans le titre et le type de référence dans un tableau à part
Tableau_nan_par_these = pd.DataFrame({"Valeurs_manquantes_dans_titre_ref" :
 → prop_nan_titre_par_these, "Valeurs_manquantes_dans_type_ref" :
 → prop_nan_type_par_these}, index = list(biblio["numéro TEL"].unique()))

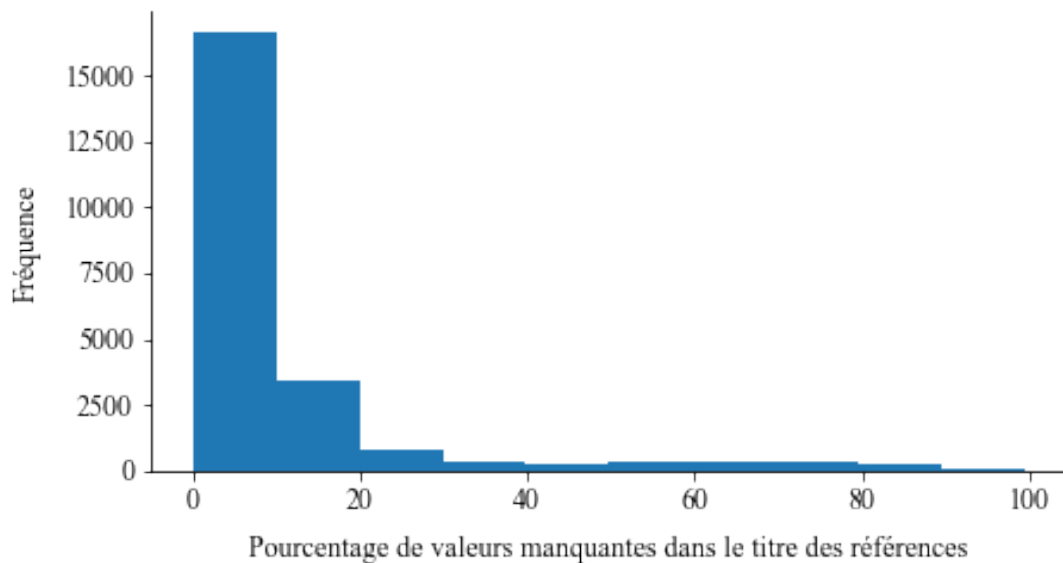
[17]: # (7) Copie du tableau dans le presse-papier pour le transférer dans Excel
Tableau_nan_par_these.to_clipboard(excel = True)

[18]: # (8) Export du tableau en format csv
Tableau_nan_par_these.to_csv("table.exportcsv.csv")

[19]: # (9) Relation entre une thèse et la probabilité que le titre des références
 → citées dans cette thèse soit manquant
```

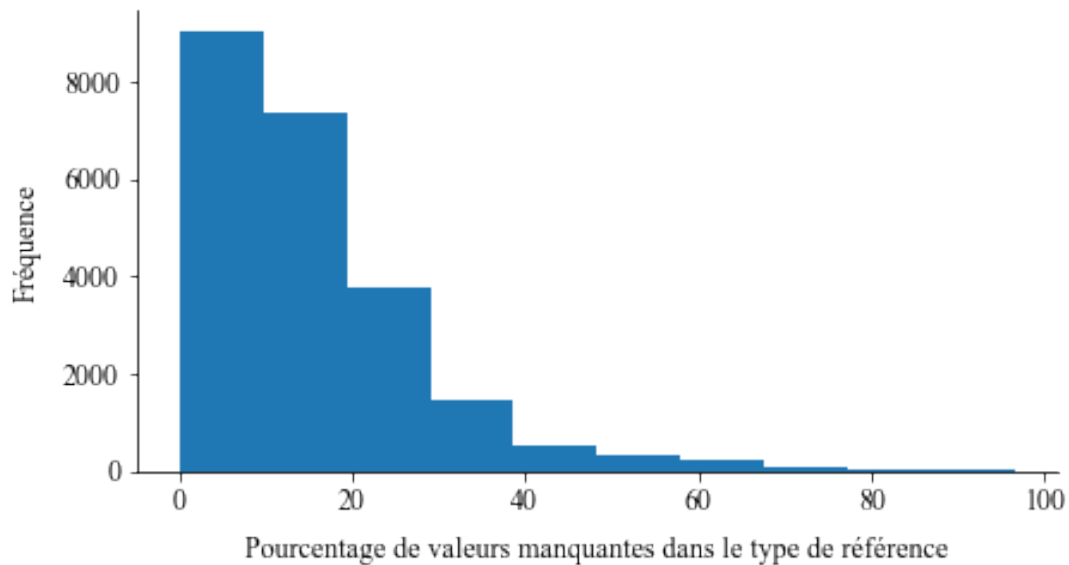


```
Tableau_nan_par_these = pd.read_csv(r"C:\Users\quent\Documents\DU Data_
↳Analyst 2022\UE n°4 - Rédaction d'un rapport d'analyse\Devoirs\table.
↳exportcsv.csv", low_memory = False, encoding = "latin-1")
plt.rc("font", family = "Times New Roman", size = 12)
fig, ax = plt.subplots(figsize = (17.5/2.54, 9/2.54))
ax.hist(Tableau_nan_par_these["Valeurs_manquantes_dans_titre_ref"], bins = 10)
ax.spines[["right", "top"]].set_visible(False)
ax.set_xlabel("Pourcentage de valeurs manquantes dans le titre des_
↳références", labelpad = 8)
ax.set_ylabel("Fréquence", labelpad = 8)
plt.show()
```



[20]: *# (10) Relation entre une thèse et la probabilité que le type de référence\_*  
*↳cité dans cette thèse soit manquant*

```
plt.rc("font", family = "Times New Roman", size = 12)
fig, ax = plt.subplots(figsize = (17.5/2.54, 9/2.54))
ax.hist(Tableau_nan_par_these["Valeurs_manquantes_dans_type_ref"], bins = 10)
ax.spines[["right", "top"]].set_visible(False)
ax.set_xlabel("Pourcentage de valeurs manquantes dans le type de référence",_
↳labelpad = 8)
ax.set_ylabel("Fréquence", labelpad = 8)
plt.show()
```



### 3 Nombre de références citées par thèse et discipline

```
[21]: # (1) Calcul du nombre total de thèses présentes dans le jeu de données par
 ↳ discipline
Nb_these_par_discipline = biblio.groupby(["discipline", "numéro TEL"]).size().
 ↳ to_frame().reset_index().groupby("discipline").size()
print(Nb_these_par_discipline)
```

```
discipline
I 229
II 1359
III 654
IV 1003
IX 9430
V 2851
VI 2768
VII 1235
VIII 1551
X 1780
pharmacie 29
dtype: int64
```

```
[22]: # (2) Retrait des thèses réalisées en pharmacie
biblio_sans_pharma = biblio[biblio["discipline"] != "pharmacie"]
```

```
[23]: # (3) Calcul de la moyenne et de l'erreur standard du nombre de références
 ↳ par thèse pour chaque discipline
Nb_ref_par_these_discipline = biblio_sans_pharma.pivot_table(values = "numéro
 ↳ citation", index = "numéro TEL", columns = "discipline", aggfunc = "sum")
my_df1 = Nb_ref_par_these_discipline.copy(deep = True)
my_df2 = pd.DataFrame()
```

