Séance 2 : Principe généraux de la statistique

Question 1 : La géographie est une discipline qui a énormément évolué depuis qu'elle existe et qu'elle est considérée comme scientifique. Les mathématiques, et plus particulièrement les statistiques, constituent un outil d'une très grande importance pour le géographe qui peut l'utiliser pour traiter une quantité importante de données. Une étude géographique d'un phénomène ou d'un espace dégage une masse de données que souvent seules les statistiques peuvent simplifier, expliciter, et transformer en une matière scientifique exploitable pour celui qui l'étudie. Néanmoins, il en reste que les mathématiques ne sont pas encore communément acceptées par tous les géographes, beaucoup restent réticents à son utilisation (réticence que l'on peut imputer à la formation de ces géographes souvent plus tournée vers les lettres que les mathématiques), et son caractère nouveau dans la discipline, qui entraîne de fait une vague de conservatisme.

Question 2 : Effectivement, il existe un hasard en géographie, comme il existe un hasard dans presque toutes les disciplines du monde. Question de positionnement philosophique dans un second temps, il semble que les scientifiques s'accordent sur le fait que le hasard existe et qu'il peut plus ou moins être limité, borné, pour établir une certitude globale. En géographie, possibilisme vidalien et déterminisme s'affrontentaffrontent sur le positionnement global de la discipline quant au hasard, et il est admis aujourd'hui que c'est celle de Vidal de la Blache qui l'a remporté. Toutefois, malgré tout cela, la géographie française reste assez ambiguëambiguë dans la mesure où elle n'a pas pleinement introduit les mathématiques dans sa formation,formation, ce qui la différencie des autres nationalités, et surtout bride sa vision complète du hasard en géographie.

Question 3 : Il existe deux séries de statistiques qui servent à étudier soit la forme géométrique des attributs d'un espace, ou bien les attributs en eux-mêmes, leur présence sur cet espace et leurs caractéristiques. Pour traiter ces données, le géograph peut mettre en place une nomenclature de celles-ci, afin d'y gagner en lisibilité et les higéographede lesérarchiser (ou non). Les métadonnéesde lesmétadonnées qui en ressortent agissent comme vérificateur de sources, c'est-à-dire qu'elles regroupent en leur en leur sein d'autres données et nomenclatures, permettant ainsi de dézoomer la masse de données à une échelle bien plus compréhensible pour le cerveau humain.

Question 4 : Cfcf.Réponses Questions 1, 2 et 3 cf.3.

Question 5 : Les statistiques descriptives ont pour but deranger, c3.de dérangerlasser, rde dérangerde classereprésenter lade classerde représenter quantité massive de données afin de décrire une situation, la réalité d'un espace, la distribution des attributs au sein d'un territoire. Elle représentera donc un phénomène ou une situation sous forme de graphiques ou d'indicateurs sans

forcément les mettre en relation avec une variable. Les statistiques explicatives de représenterexplicatives,quant à elle rexplicatives,elles,endent compte d'un phénomène en fonction d'uneelles,d'une variable. Ici, on cherche à comprendre lod'unelee lien d'inter-dépendancleinterdépendancee entre une variable dite dépendante et une autre dite indépendante. Ces statistiques sont beaucoup plus utilisée interdépendanceutiliséespour rentrer en profondeur dans l'interrogation et l'explication d'un phénomène géographique enutiliséesenf fonction des caractéristiques du lieu d'occurence.enoccurrence

Question 6 : En géographie, les types de visualisation dépendent de la nature des données étudiées. Les variables quantitatives continues (comme l'altitude, le revenu ou le nombre d'habitants) sont souvent représentées par des histogrammes, ou des courbes cumulatives. Les variables qualitatives (comme le type d'usage du sol ou le parti politique en tête) sont mieux représentées par des diagrammes en barres ou des camemberts.

En cartographie, les données peuvent aussi être visualisées à travers des symboles proportionnels ou des cartes de flux. Le choix de la visualisation dépend donc à la fois du type de variable (qualitative ou quantitative), de la question posée (comparer, répartir, localiser) et de l'échelle géographique d'analyse.

Question 7 : On distingue plusieurs grandes familles de méthodes. Les méthodes descriptives regroupent les techniques de réduction et de visualisation de données comme l'analyse en composantes principales (ACP) pour les variables quantitatives, ou l'analyse factorielle des correspondances (AFC) pour les variables qualitatives. Les méthodes explicatives s'appuient sur des modèles statistiques permettant de relier une variable à expliquer à d'autres variables explicatives, par exemple à travers une régression linéaire, une analyse de variance ou une régression logistique. Enfin, les méthodes de prévision concernent l'étude de séries chronologiques, où l'on cherche à prédire une valeur future à partir des données passées.