```

my_df2.index = ["Moyenne", "ESM"]
for i in my_df1.columns:
 my_df2[i] = [my_df1[i].values[~np.isnan(my_df1[i].values)].mean(),
 ↪ my_df1[i].values[~np.isnan(my_df1[i].values)].std() / np.sqrt(len(my_df1[i].
 ↪ values[~np.isnan(my_df1[i].values)))]
Moyenne_ESM_nb_ref_par_these_discipline = my_df2.copy(deep = True)
Moyenne_ESM_nb_ref_par_these_discipline

```

[23]:

|         | I          | II         | III        | IV         | IX        | V         |
|---------|------------|------------|------------|------------|-----------|-----------|
| Moyenne | 105.170306 | 108.726269 | 105.068807 | 104.039880 | 74.667232 | 67.926692 |
| ESM     | 4.800131   | 1.690197   | 2.332825   | 2.149698   | 0.523056  | 0.851124  |

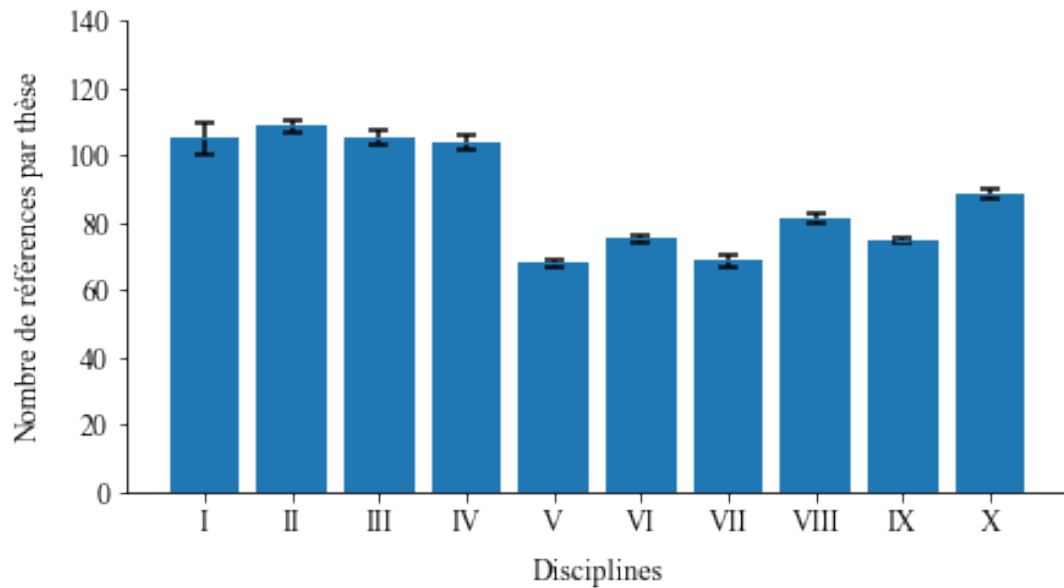
|         | VI        | VII       | VIII      | X         |
|---------|-----------|-----------|-----------|-----------|
| Moyenne | 75.084899 | 68.682591 | 81.056738 | 88.670225 |
| ESM     | 0.945796  | 1.847050  | 1.566584  | 1.626058  |

[24]:

```

(4) Représentation en barplot
plt.rc("font", family = "Times New Roman", size = 12)
fig, ax = plt.subplots(figsize = (17.5/2.54, 9/2.54))
ax.bar(x = Moyenne_ESM_nb_ref_par_these_discipline.
 ↪ columns[[0,1,2,3,5,6,7,8,4,9]], height =
 ↪ Moyenne_ESM_nb_ref_par_these_discipline.loc["Moyenne"].
 ↪ values[[0,1,2,3,5,6,7,8,4,9]])
ax.errorbar(x = Moyenne_ESM_nb_ref_par_these_discipline.
 ↪ columns[[0,1,2,3,5,6,7,8,4,9]], y = Moyenne_ESM_nb_ref_par_these_discipline.
 ↪ loc["Moyenne"].values[[0,1,2,3,5,6,7,8,4,9]], yerr =
 ↪ Moyenne_ESM_nb_ref_par_these_discipline.loc["ESM"].
 ↪ values[[0,1,2,3,5,6,7,8,4,9]], fmt='_', capsize = 4, capthick = 1.5, ecolor
 ↪ = "black")
ax.spines[["right", "top"]].set_visible(False)
ax.set_xlabel("Disciplines", labelpad = 8)
ax.set_ylabel("Nombre de références par thèse", labelpad = 8)
ax.set_ylim(0,140)
plt.show()

```



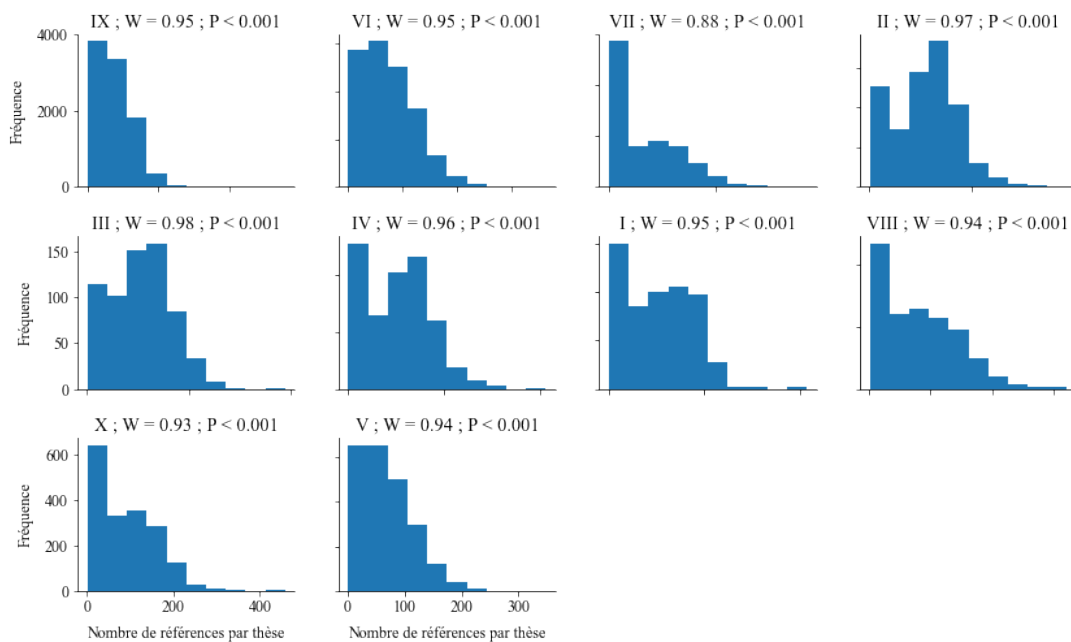
```
[25]: # (5) Test de la normalité des données pour chaque discipline (représentation
 ↳ des distributions en affichant le résultat des tests dans le titre des
 ↳ graphiques)
Nb_ref_avec_discipline = Nb_ref_par_these_discipline.stack().reset_index()
Nb_ref_avec_discipline = Nb_ref_avec_discipline.rename(columns = {"numéro_
 ↳ TEL": "these"})
Nb_ref_avec_discipline["nb_ref"] =
 ↳ Nb_ref_avec_discipline[Nb_ref_avec_discipline.columns[2]]
Nb_ref_avec_discipline = Nb_ref_avec_discipline.drop(columns =
 ↳ [Nb_ref_avec_discipline.columns[2]])

plt.rc("font", family = "Times New Roman", size = 12)
fig, ax = plt.subplots(nrows = 3, ncols = 4, figsize = (35/2.54, 20/2.54))
row = 0
col = 0
for i in Nb_ref_avec_discipline["discipline"].unique():
 test_shapiro = stats.
 ↳ shapiro(Nb_ref_avec_discipline[Nb_ref_avec_discipline["discipline"] ==
 ↳ i]["nb_ref"])
 ax[row,col].
 ↳ hist(Nb_ref_avec_discipline[Nb_ref_avec_discipline["discipline"] ==
 ↳ i]["nb_ref"], bins = 10)
 if test_shapiro[1] < 0.001:
 ax[row,col].set_title(i + " ; " + "W = " + str(round(test_shapiro[0],
 ↳ 2)) + " ; P < " + str(0.001))
 else:
 ax[row,col].set_title(i + " ; " + "W = " + str(round(test_shapiro[0],
 ↳ 2)) + " ; P = " + str(round(test_shapiro[1], 3)))
 ax[row,col].spines[["right", "top"]].set_visible(False)
 ax[row,col].set_xlabel("Nombre de références par thèse", labelpad = 8)
```

```

ax[row,col].set_ylabel("Fréquence", labelpad = 8)
col = col + 1
if col % 4 == 0:
 row = row + 1
 col = 0
fig.delaxes(ax[2][3])
fig.delaxes(ax[2][2])
for ax in ax.flat:
 ax.label_outer()
plt.subplots_adjust(wspace = 0.20, hspace = 0.32)
plt.show()

```



```

[26]: # (6) Kruskal-Wallis sur le nombre de références par thèse en fonction de la discipline
 ↪ discipline
my_dict = {}
for i in list(Moyenne_ESM_nb_ref_par_these_discipline.
 ↪ columns[[0,1,2,3,5,6,7,8,4,9]]):
 my_dict[i] = Nb_ref_avec_discipline[Nb_ref_avec_discipline["discipline"]
 ↪ == i]["nb_ref"]
stats.kruskal(my_dict["I"], my_dict["II"], my_dict["III"], my_dict["IV"],
 ↪ my_dict["V"], my_dict["VI"], my_dict["VII"], my_dict["VIII"],
 ↪ my_dict["IX"], my_dict["X"])

```

[26]: KruskalResult(statistic=850.2820953446758, pvalue=3.1724295932050167e-177)

```

[27]: # (7) Tests post-hoc ; rq : le seuil de significativité est abaissé à 0.05 / 55
 ↪ 55 = 9.09e-4 (55 étant le nombre de comparaisons deux-à-deux effectuées)

```

```
post_hoc_result = sp.posthoc_dunn([my_dict["I"], my_dict["II"],
↳my_dict["III"], my_dict["IV"], my_dict["V"], my_dict["VI"], my_dict["VII"],
↳my_dict["VIII"], my_dict["IX"], my_dict["X"]])
post_hoc_result
```

[27]:

|    | 1            | 2            | 3            | 4            | 5 \          |
|----|--------------|--------------|--------------|--------------|--------------|
| 1  | 1.000000e+00 | 4.388670e-02 | 2.030350e-01 | 7.502829e-01 | 5.540655e-14 |
| 2  | 4.388670e-02 | 1.000000e+00 | 3.315939e-01 | 3.754412e-03 | 2.844549e-89 |
| 3  | 2.030350e-01 | 3.315939e-01 | 1.000000e+00 | 1.386127e-01 | 1.508697e-45 |
| 4  | 7.502829e-01 | 3.754412e-03 | 1.386127e-01 | 1.000000e+00 | 6.319132e-49 |
| 5  | 5.540655e-14 | 2.844549e-89 | 1.508697e-45 | 6.319132e-49 | 1.000000e+00 |
| 6  | 8.114970e-09 | 7.591888e-60 | 6.088617e-30 | 4.671974e-30 | 6.979874e-06 |
| 7  | 1.000366e-17 | 1.931697e-83 | 2.067043e-49 | 2.914581e-51 | 3.184638e-03 |
| 8  | 8.688445e-07 | 4.720200e-40 | 1.115405e-21 | 4.732379e-20 | 9.836198e-08 |
| 9  | 1.137477e-09 | 1.847870e-80 | 8.705812e-36 | 1.986936e-38 | 3.243062e-07 |
| 10 | 3.086026e-04 | 2.812836e-28 | 1.628547e-14 | 2.454781e-12 | 3.043110e-18 |

|    | 6            | 7            | 8            | 9            | 10           |
|----|--------------|--------------|--------------|--------------|--------------|
| 1  | 8.114970e-09 | 1.000366e-17 | 8.688445e-07 | 1.137477e-09 | 3.086026e-04 |
| 2  | 7.591888e-60 | 1.931697e-83 | 4.720200e-40 | 1.847870e-80 | 2.812836e-28 |
| 3  | 6.088617e-30 | 2.067043e-49 | 1.115405e-21 | 8.705812e-36 | 1.628547e-14 |
| 4  | 4.671974e-30 | 2.914581e-51 | 4.732379e-20 | 1.986936e-38 | 2.454781e-12 |
| 5  | 6.979874e-06 | 3.184638e-03 | 9.836198e-08 | 3.243062e-07 | 3.043110e-18 |
| 6  | 1.000000e+00 | 1.189887e-10 | 1.283139e-01 | 6.194251e-01 | 2.442870e-06 |
| 7  | 1.189887e-10 | 1.000000e+00 | 1.870790e-12 | 4.260971e-12 | 9.463901e-23 |
| 8  | 1.283139e-01 | 1.870790e-12 | 1.000000e+00 | 3.137842e-02 | 6.266030e-03 |
| 9  | 6.194251e-01 | 4.260971e-12 | 3.137842e-02 | 1.000000e+00 | 2.582618e-09 |
| 10 | 2.442870e-06 | 9.463901e-23 | 6.266030e-03 | 2.582618e-09 | 1.000000e+00 |

[28]:

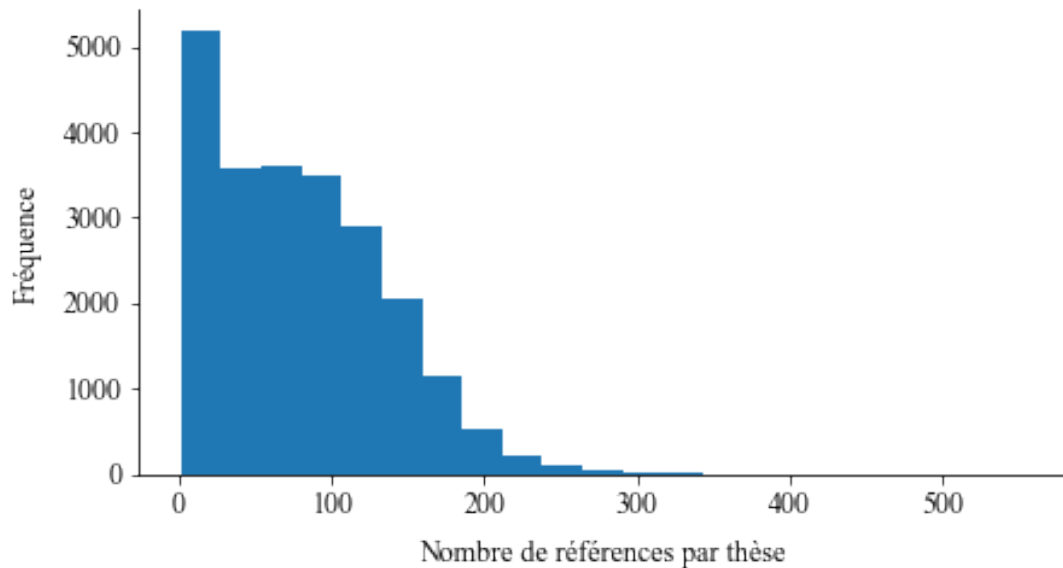
```
(8) Focus sur les thèses citant peu de références : affichage du nombre et
↳du pourcentage de thèses citant moins de 20 références
Nb_ref_par_these_discipline["somme"] = Nb_ref_par_these_discipline.sum(axis =
↳1, numeric_only = True)
print("Nombre de thèses citant moins de 20 références : " +
↳str(Nb_ref_par_these_discipline[Nb_ref_par_these_discipline["somme"] < 20].
↳shape[0]))
print("Pourcentage : " +
↳str(round(Nb_ref_par_these_discipline[Nb_ref_par_these_discipline["somme"]
↳< 20].shape[0] * 100 / Nb_ref_par_these_discipline.shape[0], 2)) + "%")
```

Nombre de thèses citant moins de 20 références : 3955  
 Pourcentage : 17.3%

[29]:

```
(9) Représentation de la distribution du nombre de références par thèse,
↳toute discipline confondue
plt.rc("font", family = "Times New Roman", size = 12)
fig, ax = plt.subplots(figsize = (17.5/2.54, 9/2.54))
ax.hist(Nb_ref_par_these_discipline["somme"], bins = 21)
ax.spines[["right", "top"]].set_visible(False)
ax.set_xlabel("Nombre de références par thèse", labelpad = 8)
ax.set_ylabel("Fréquence", labelpad = 8)
```

```
plt.show()
```



## 4 Type de référence citée par thèse et discipline

```
[30]: # (1) Retrait des références dont le type est inconnu
biblio_sans_pharma_ni_nan = biblio_sans_pharma[biblio_sans_pharma["type"].
→notnull()]