Question 8 : (a) Une population statistique peut être définitoccurrencedéfinie comme un ensemble d'éléments ou d'attributs qui partage pdéfiniepartagentlus ou moins les mêmes caractéristiques, à entendre comme un ensemble mathématique.

(b) Aussi appelé unité statistique, l'individu statistique correspond à un seul élément de la population statistique.

(partagent(c ()) On appelle caractère statistique toutes les singularités propres à chacune)chacune des unités et sur lesquelles peuvent se reposer l'étude statistique.

(d) La modalité est une condition que l'on pose pour trier les données selon si oui ou non elles présentent un (ou des) caractères spécifiques.

Il existe deux types de caractères : les variables qualitatives ou quantitatives. Toutefois, il n'existe pas de hiérarchie claire entre elles..

Question 9 : L'amplitude en statistique se calcule en soustrayant à la valeur la plus forte de la population la valeur la plus basse. Ainsi, sur la population d'un club de foot, on prendra l'âge du licencié le plus vieux et du plus jeune, leur soustraction nous indiquera l'amplitude d'âge dans le club. La densité, c'est l'effectif divisé par l'amplitude, elle permet de calculer des classes d'amplitude.

Question 10 : Les formules de Sturges et de Yule servent à déterminer le nombre de classes optimal pour construire un histogramme et gagner le plus possible en lisibilité. Elles permettent d'éviter un découpage trop fin (qui rend le graphique illisible) ou trop grossier (qui fait perdre de l'information). Elle se base sur le nombre de classess et l'amplitude des données recueillies.

Question 11 : L'effectif se définit comme le nombre d'occurrenceses d'une variable dans l'ensemble d'une classe. Dans la classe régions françaisess avec plus de 5 millions d'habitants, s'il y a 7 régions avec plus de 5 5 M d'habitantsts, alors la fréquence absolue est de 7.

Elle se calcule de la sorte : fi=ni

La fréquence cumulée additionne les fréquences jusqu'à une certaine modalité ou classe, ce qui permet de connaître la part d'observations inférieures ou égales à une valeur donnée. Elle se calcule de la sorte : fi = ∑i=1k ni ≤ k.