[31]: # (2) Calcul des proportions de chaque type de références citées par thèse,
→pour chaque discipline (création d'un tableau par discipline, regroupés
→ensuite dans un même dictionnaire)
my_dict = {}
for discipline in biblio_sans_pharma_ni_nan["discipline"].unique():
 my_df_number =
→biblio_sans_pharma_ni_nan[biblio_sans_pharma_ni_nan["discipline"] ==
→discipline].pivot_table(values = "numéro citation", index = "numéro TEL",
→columns = "type", aggfunc = "sum")
 my_df_prop = pd.DataFrame(index = my_df_number.index, columns =
→biblio_sans_pharma_ni_nan["type"].unique())
 my_df_prop = my_df_prop.fillna(0)
 for i in my_df_number.index:
 for j in my_df_number.columns:
 my_df_prop.loc[i,j] = my_df_number.loc[i,j]*100 / my_df_number.
→loc[i,:].sum()
 my_df_prop = my_df_prop.fillna(0)
 my_dict[discipline] = my_df_prop.copy(deep = True)
print("Proportion de chaque type de référence dans les thèses de la
→discipline I :")
print("")
print(my_dict["I"].head())
```

Proportion de chaque type de référence dans les thèses de la discipline I :

|                   | chapitre   | livre     | article   | papier de conférence | \        |
|-------------------|------------|-----------|-----------|----------------------|----------|
| numéro TEL        |            |           |           |                      |          |
| halshs-00005971v1 | 24.615385  | 16.153846 | 56.923077 |                      | 2.307692 |
| tel-00125323v1    | 13.215859  | 37.004405 | 49.779736 |                      | 0.000000 |
| tel-00150545v1    | 3.571429   | 69.642857 | 26.785714 |                      | 0.000000 |
| tel-00168467v1    | 100.000000 | 0.000000  | 0.000000  |                      | 0.000000 |
| tel-00184150v1    | 14.634146  | 36.585366 | 48.780488 |                      | 0.000000 |

|                   | thèse | rapport | page web | manuscrit | brevet | interview | \ |
|-------------------|-------|---------|----------|-----------|--------|-----------|---|
| numéro TEL        |       |         |          |           |        |           |   |
| halshs-00005971v1 | 0.0   | 0.0     | 0.0      | 0         | 0      | 0         |   |
| tel-00125323v1    | 0.0   | 0.0     | 0.0      | 0         | 0      | 0         |   |
| tel-00150545v1    | 0.0   | 0.0     | 0.0      | 0         | 0      | 0         |   |
| tel-00168467v1    | 0.0   | 0.0     | 0.0      | 0         | 0      | 0         |   |
| tel-00184150v1    | 0.0   | 0.0     | 0.0      | 0         | 0      | 0         |   |

|                   | film | communication | personnelle |
|-------------------|------|---------------|-------------|
| numéro TEL        |      |               |             |
| halshs-00005971v1 | 0    |               | 0           |
| tel-00125323v1    | 0    |               | 0           |
| tel-00150545v1    | 0    |               | 0           |
| tel-00168467v1    | 0    |               | 0           |
| tel-00184150v1    | 0    |               | 0           |

```
[32]: # (3) Calcul de la moyenne et de l'erreur standard de la proportion de chaque
 ↪ type de références citées par thèse, pour chaque discipline
Moyenne_Prop_type_ref_par_these = pd.DataFrame(index = my_dict.keys(),
 ↪ columns = biblio_sans_pharma_ni_nan["type"].unique())
ESM_Prop_type_ref_par_these = pd.DataFrame(index = my_dict.keys(), columns =
 ↪ biblio_sans_pharma_ni_nan["type"].unique())
for discipline in my_dict.keys():
 for i in my_dict[discipline].columns:
 Moyenne_Prop_type_ref_par_these.loc[discipline, i] =
 ↪ my_dict[discipline][i].values[~np.isnan(my_dict[discipline][i].values)].
 ↪ mean()
 ESM_Prop_type_ref_par_these.loc[discipline, i] =
 ↪ my_dict[discipline][i].values[~np.isnan(my_dict[discipline][i].values)].
 ↪ std() / np.sqrt(len(my_dict[discipline][i].values[~np.
 ↪ isnan(my_dict[discipline][i].values)]))
```

```
[33]: # (4) Représentation de la proportion moyenne de chaque type de références
 ↪ citées par thèse, en fonction de la discipline
___(4a) Regroupement des types autres que chapitre, livre et article dans
 ↪ une catégorie autre
Moyenne_Prop_type_ref_par_these_4_cat = Moyenne_Prop_type_ref_par_these.
 ↪ copy(deep = True)
Autre = []
for i in range(0, len(list(Moyenne_Prop_type_ref_par_these.index))):
```



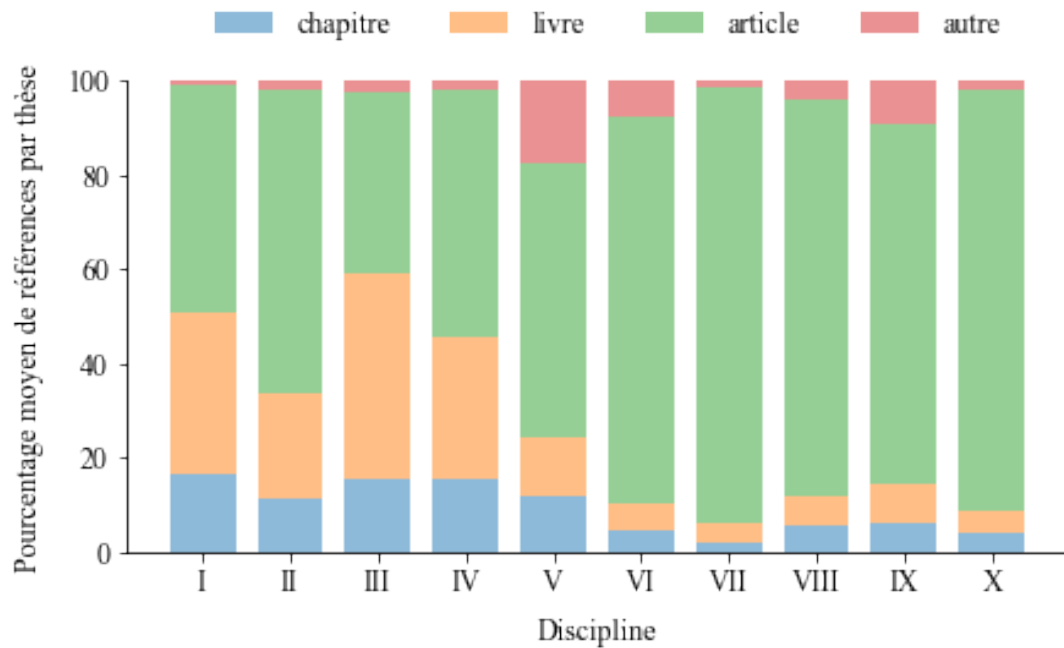
```

 Autre.append(np.array(Moyenne_Prop_type_ref_par_these_4_cat.iloc[i,3:]).
→sum())
Moyenne_Prop_type_ref_par_these_4_cat["autre"] = Autre
Moyenne_Prop_type_ref_par_these_4_cat = Moyenne_Prop_type_ref_par_these_4_cat.
→drop(columns = list(Moyenne_Prop_type_ref_par_these_4_cat.columns)[3:])

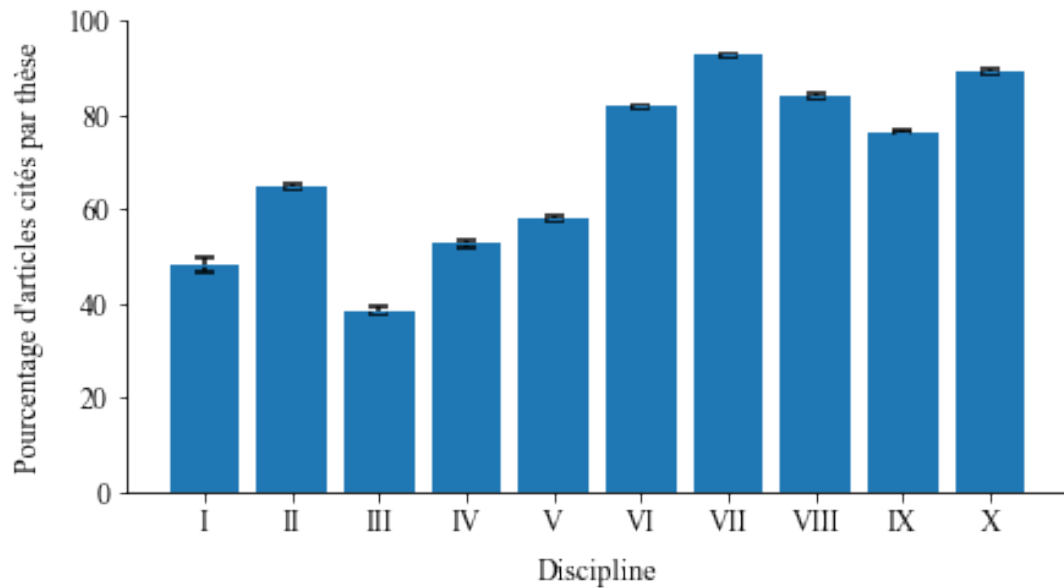
--- (4b) Représentation en stacked bar chart
List_types = list(Moyenne_Prop_type_ref_par_these_4_cat.columns)
k = np.array(list(Moyenne_Prop_type_ref_par_these_4_cat["chapitre"].
→values[[0,1,2,3,5,6,7,8,4,9]]))

plt.rc("font", family = "Times New Roman", size = 12)
fig, ax = plt.subplots(figsize = (17.5/2.54, 9/2.54))
plt.gca().spines["right"].set_visible(False)
plt.gca().spines["top"].set_visible(False)
ax.set_xlabel("Discipline", labelpad = 8)
ax.set_ylabel("Pourcentage moyen de références par thèse", labelpad = 8)
ax.set_ylim(0,100)
ax.bar(list(Moyenne_Prop_type_ref_par_these_4_cat["chapitre"].
→index[[0,1,2,3,5,6,7,8,4,9]]),
→list(Moyenne_Prop_type_ref_par_these_4_cat["chapitre"].
→values[[0,1,2,3,5,6,7,8,4,9]]), label = "chapitre", width = 0.75, alpha = 0.
→5)
for i in List_types[1:]:
 ax.bar(list(Moyenne_Prop_type_ref_par_these_4_cat[i].
→index[[0,1,2,3,5,6,7,8,4,9]]),
→list(Moyenne_Prop_type_ref_par_these_4_cat[i].
→values[[0,1,2,3,5,6,7,8,4,9]]), bottom = k, label = i, width = 0.75, alpha
→= 0.5)
 k = k + np.array(list(Moyenne_Prop_type_ref_par_these_4_cat[i].
→values[[0,1,2,3,5,6,7,8,4,9]]))
ax.legend(bbox_to_anchor = (0.065, 1.03), frameon = False, ncol = 4)
plt.show()

```



```
[34]: # (5) Représentation de la proportion d'articles scientifiques cités par
 ↳ thèse, en fonction de la discipline
plt.rc("font", family = "Times New Roman", size = 12)
fig, ax = plt.subplots(figsize = (17.5/2.54, 9/2.54))
ax.bar(x = list(Moyenne_Prop_type_ref_par_these.
↳ index[[0,1,2,3,5,6,7,8,4,9]]), height =
↳ list(Moyenne_Prop_type_ref_par_these["article"][[0,1,2,3,5,6,7,8,4,9]]))
ax.errorbar(x = list(Moyenne_Prop_type_ref_par_these.
↳ index[[0,1,2,3,5,6,7,8,4,9]]), y =
↳ list(Moyenne_Prop_type_ref_par_these["article"][[0,1,2,3,5,6,7,8,4,9]]),
↳ yerr = list(ESM_Prop_type_ref_par_these["article"][[0,1,2,3,5,6,7,8,4,9]]),
↳ fmt='_', capsize = 4, capthick = 1.5, ecolor = "black")
ax.spines[["right", "top"]].set_visible(False)
ax.set_xlabel("Discipline", labelpad = 8)
ax.set_ylabel("Pourcentage d'articles cités par thèse", labelpad = 8)
ax.set_ylim(0,100)
plt.show()
```



[35] : *# (6) Test de la normalité des données pour chaque discipline (représentation des distributions en affichant le résultat des tests dans le titre des graphiques)*

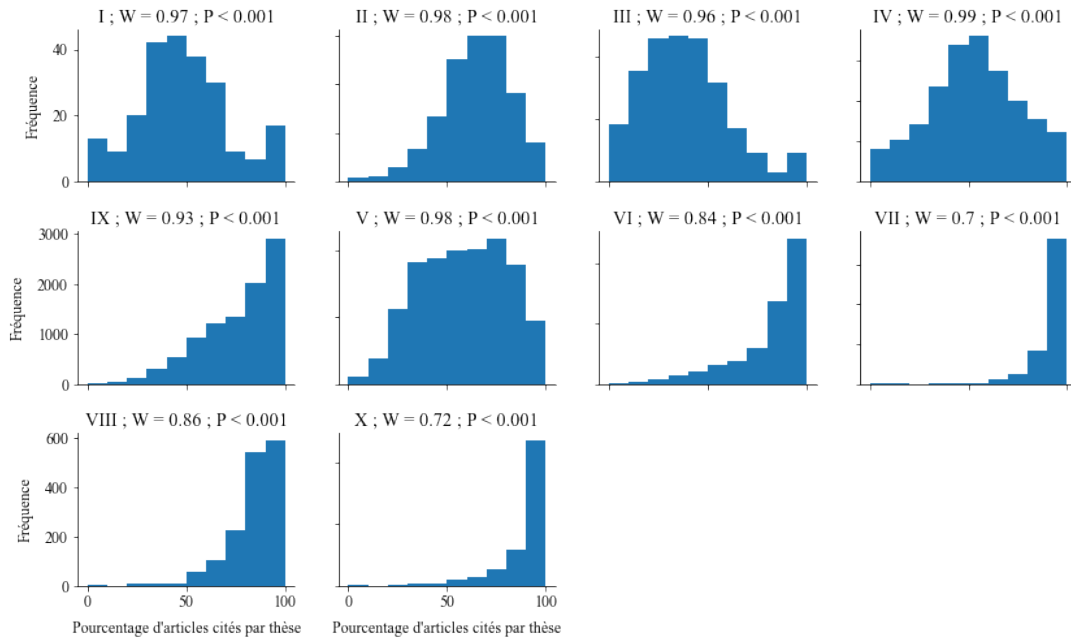
```
plt.rc("font", family = "Times New Roman", size = 12)
fig, ax = plt.subplots(nrows = 3, ncols = 4, figsize = (35/2.54, 20/2.54))
list_articles = []
for i in my_dict.keys():
 list_articles.append(list(my_dict[i]["article"]))

row = 0
col = 0
for i in range(0, len(list_articles)):
 test_shapiro = stats.shapiro(list_articles[i])
 ax[row,col].hist(list_articles[i], bins = 10)
 if test_shapiro[1] < 0.001:
 ax[row,col].set_title(list(my_dict.keys())[i] + " ; " + "W = " +
 str(round(test_shapiro[0], 2)) + " ; P < " + str(0.001))
 else:
 ax[row,col].set_title(list(my_dict.keys())[i] + " ; " + "W = " +
 str(round(test_shapiro[0], 2)) + " ; P = " + str(round(test_shapiro[1], 3)))
 ax[row,col].spines[["right", "top"]].set_visible(False)
 ax[row,col].set_xlabel("Pourcentage d'articles cités par thèse", labelpad=
 8)
 ax[row,col].set_ylabel("Fréquence", labelpad = 8)
 col = col + 1
 if col % 4 == 0:
 row = row + 1
 col = 0
fig.delaxes(ax[2][2])
fig.delaxes(ax[2][3])
```

```

for ax in ax.flat:
 ax.label_outer()
plt.subplots_adjust(wspace = 0.20, hspace = 0.32)
plt.show()

```



```

[36]: # (7) Kruskal-Wallis sur le pourcentage d'articles cités par thèse en
 ↪ fonction de la discipline
stats.kruskal(list_articles[0], list_articles[1], list_articles[2],
 ↪ list_articles[3], list_articles[4], list_articles[5], list_articles[6],
 ↪ list_articles[7], list_articles[8], list_articles[9])

```

[36]: KruskalResult(statistic=6605.611024663464, pvalue=0.0)

```

[37]: # (8) Tests post-hoc (le seuil de significativité est abaissé à 0.05 / 55 = 9.
 ↪ 09e-4)
post_hoc_result = sp.posthoc_dunn([list_articles[0], list_articles[1],
 ↪ list_articles[2], list_articles[3], list_articles[4], list_articles[5],
 ↪ list_articles[6], list_articles[7], list_articles[8], list_articles[9]])
post_hoc_result

```