La fréquence permet d'établir une distribution statistique, qui elle-même se définit comme un tableau qui répertorie les classes de valeurs obtenues.

```
# coding:utf-8
import os
import pandas as pd

# === Chemin du CSV ===
CHEMIN_CSV =
"/home/thomas04dupuy/Desktop/resultats-elections-presidentielles-2022-1
er-tour.csv"

# SÃ©curisation : si le fichier n'existe pas, message clair et sortie
propre
if not os.path.exists(CHEMIN_CSV):
   print("â Œ Erreur : fichier introuvable :")
   print(f"   {CHEMIN_CSV}")
```

```
   print("ðŸ‘‰ VÃ©rifie l'emplacement exact du fichier puis mets Ã
jour CHEMIN_CSV.")
   raise SystemExit(1)

# === Chargement des donnÃ©es ===
df = pd.read_csv(CHEMIN_CSV)

# --------------------------------
# Question 5 : afficher les 5 premiÃ¨res lignes
# --------------------------------
print("\n=== Q5 â€¢ AperÃ§u des donnÃ©es (head) ===")
print(df.head())

# --------------------------------
# Question 6 : nb de lignes / colonnes
# --------------------------------
nb_lignes = len(df)
nb_colonnes = len(df.columns)
print("\n=== Q6 â€¢ Dimensions du tableau ===")
print("Nombre de lignes :", nb_lignes)
print("Nombre de colonnes :", nb_colonnes)

# --------------------------------
# Question 7 : type (statistique) de chaque variable
# Conversion en types simples : int / float / bool / str
# --------------------------------
print("\n=== Q7 â€¢ Types de donnÃ©es par colonne ===")
for colonne, type_donnee in df.dtypes.items():
   t = str(type_donnee)
   if "int" in t:
       type_python = "int"
   elif "float" in t:
       type_python = "float"
   elif "bool" in t:
       type_python = "bool"
   else:
       type_python = "str"
   print(f"- {colonne} : {type_python}")

# --------------------------------
# Question 8 : premiÃ¨re ligne + noms des colonnes
# --------------------------------
print("\n=== Q8 â€¢ PremiÃ¨re ligne du tableau ===")
```

```
print(df.head(1))
print("\n=== Q8 â€¢ Noms des colonnes ===")
print(df.columns.tolist())

# ------------------------------
# Question 9 : sÃ©lectionner la colonne 'Inscrits'
# ------------------------------
print("\n=== Q9 â€¢ Colonne 'Inscrits' ===")
if "Inscrits" in df.columns:
   inscrits = df["Inscrits"]
   print(inscrits)
else:
   print("â Œ La colonne 'Inscrits' n'existe pas dans ce fichier.")

# ------------------------------
# Question 10 : somme (effectif) des colonnes quantitatives
# - Boucle
# - sum() Pandas
# - Filtrage par type (int/float)
# - Stockage des rÃ©sultats dans une liste via append()
# ------------------------------
print("\n=== Q10 â€¢ Somme des colonnes quantitatives ===")
liste_sommes = []  # contiendra des tuples (nom_colonne, somme)

for colonne in df.columns:
   dtype_str = str(df[colonne].dtype)
   if ("int" in dtype_str) or ("float" in dtype_str):
       somme = df[colonne].sum()
       liste_sommes.append((colonne, somme))  # mÃ©thode native append
   # sinon on ignore la colonne (texte/bool/etc.)

# Affichage des rÃ©sultats
for nom, total in liste_sommes:
   print(f"- {nom} : {total}")

#Question 11

# coding:utf-8
import os
import re
import math
import pandas as pd
import matplotlib.pyplot as plt
```

```
# === Chemin du CSV (adapte si besoin) ===
CHEMIN_CSV =
"/home/thomas04dupuy/Desktop/resultats-elections-presidentielles-2022-1
er-tour.csv"

# === Lecture des donnÃ©es ===
df = pd.read_csv(CHEMIN_CSV)

# VÃ©rifications basiques
colonnes_requises = ["Code du dÃ©partement", "LibellÃ© du
dÃ©partement", "Inscrits", "Votants"]
manquantes = [c for c in colonnes_requises if c not in df.columns]
if manquantes:
    raise ValueError(f"Colonnes manquantes dans le CSV : {manquantes}")

# Remplace NaN par 0 pour Ã©viter les plantages au tracÃ©/somme
df[["Inscrits", "Votants"]] = df[["Inscrits", "Votants"]].fillna(0)

# === Dossier de sortie pour les images ===
OUT_DIR = "charts_inscrits_votants"
os.makedirs(OUT_DIR, exist_ok=True)

def slugify(s: str) -> str:
    """Nettoie une chaÃ®ne pour l'utiliser dans un nom de fichier."""
    s = s.strip().lower()
    s = re.sub(r"[^\w\s-]", "", s, flags=re.UNICODE)  # enlÃ¨ve
accents/ponctuation
    s = re.sub(r"[\s_-]+", "-", s)
    return s

# === Boucle : 1 image par dÃ©partement ===
for _, row in df.iterrows():
    code = str(row["Code du dÃ©partement"])
    libelle = str(row["LibellÃ© du dÃ©partement"])
    inscrits = 0 if (pd.isna(row["Inscrits"]) or
(isinstance(row["Inscrits"], float) and math.isnan(row["Inscrits"])))
else float(row["Inscrits"])
    votants  = 0 if (pd.isna(row["Votants"])  or
(isinstance(row["Votants"],  float) and math.isnan(row["Votants"])))
else float(row["Votants"])

    # PrÃ©pare le nom de fichier
```

```
    fname = f"{code}-{slugify(libelle)}.png"
    out_path = os.path.join(OUT_DIR, fname)

    # --- TracÃ© ---
    # Une figure par dÃ©partement (2 barres : Inscrits, Votants)
    plt.figure(figsize=(6, 4))
    categories = ["Inscrits", "Votants"]
    valeurs = [inscrits, votants]
    plt.bar(categories, valeurs)

    # Titres et axes
    plt.title(f"{code} â€“ {libelle} : Inscrits vs Votants")
    plt.ylabel("Effectif")
    # Optionnel : valeurs au-dessus des barres
    for i, v in enumerate(valeurs):
        plt.text(i, v, f"{int(v):,}".replace(",", " "), ha="center",
va="bottom", fontsize=9)

    plt.tight_layout()
    plt.savefig(out_path, dpi=150)
    plt.close()

print(f" Diagrammes gÃ©nÃ©rÃ©s dans : {os.path.abspath(OUT_DIR)}")

#Question 12
# Camemberts Blancs / Nuls / ExprimÃ©s / Abstentions par dÃ©partement
===
import os
import re
import math
import pandas as pd
import matplotlib.pyplot as plt

# VÃ©rification des colonnes nÃ©cessaires
colonnes_pie = [
    "Blancs", "Nuls", "ExprimÃ©s", "Abstentions",
    "Code du dÃ©partement", "LibellÃ© du dÃ©partement"
]
manquantes = [c for c in colonnes_pie if c not in df.columns]
if manquantes:
    raise ValueError(f"Colonnes manquantes pour les camemberts :
{manquantes}")
```

```