```

[37]:
 ↪ \
1 1.000000e+00 6.216771e-10 3.574583e-04 1.137579e-01 7.531515e-54
2 6.216771e-10 1.000000e+00 3.855658e-51 4.851459e-15 2.240196e-92
3 3.574583e-04 3.855658e-51 1.000000e+00 8.620754e-15 2.482583e-229
4 1.137579e-01 4.851459e-15 8.620754e-15 1.000000e+00 5.677217e-168
5 7.531515e-54 2.240196e-92 2.482583e-229 5.677217e-168 1.000000e+00
6 1.279968e-04 5.902426e-08 2.973113e-35 6.044975e-05 2.204597e-284
7 2.193101e-81 9.242955e-153 4.808535e-292 8.694933e-232 1.635038e-38

```

|    |               |               |               |               |               |
|----|---------------|---------------|---------------|---------------|---------------|
| 8  | 2.108967e-160 | 0.000000e+00  | 0.000000e+00  | 0.000000e+00  | 5.906274e-198 |
| 9  | 5.457302e-83  | 1.234369e-136 | 3.762233e-271 | 4.103269e-209 | 5.677724e-34  |
| 10 | 2.469663e-136 | 1.877489e-286 | 0.000000e+00  | 0.000000e+00  | 8.032425e-167 |

|    |               |               |               |               |               |
|----|---------------|---------------|---------------|---------------|---------------|
|    | 6             | 7             | 8             | 9             | 10            |
| 1  | 1.279968e-04  | 2.193101e-81  | 2.108967e-160 | 5.457302e-83  | 2.469663e-136 |
| 2  | 5.902426e-08  | 9.242955e-153 | 0.000000e+00  | 1.234369e-136 | 1.877489e-286 |
| 3  | 2.973113e-35  | 4.808535e-292 | 0.000000e+00  | 3.762233e-271 | 0.000000e+00  |
| 4  | 6.044975e-05  | 8.694933e-232 | 0.000000e+00  | 4.103269e-209 | 0.000000e+00  |
| 5  | 2.204597e-284 | 1.635038e-38  | 5.906274e-198 | 5.677724e-34  | 8.032425e-167 |
| 6  | 1.000000e+00  | 0.000000e+00  | 0.000000e+00  | 8.067277e-268 | 0.000000e+00  |
| 7  | 0.000000e+00  | 1.000000e+00  | 3.523855e-75  | 9.857099e-02  | 1.195265e-45  |
| 8  | 0.000000e+00  | 3.523855e-75  | 1.000000e+00  | 1.941731e-51  | 1.048360e-07  |
| 9  | 8.067277e-268 | 9.857099e-02  | 1.941731e-51  | 1.000000e+00  | 1.208050e-27  |
| 10 | 0.000000e+00  | 1.195265e-45  | 1.048360e-07  | 1.208050e-27  | 1.000000e+00  |

## 5 Nombre de références citées par thèse et type de référence

### 5.1 Thèses de SHS

```
[38]: # (1) Création d'un tableau avec une colonne "thèse", une colonne
 ↪ "discipline", une colonne "type de références", une colonne "pourcentage de
 ↪ références citées" et une colonne "nombre de thèses"

Nb_ref_par_these_discipline_sans_nan = biblio_sans_pharma_ni_nan.
 ↪ pivot_table(values = "numéro citation", index = "numéro TEL", columns =
 ↪ "discipline", aggfunc = "sum").fillna(0)

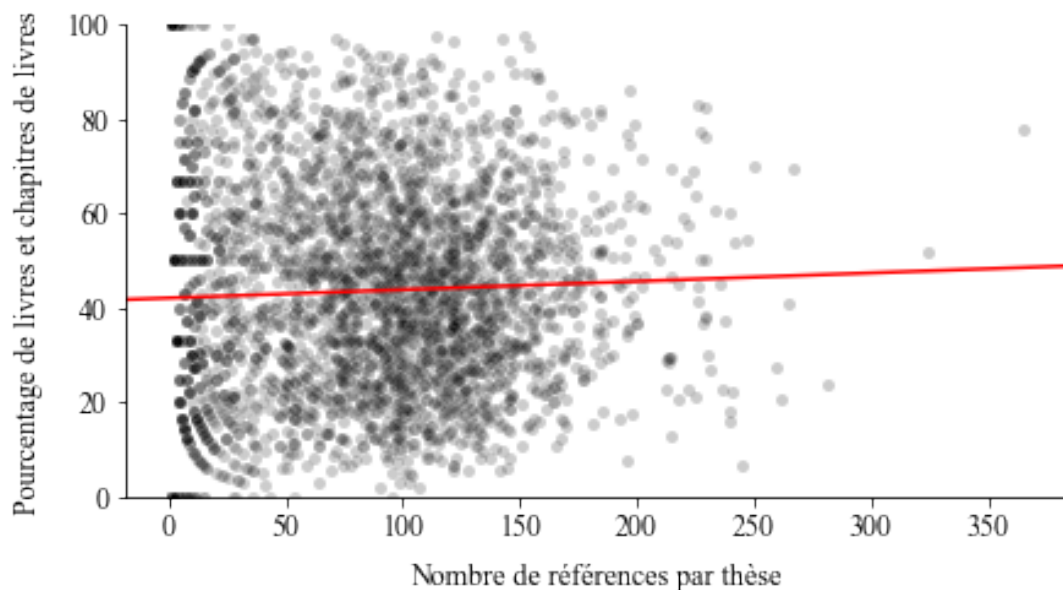
theses = list(Nb_ref_par_these_discipline_sans_nan.index)
nb_ref = [Nb_ref_par_these_discipline_sans_nan.loc[i,:].max() for i in
 ↪ Nb_ref_par_these_discipline_sans_nan.index]
Nb_ref_par_these = pd.DataFrame({"these": theses, "nb_ref": nb_ref})

df_final = pd.DataFrame(columns = ["these", "discipline", "type_ref",
 ↪ "prop_type_ref"])
for i in my_dict.keys():
 my_df = my_dict[i].copy(deep = True)
 my_df = my_df.stack().reset_index()
 my_df["these"] = my_df["numéro TEL"]
 my_df["discipline"] = [i] * my_df.shape[0]
 my_df["type_ref"] = my_df["level_1"]
 my_df["prop_type_ref"] = my_df[my_df.columns[2]]
 my_df = my_df.drop(columns = ["numéro TEL", "level_1", my_df.columns[2]])
 df_final = pd.merge(df_final, my_df, how = "outer")
df_final = pd.merge(df_final, Nb_ref_par_these, how = "left", on = "these")
df_final.head()
```

|   | these             | discipline | type_ref             | prop_type_ref | nb_ref |
|---|-------------------|------------|----------------------|---------------|--------|
| 0 | halshs-00005971v1 | I          | chapitre             | 24.615385     | 130.0  |
| 1 | halshs-00005971v1 | I          | livre                | 16.153846     | 130.0  |
| 2 | halshs-00005971v1 | I          | article              | 56.923077     | 130.0  |
| 3 | halshs-00005971v1 | I          | papier de conférence | 2.307692      | 130.0  |

```
[39]: # (2) Représentation de la proportion de livres et chapitres de livres cités
 ↪ par thèse en SHS en fonction du nombre de références citées par thèse
 # ___(2a) Sélection des thèses de SHS
 SHS = ["I", "II", "III", "IV"]
 df_SHS = pd.DataFrame(columns = list(df_final.columns))
 for i in SHS:
 my_df = df_final[df_final["discipline"] == i]
 df_SHS = pd.merge(df_SHS, my_df, how = "outer")
 prop_livres_chapitres = list(df_SHS[df_SHS["type_ref"] ==
 ↪ "livre"]["prop_type_ref"].values + df_SHS[df_SHS["type_ref"] ==
 ↪ "chapitre"]["prop_type_ref"].values)
 nb_ref = list(df_SHS[df_SHS["type_ref"] == "livre"]["nb_ref"].values)
 discipline = ["SHS"] * len(nb_ref)
 df_SHS = pd.DataFrame({"prop_livres_chapitres": prop_livres_chapitres,
 ↪ "nb_ref": nb_ref, "discipline": discipline})

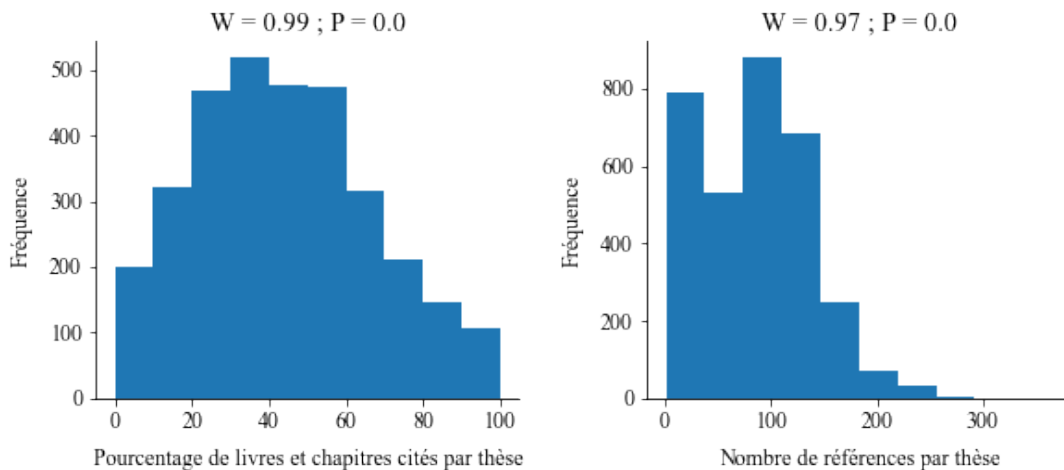
 # ___(2b) Représentation en scatter plot, la ligne rouge représentant la
 ↪ droite de régression
 model_SHS = ols("prop_livres_chapitres ~ nb_ref", data = df_SHS).fit()
 coeffs_SHS = model_SHS.params
 ic_SHS, slope_SHS = coeffs_SHS
 fig, ax = plt.subplots(figsize = (17.5/2.54, 9/2.54))
 sns.scatterplot(x = "nb_ref", y = "prop_livres_chapitres", data = df_SHS,
 ↪ alpha = 0.2, color = "black", s = 25)
 ax.axline(xy1 = (0, ic_SHS), slope = slope_SHS, color = "red")
 ax.set_ylim(0, 100)
 ax.set_xlabel("Nombre de références par thèse", labelpad = 8)
 ax.set_ylabel("Pourcentage de livres et chapitres de livres", labelpad = 8)
 ax.spines[["top", "right"]].set_visible(False)
 plt.show()
```



```
[40]: # (3) Affichage de l'équation de la droite de régression :
print("y = " + str(ic_SHS) + " + " + str(slope_SHS) + "x")
```

y = 42.059626652741066 + 0.017718721528044858x

```
[41]: # (4) Test de la normalité de la distribution des données des deux variables
→ analysées
test_shapiro_1 = stats.shapiro(df_SHS["prop_livres_chapitres"])
test_shapiro_2 = stats.shapiro(df_SHS["nb_ref"])
plt.rc("font", family = "Times New Roman", size = 12)
fig, ax = plt.subplots(nrows = 1, ncols = 2, figsize = (24/2.54, 9/2.54))
ax[0].hist(df_SHS["prop_livres_chapitres"], bins = 10)
ax[0].spines[["right", "top"]].set_visible(False)
ax[0].set_title("W = " + str(round(test_shapiro_1[0], 2)) + " ; P = " +
→ str(round(test_shapiro_1[1], 2)))
ax[0].set_xlabel("Pourcentage de livres et chapitres cités par thèse",
→ labelpad = 8)
ax[0].set_ylabel("Fréquence", labelpad = 8)
ax[1].hist(df_SHS["nb_ref"], bins = 10)
ax[1].spines[["right", "top"]].set_visible(False)
ax[1].set_title("W = " + str(round(test_shapiro_2[0], 2)) + " ; P = " +
→ str(round(test_shapiro_2[1], 2)))
ax[1].set_xlabel("Nombre de références par thèse", labelpad = 8)
ax[1].set_ylabel("Fréquence", labelpad = 8)
plt.subplots_adjust(wspace = 0.30)
plt.show()
```



```
[42]: # (5) Affichage des paramètres du modèle
print(model_SHS.summary())
```

OLS Regression Results

=====

```

=
Dep. Variable: prop_livres_chapitres R-squared:
0.002
Model: OLS Adj. R-squared:
0.001
Method: Least Squares F-statistic:
5.329
Date: Sat, 10 Sep 2022 Prob (F-statistic):
0.0210
Time: 11:07:02 Log-Likelihood:
-14815.
No. Observations: 3245 AIC:
2.963e+04
Df Residuals: 3243 BIC:
2.965e+04
Df Model: 1
Covariance Type: nonrobust
=====
 coef std err t P>|t| [0.025 0.975]

Intercept 42.0596 0.766 54.891 0.000 40.557 43.562
nb_ref 0.0177 0.008 2.308 0.021 0.003 0.033
=====
Omnibus: 104.517 Durbin-Watson: 1.619
Prob(Omnibus): 0.000 Jarque-Bera (JB): 84.084
Skew: 0.315 Prob(JB): 5.51e-19
Kurtosis: 2.525 Cond. No. 187.
=====

```

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

```

[43]: # (6) Calcul de l'erreur standard résiduelle
___(6a) Création d'une fonction dédiée à ce calcul
def erreur_standard_residuelle(my_model, nb_ind):
 """Calcule l'erreur standard résiduelle (ESR) d'un modèle de régression_
 ↪ linéaire.

 Parameters

 arg_1 : statsmodels.regression.linear_model.RegressionResultsWrapper
 Le modèle à partir duquel calculer l'ESR.
 arg_2 : int
 La taille de l'échantillon.

 Returns

 float
 """
 residuals_sq = my_model.resid ** 2

```



```

 resid_sum_of_sq = sum(residuals_sq)
 deg_freedom = nb_ind - 2
 res = np.sqrt(resid_sum_of_sq/deg_freedom)
 return res

___(6b) Affichage du résultat
print("ESR = " + str(erreur_standard_residuelle(model_SHS,
 ↪len(df_SHS["prop_livres_chapitres"].index))))

```

ESR = 23.259931079516917

```

[44]: # (7) Tests de corrélation de Pearson et de Spearman
print("Pearson :")
print(stats.pearsonr(df_SHS["prop_livres_chapitres"].dropna(),
 ↪df_SHS["nb_ref"].dropna()))
print("")
print("Spearman :")
print(stats.spearmanr(df_SHS["prop_livres_chapitres"].dropna(),
 ↪df_SHS["nb_ref"].dropna()))

```

Pearson :  
(0.04050192743832813, 0.021041105356744992)

Spearman :  
SpearmanrResult(correlation=0.052308319071037175, pvalue=0.0028765538946782044)

## 5.2 Thèses de sciences dures

```

[45]: # (1) Représentation de la proportion de livres et chapitres de livres cités
 ↪par thèse en sciences dures en fonction du nombre de références citées par
 ↪thèse

___(1a) Sélection des thèses de sciences dures
sciences_dures = ["V", "VI", "VII", "VIII", "IX", "X"]
df_sciences_dures = pd.DataFrame(columns = list(df_final.columns))
for i in sciences_dures:
 my_df = df_final[df_final["discipline"] == i]
 df_sciences_dures = pd.merge(df_sciences_dures, my_df, how = "outer")
prop_livres_chapitres = list(df_sciences_dures[df_sciences_dures["type_ref"]
 ↪== "livre"]["prop_type_ref"].values +
 ↪df_sciences_dures[df_sciences_dures["type_ref"] ==
 ↪"chapitre"]["prop_type_ref"].values)
nb_ref = list(df_sciences_dures[df_sciences_dures["type_ref"] ==
 ↪"livre"]["nb_ref"].values)
discipline = ["sciences_dures"] * len(nb_ref)
df_sciences_dures = pd.DataFrame({"prop_livres_chapitres":
 ↪prop_livres_chapitres, "nb_ref": nb_ref, "discipline": discipline})