# Remplir les NaN par 0 pour Ã©viter les plantages
df[["Blancs", "Nuls", "ExprimÃ©s", "Abstentions"]] = df[
    ["Blancs", "Nuls", "ExprimÃ©s", "Abstentions"]
].fillna(0)

# Dossier de sortie
OUT_DIR_PIE = "charts_votes_pie"
os.makedirs(OUT_DIR_PIE, exist_ok=True)


def slugify(s: str) -> str:
    """Nettoie une chaÃ®ne pour l'utiliser dans un nom de fichier."""
    s = s.strip().lower()
    s = re.sub(r"[^\w\s-]", "", s, flags=re.UNICODE)
    s = re.sub(r"[\s_-]+", "-", s)
    return s


# Boucle : 1 image par dÃ©partement
for _, row in df.iterrows():
    code = str(row["Code du dÃ©partement"])
    libelle = str(row["LibellÃ© du dÃ©partement"])

    blancs = float(row["Blancs"]) if pd.notna(row["Blancs"]) else 0.0
    nuls = float(row["Nuls"]) if pd.notna(row["Nuls"]) else 0.0
    exprimes = float(row["ExprimÃ©s"]) if pd.notna(row["ExprimÃ©s"])
else 0.0
    abst = float(row["Abstentions"]) if pd.notna(row["Abstentions"])
else 0.0

    labels = ["Blancs", "Nuls", "ExprimÃ©s", "Abstentions"]
    vals = [blancs, nuls, exprimes, abst]
    total = sum(vals)

    # Si tout est Ã  0, on ignore
    if total == 0:
        continue

    # Nom du fichier image
    fname = f"{code}-{slugify(libelle)}-votes-pie.png"
    out_path = os.path.join(OUT_DIR_PIE, fname)

    # CrÃ©ation du diagramme circulaire
```

```
    plt.figure(figsize=(5.5, 5.5))
    plt.pie(
        vals,
        labels=labels,
        autopct=lambda p: f"{p:.1f}%" if p > 0 else "",
        startangle=90
    )
    plt.title(f"{code} â€“ {libelle}\nRÃ©partition Blancs / Nuls / ExprimÃ©s / Abstentions")
    plt.axis("equal")  # Rend le camembert bien rond
    plt.tight_layout()
    plt.savefig(out_path, dpi=150)
    plt.close()

print(f" Camemberts gÃ©nÃ©rÃ©s dans : {os.path.abspath(OUT_DIR_PIE)}")

#Question 13

# === Histogramme de la distribution des inscrits ===
import matplotlib.pyplot as plt
import os

# VÃ©rifier que la colonne "Inscrits" existe
if "Inscrits" not in df.columns:
    raise ValueError("La colonne 'Inscrits' n'existe pas dans le DataFrame.")

# Supprimer les valeurs manquantes (NaN) avant le tracÃ©
donnees_inscrits = df["Inscrits"].dropna()

# Dossier de sortie pour enregistrer le graphique
OUT_DIR_HISTO = "charts_histogrammes"
os.makedirs(OUT_DIR_HISTO, exist_ok=True)

# CrÃ©ation de l'histogramme
plt.figure(figsize=(8, 5))
plt.hist(donnees_inscrits, bins=20, color="skyblue", edgecolor="black")

# Titres et labels
plt.title("Distribution du nombre d'inscrits par dÃ©partement")
plt.xlabel("Nombre d'inscrits")
plt.ylabel("FrÃ©quence")
```

```
# Grille et mise en page
plt.grid(axis="y", linestyle="--", alpha=0.7)
plt.tight_layout()

# Sauvegarde de l'image
out_path = os.path.join(OUT_DIR_HISTO, "histogramme_inscrits.png")
plt.savefig(out_path, dpi=150)
plt.close()

print(f" Histogramme gÃ©nÃ©rÃ© : {os.path.abspath(out_path)}")

#Questionn Bonus
# === Camemberts : rÃ©partition des VOIX par candidat, pour chaque
dÃ©partement ===
import os, re
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

# Nettoyage des noms de colonnes (espaces superflus)
df.columns = df.columns.str.strip()

# VÃ©rifs de base
cols_dept = ["Code du dÃ©partement", "LibellÃ© du dÃ©partement"]
for c in cols_dept:
   if c not in df.columns:
       raise ValueError(f"Colonne manquante : {c}")

# Trouver dynamiquement toutes les colonnes de voix et leurs suffixes
("" , ".1", ".2", ...)
voix_cols = [c for c in df.columns if re.fullmatch(r"Voix(\.\d+)?", c)]
if not voix_cols:
   raise ValueError("Aucune colonne 'Voix' trouvÃ©e (ex: 'Voix',
'Voix.1', ...).")

# DÃ©duire les suffixes Ã  partir des colonnes 'Voix'
# Exemple: 'Voix' -> suffixe "", 'Voix.1' -> suffixe ".1"
suffixes = [c.replace("Voix", "") for c in voix_cols]

# Dossier de sortie
OUT_DIR = "charts_voix_par_candidat_pie"
os.makedirs(OUT_DIR, exist_ok=True)
```

```
def slugify(s: str) -> str:
    s = (s or "").strip().lower()
    s = re.sub(r"[^\w\s-]", "", s, flags=re.UNICODE)
    s = re.sub(r"[\s_-]+", "-", s)
    return s or "inconnu"

# Boucle : 1 camembert par dÃ©partement
for _, row in df.iterrows():
    code = str(row["Code du dÃ©partement"])
    libelle = str(row["LibellÃ© du dÃ©partement"])

    labels = []
    values = []

    for suf in suffixes:
        col_nom = f"Nom{suf}"
        col_pre = f"PrÃ©nom{suf}"
        col_vox = f"Voix{suf}"

        # Si la colonne 'Voix' existe bien pour ce suffixe
        if col_vox in df.columns:
            voix = row[col_vox]
            # Convertir en numÃ©rique, ignorer NaN ou 0
            try:
                v = float(voix) if pd.notna(voix) else 0.0
            except Exception:
                v = 0.0
            if v > 0:
                # Construire un label propre : "PrÃ©nom Nom" si dispo,
sinon "Nom"
                nom = str(row[col_nom]) if col_nom in df.columns and
pd.notna(row[col_nom]) else ""
                pre = str(row[col_pre]) if col_pre in df.columns and
pd.notna(row[col_pre]) else ""
                label = (pre + " " + nom).strip()
                if not label:
                    label = f"Candidat{suf or ''}".strip()
                labels.append(label)
                values.append(v)

    total = sum(values)
    if total <= 0:
        # Rien Ã  afficher pour ce dÃ©partement
```

```
        continue

    # CrÃ©er la figure
    plt.figure(figsize=(6, 6))
    # Matplotlib choisira les couleurs par dÃ©faut (consigne : aucune
palette imposÃ©e)
    plt.pie(
        values,
        labels=labels,
        autopct=lambda p: f"{p:.1f}%" if p > 0 else "",
        startangle=90
    )
    plt.title(f"{code} â€“ {libelle}\nRÃ©partition des voix par candidat
(1er tour)")
    plt.axis("equal")
    plt.tight_layout()

    # Sauvegarde
    fname = f"{code}-{slugify(libelle)}-voix-par-candidat.png"
    out_path = os.path.join(OUT_DIR, fname)
    plt.savefig(out_path, dpi=150)
    plt.close()

print(f" Camemberts par candidat gÃ©nÃ©rÃ©s dans :
{os.path.abspath(OUT_DIR)}")
```

Séance 3 : statistiques univariées (1). Paramètres statistiques élémentaires

Questions de cour :

Question 1 : Les caractères dits qualitatifs sont plus généraux que les caractères dits quantitatifs. Les variables qualitatives recensent une qualité, une propriété d'une population, comme par exemple l'appartenance politique. Les variables quantitatives mesurent une quantité, donc quelque chose que l'on peut mesurer, par exemple, la taille (en cm). Toutefois, avec ce même exemple, on peut la transformer en variable qualitative si l'on divise les catégories en "petit", "moyen" et "grand". En ça, les caractères qualitatifs sont plus généraux.

Question 2 : Les variables quantitatives discrètes ne peuvent prendre qu'une certaine valeur, souvent entière, comprise dans un intervalle connu. Par exemple, si l'on considère un groupe d'individus où la personne avec le plus d'enfants au sein de celui-ci aurait 4 enfants, le probable nombre d'enfants des autres individus se situera

forcément dans l'intervalle [0 ; 4], soit 5 possibilités. Le caractère quantitatif continu s'inscrit aussi dans un intervalle mais peut avoir une infinité de valeurs possibles, notamment avec les nombres décimaux. La grande différence entre ces deux caractères, c'est qu'un sert à compter/décompter tandis que l'autre cherche plus à mesurer.

Question 3 : Il existe plusieurs types de moyennes car chacune rend compte plus ou moins précisément d'un caractère que l'on souhaite mettre en valeur dans nos calculs, et toutes comportent leur lot de qualités et de défauts. Par exemple, la moyenne dite arithmétique est beaucoup plus sensible aux valeurs extrêmes.