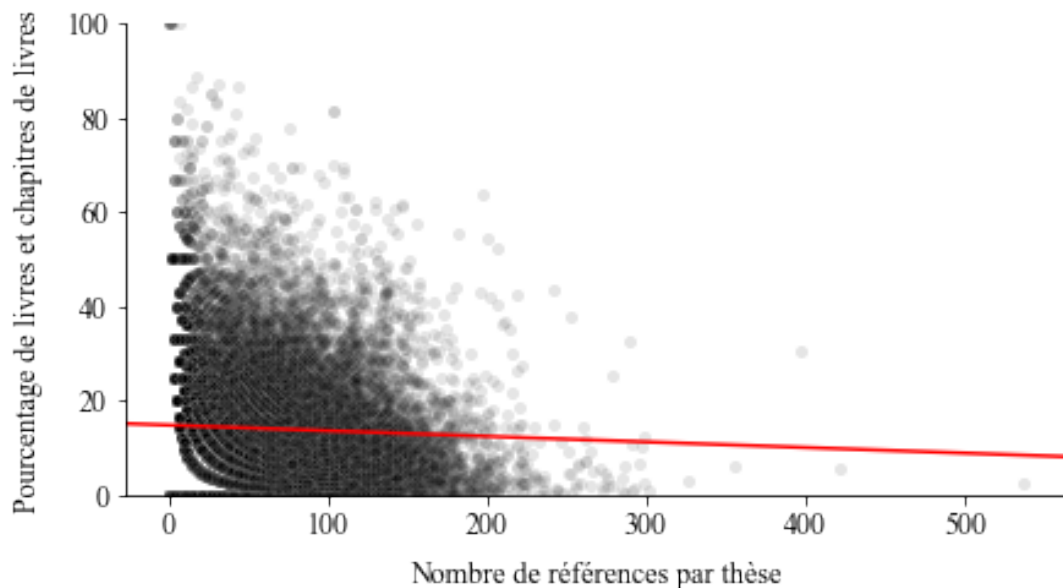
___(1b) Représentation en scatter plot, la ligne rouge représentant la
 ↪droite de régression
model_sciences_dures = ols("prop_livres_chapitres ~ nb_ref", data =
 ↪df_sciences_dures).fit()

```

```

coeffs_sciences_dures = model_sciences_dures.params
ic_sciences_dures, slope_sciences_dures = coeffs_sciences_dures
fig, ax = plt.subplots(figsize = (17.5/2.54, 9/2.54))
sns.scatterplot(x = "nb_ref", y = "prop_livres_chapitres", data =_
 ↳df_sciences_dures, alpha = 0.1, color = "black", s = 25)
ax.axline(xy1 = (0, ic_sciences_dures), slope = slope_sciences_dures, color =_
 ↳"red")
ax.set_ylim(0, 100)
ax.set_xlabel("Nombre de références par thèse", labelpad = 8)
ax.set_ylabel("Pourcentage de livres et chapitres de livres", labelpad = 8)
ax.spines[["top", "right"]].set_visible(False)
plt.show()

```



```

[46]: # (2) Affichage de l'équation de la droite de régression :
print("y = " + str(ic_sciences_dures) + " + " + str(slope_sciences_dures) +_
 ↳"x")

```

$y = 14.80058367648076 + -0.011882279487390686x$

```

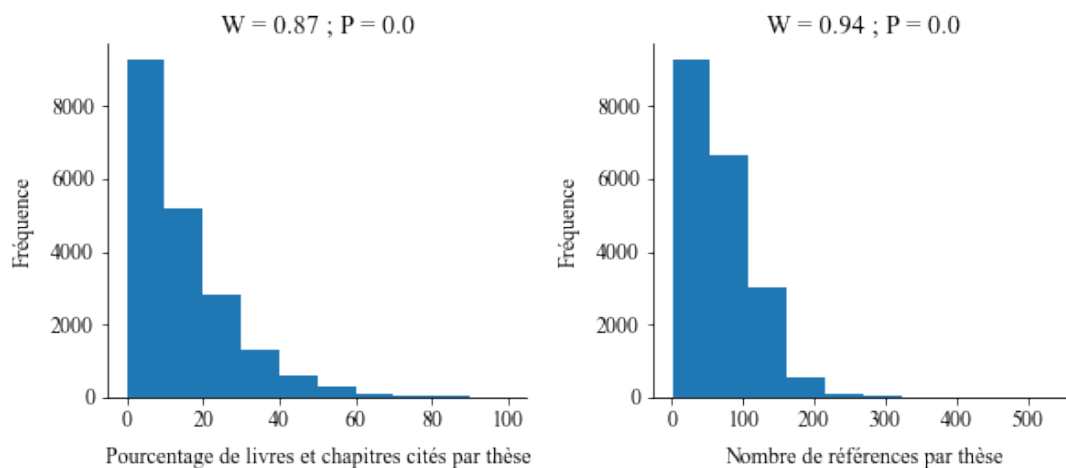
[47]: # (3) Test de la normalité de la distribution des données des deux variables_
 ↳analysées
test_shapiro_1 = stats.shapiro(df_sciences_dures["prop_livres_chapitres"])
test_shapiro_2 = stats.shapiro(df_sciences_dures["nb_ref"])
plt.rc("font", family = "Times New Roman", size = 12)
fig, ax = plt.subplots(nrows = 1, ncols = 2, figsize = (24/2.54, 9/2.54))
ax[0].hist(df_sciences_dures["prop_livres_chapitres"], bins = 10)
ax[0].spines[["right", "top"]].set_visible(False)
ax[0].set_title("W = " + str(round(test_shapiro_1[0], 2)) + " ; P = " +_
 ↳str(test_shapiro_1[1]))

```

```

ax[0].set_xlabel("Pourcentage de livres et chapitres cités par thèse",
 ↳labelpad = 8)
ax[0].set_ylabel("Fréquence", labelpad = 8)
ax[1].hist(df_sciences_dures["nb_ref"], bins = 10)
ax[1].spines[["right", "top"]].set_visible(False)
ax[1].set_title("W = " + str(round(test_shapiro_2[0], 2)) + " ; P = " +
 ↳str(test_shapiro_2[1]))
ax[1].set_xlabel("Nombre de références par thèse", labelpad = 8)
ax[1].set_ylabel("Fréquence", labelpad = 8)
plt.subplots_adjust(wspace = 0.30)
plt.show()

```



```

[48]: # (4) Affichage des paramètres du modèle
print(model_sciences_dures.summary())

```

```

 OLS Regression Results
=====
=
Dep. Variable: prop_livres_chapitres R-squared:
0.002
Model: OLS Adj. R-squared:
0.002
Method: Least Squares F-statistic:
35.67
Date: Sat, 10 Sep 2022 Prob (F-statistic):
2.38e-09
Time: 11:07:03 Log-Likelihood:
-78627.
No. Observations: 19615 AIC:
1.573e+05
Df Residuals: 19613 BIC:
1.573e+05
Df Model: 1
Covariance Type: nonrobust
=====

```

|                | coef    | std err  | t                 | P> t  | [0.025 | 0.975]    |
|----------------|---------|----------|-------------------|-------|--------|-----------|
| Intercept      | 14.8006 | 0.161    | 92.164            | 0.000 | 14.486 | 15.115    |
| nb_ref         | -0.0119 | 0.002    | -5.972            | 0.000 | -0.016 | -0.008    |
| =====          |         |          |                   |       |        |           |
| Omnibus:       |         | 5054.250 | Durbin-Watson:    |       |        | 1.702     |
| Prob(Omnibus): |         | 0.000    | Jarque-Bera (JB): |       |        | 12742.355 |
| Skew:          |         | 1.412    | Prob(JB):         |       |        | 0.00      |
| Kurtosis:      |         | 5.759    | Cond. No.         |       |        | 136.      |
| =====          |         |          |                   |       |        |           |

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

```
[49]: # (5) Calcul de l'erreur standard résiduelle
print("ESR = " + str(erreur_standard_residuelle(model_sciences_dures,
→len(df_sciences_dures["prop_livres_chapitres"].index))))
```

ESR = 13.324698101939783

```
[50]: # (6) Tests de corrélation de Pearson et de Spearman
print("Pearson :")
print(stats.pearsonr(df_sciences_dures["prop_livres_chapitres"].dropna(),
→df_sciences_dures["nb_ref"].dropna()))
print("")
print("Spearman :")
print(stats.spearmanr(df_sciences_dures["prop_livres_chapitres"].dropna(),
→df_sciences_dures["nb_ref"].dropna()))
```

Pearson :

(-0.04260592596936459, 2.380665738031805e-09)

Spearman :

SpearmanrResult(correlation=0.09830164150640021, pvalue=2.546745904878555e-43)