La médiane a l'avantage de classer la population en deux catégories comprenant la même quantité de part et d'autre d'une valeur qu'on a fixée. Typiquement, le salaire médian permet de comprendre à combien de rémunération on doit être pour être plus riche que la moitié des Français. Elle n'est pas influencée par les valeurs extrêmes.

Le mode correspond à la valeur qui a le plus de chances de tomber dans une série statistique, il s'agit d'une moyenne de fréquence. Il peut être calculé par tous les types de variables, nominale, ordinale, discrète mais pas continue. Et si toutes les valeurs n'apparaissent qu'une seule fois, alors la série statistique ne peut avoir de mode.

Question 4 : L'intérêt principal de la médiale est qu'elle correspond plus ou moins à la médiane, sauf qu'elle en règle un des plus gros défauts, elle prend en compte les valeurs globales relatives de la population statistique. Ainsi, au lieu de seulement diviser la population en 2 selon une variable, il prend en compte le nombre total de la population et la divise en 2 pour que chaque partie ait une valeur égale. L'indice de Gini est un peu comme la mise en application graphique de la médiane puisqu'elle a pour but de représenter dans un repère orthonormé l'effet de concentration d'une population. En lisant la courbe de l'indice de Gini, on comprend mieux si la concentration au sein d'une population est grande ou pas, et donc les effets que cela peut avoir sur la série statistique.

Question 5 : Pour mesurer la dispersion, l'écart à la moyenne n'est pas un bon moyen car il renvoie toujours à 0. Ainsi, en élevant toutes les valeurs au carré, on obtient un résultat positif qui nous permet de mieux interpréter les résultats même si les valeurs au carré ne nous aident pas. Alors, l'écart-type se retrouve en mettant au carré les valeurs obtenues avec la variance. De fait, on retombe dans l'unité statistique de base et on comprend dès lors beaucoup mieux l'écart à la moyenne.

L'étendue sert à mesurer l'amplitude totale des valeurs observées,

c'est-à-dire l'écart global entre les extrêmes d'un ensemble de données. Elle indique jusqu'où les données s'étalent.

Un quantile sert à découper une série de données ordonnées en parties égales afin d'analyser leur répartition. Les quantiles les plus utilisés sont les centiles (10 groupes créés), le quartile (4 groupes créés) et les centiles (100 groupes créés).

Une boîte de dispersion est une représentation graphique d'un traitement statistique permettant d'observer visuellement la répartition, la concentration et la variabilité d'une série de données. Pour l'interpréter, il faut prendre en compte 5 données importantes : le minimum, le premier quartile, la médiane, le second quartile et le maximum. Avec toutes ces données en tête, la boîte à moustaches en dit beaucoup plus sur la répartition statistique et s'impose comme le meilleur des graphiques pour cette catégorie.

Question 6 : Vérifier la symétrie d'une distribution, c'est voir si les données sont réparties de façon équilibrée autour de la moyenne ou de la médiane. On cherche donc à savoir si la distribution des valeurs est centrée ou décentrée. Et justement, l'asymétrie ou la symétrie des données joue un rôle sur l'interprétation de celle-ci et le choix des outils graphiques que l'on va utiliser pour les représenter. Ce sont les coefficients β1 et β2 de Pearson et de Fisher qui aident à calculer la symétrie ou l'asymétrie des données.

Exercice Python :

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# Source des donnÃ©es :
https://www.data.gouv.fr/datasets/election-presidentielle-des-10-et-24-
avril-2022-resultats-definitifs-du-1er-tour/

# Sources des donnÃ©es : production de M. Forriez, 2016-2023

#Question 4

with
open("/home/thomas04dupuy/Desktop/resultats-elections-presidentielles-2
022-1er-tour.csv", "r") as fichier:
   df = pd.read_csv(fichier)

print(df.head())

#Question 5
```

```
# df =
pd.read_csv("/home/thomas04dupuy/Desktop/resultats-elections-presidenti
elles-2022-1er-tour.csv")

# SÃ©lection des colonnes quantitatives
colonnes_quantitatives = df.select_dtypes(include=["int",
"float"]).columns
df_quantitatif = df[colonnes_quantitatives]

# Calcul des moyennes avec la mÃ©thode Pandas .mean()
moyennes_series = df_quantitatif.mean(numeric_only=True)

# Conversion en liste [(nom_colonne, moyenne), ...]
liste_moyennes = [(col, float(moy)) for col, moy in
moyennes_series.items()]

# Affichage dans le terminal
print("Moyennes des colonnes quantitatives :")
for col, moyenne in liste_moyennes:
   print(f"- {col} : {moyenne}")

colonnes_quantitatives = df.select_dtypes(include=["int",
"float"]).columns
# Calcul des mÃ©dianes avec la mÃ©thode Pandas .median()
medians_series = df_quantitatif.median(numeric_only=True)

# Conversion en liste [(nom_colonne, mÃ©diane), ...]
liste_medianes = [(col, float(med)) for col, med in
medians_series.items()]

# Affichage du rÃ©sultat dans le terminal
print("MÃ©dianes des colonnes quantitatives :")
for col, med in liste_medianes:
   print(f"- {col} : {med}")

# SÃ©lection des colonnes quantitatives
colonnes_quantitatives = df.select_dtypes(include=["int",
"float"]).columns
df_quantitatif = df[colonnes_quantitatives]

# df =
pd.read_csv("/home/thomas04dupuy/Desktop/resultats-elections-presidenti
elles-2022-1er-tour.csv")
```

```
# SÃ©lection des colonnes quantitatives (nombres uniquement)
colonnes_quantitatives = df.select_dtypes(include=["int",
"float"]).columns
df_quantitatif = df[colonnes_quantitatives]

# Calcul des mÃ©dianes avec la mÃ©thode Pandas .median()
medians_series = df_quantitatif.median(numeric_only=True)

# Conversion en liste [(nom_colonne, mÃ©diane), ...]
liste_medianes = [(col, float(med)) for col, med in
medians_series.items()]

# Affichage du rÃ©sultat dans le terminal
print("MÃ©dianes des colonnes quantitatives :")
for col, med in liste_medianes:
   print(f"- {col} : {med}")

# df =
pd.read_csv("/home/thomas04dupuy/Desktop/resultats-elections-presidenti
elles-2022-1er-tour.csv")

# SÃ©lection des colonnes quantitatives (numÃ©riques)
colonnes_quantitatives = df.select_dtypes(include=["int",
"float"]).columns
df_quantitatif = df[colonnes_quantitatives]

# Calcul de l'Ã©cart-type avec la mÃ©thode Pandas .std()
ecarts_type_series = df_quantitatif.std(numeric_only=True)

# Conversion en liste [(nom_colonne, ecart_type), ...]
liste_ecarts_type = [(col, float(std)) for col, std in
ecarts_type_series.items()]

# Affichage du rÃ©sultat dans le terminal
print("Ã‰cart-type des colonnes quantitatives :")
for col, ecart in liste_ecarts_type:
   print(f"- {col} : {ecart}")

# df =
pd.read_csv("/home/thomas04dupuy/Desktop/resultats-elections-presidenti
elles-2022-1er-tour.csv")
```

```
# SÃ©lection des colonnes quantitatives
colonnes_quantitatives = df.select_dtypes(include=["int",
"float"]).columns
df_quantitatif = df[colonnes_quantitatives]

# Calcul manuel de l'Ã©cart absolu Ã  la moyenne
# (Ã©quivalent Ã  l'ancienne mÃ©thode .mad())
ecarts_absolus = df_quantitatif.apply(lambda col: np.mean(np.abs(col -
col.mean())))

# Conversion en liste [(colonne, valeur)]
liste_ecarts_absolus = [(col, float(val)) for col, val in
ecarts_absolus.items()]

# Affichage
print("Ã‰cart absolu Ã  la moyenne (Ã©cart moyen absolu) de chaque
colonne :")
for col, ecart in liste_ecarts_absolus:
    print(f"- {col} : {ecart}")

colonnes_quantitatives = df.select_dtypes(include=["int",
"float"]).columns
# Calcul de l'Ã©tendue : max - min
etendues = df_quantitatif.max(numeric_only=True) -
df_quantitatif.min(numeric_only=True)

# Conversion en liste [(nom_colonne, etendue), ...]
liste_etendues = [(col, float(et)) for col, et in etendues.items()]

# Affichage des rÃ©sultats
print("Ã‰tendue des colonnes quantitatives :")
for col, et in liste_etendues:
    print(f"- {col} : {et}")



#Question 6
df =
pd.read_csv("/home/thomas04dupuy/Desktop/resultats-elections-presidenti
elles-2022-1er-tour.csv")

# === SÃ©lection des colonnes quantitatives ===
colonnes_quantitatives = df.select_dtypes(include=["int",
"float"]).columns
```

```
df_quantitatif = df[colonnes_quantitatives]

# === Calculs statistiques ===
moyennes = df_quantitatif.mean()
medianes = df_quantitatif.median()
variances = df_quantitatif.var()
ecarts_type = df_quantitatif.std()
etendues = df_quantitatif.max() - df_quantitatif.min()
ecarts_absolus = df_quantitatif.apply(lambda col: np.mean(np.abs(col -
col.mean())))

# Mode â†’ nÃ©cessite une petite prÃ©caution
modes = df_quantitatif.mode().iloc[0]  # on prend le premier mode s'il
y en a plusieurs

# === Affichage sur le terminal ===
print("\n=== ParamÃ¨tres statistiques des colonnes quantitatives
===\n")

for col in colonnes_quantitatives:
   print(f"â–¶ {col}")
   print(f"   Moyenne : {moyennes[col]:.2f}")
   print(f"   MÃ©diane : {medianes[col]:.2f}")
   print(f"   Mode : {modes[col]:.2f}")
   print(f"   Variance : {variances[col]:.2f}")
   print(f"   Ã‰cart-type : {ecarts_type[col]:.2f}")
   print(f"   Ã‰tendue : {etendues[col]:.2f}")
   print(f"   Ã‰cart absolu Ã  la moyenne : {ecarts_absolus[col]:.2f}")
   print("-" * 50)

#Question 7
# df =
pd.read_csv("/home/thomas04dupuy/Desktop/resultats-elections-presidenti
elles-2022-1er-tour.csv")

# SÃ©lectionner les colonnes quantitatives
colonnes_quantitatives = df.select_dtypes(include=["int",
"float"]).columns
df_quantitatif = df[colonnes_quantitatives]

# Calcul des quartiles et dÃ©ciles
q1 = df_quantitatif.quantile(0.25)
q3 = df_quantitatif.quantile(0.75)
```

```
d1 = df_quantitatif.quantile(0.1)
d9 = df_quantitatif.quantile(0.9)

# Distance interquartile (IQR)
iqr = q3 - q1

# Distance interdÃ©cile (IDR)
idr = d9 - d1

# Conversion en listes [(colonne, valeur)]
liste_iqr = [(col, float(val)) for col, val in iqr.items()]
liste_idr = [(col, float(val)) for col, val in idr.items()]

# Affichage
print("Distance interquartile (IQR) :")
for col, val in liste_iqr:
   print(f"- {col} : {val}")

print("\nDistance interdÃ©cile (IDR) :")
for col, val in liste_idr:
   print(f"- {col} : {val}")

#Question 8
import os
import re
import pandas as pd
import matplotlib.pyplot as plt

# === Charger le fichier CSV ===
df =
pd.read_csv("/home/thomas04dupuy/Desktop/resultats-elections-presidenti
elles-2022-1er-tour.csv")

# === SÃ©lectionner les colonnes quantitatives ===
colonnes_quantitatives = df.select_dtypes(include=["int",
"float"]).columns

# === CrÃ©er le dossier de sortie ===
OUT_DIR = "img"
os.makedirs(OUT_DIR, exist_ok=True)

# === Fonction pour nettoyer les noms de colonnes ===
def slugify(s: str) -> str:
```

```
    s = (s or "").strip().lower()
    s = re.sub(r"[^\w\s-]", "", s)
    s = re.sub(r"[\s_-]+", "-", s)
    return s or "colonne"

# === Boucle pour tracer chaque boÃ®te Ã  moustaches ===
for col in colonnes_quantitatives:
    serie = df[col].dropna()
    if serie.empty or len(serie) < 2:
        continue  # saute les colonnes vides ou avec une seule valeur

    plt.figure(figsize=(4, 6))
    plt.boxplot(serie, vert=True, whis=1.5, showmeans=True)
    plt.title(f"BoÃ®te Ã  moustaches â€” {col}")
    plt.ylabel(col)

    fname = f"boxplot_{slugify(col)}.png"
    out_path = os.path.join(OUT_DIR, fname)
    plt.tight_layout()
    plt.savefig(out_path, dpi=150)
    plt.close()

print(f"âœ… BoÃ®tes Ã  moustaches enregistrÃ©es dans :
{os.path.abspath(OUT_DIR)}")

#Question 10
import re
import numpy as np
import pandas as pd

# === Chemin simplifiÃ© ===
df = pd.read_csv("/home/thomas04dupuy/Desktop/island-index.csv")


candidats = [c for c in df.columns if
re.search(r"(surface|superficie|area)", c, flags=re.I)]
candidats_km = [c for c in candidats if re.search(r"km", c,
flags=re.I)]

if candidats_km:
    COL_SURF = candidats_km[0]
elif candidats:
    COL_SURF = candidats[0]
```

```
else:
    print("Colonnes disponibles :", df.columns.tolist())
    raise ValueError("Aucune colonne de surface dÃ©tectÃ©e. VÃ©rifie le
nom exact dans ton CSV.")

print(f"âœ… Colonne de surface dÃ©tectÃ©e : {COL_SURF}")


surf = (
    df[COL_SURF]
    .astype(str)
    .str.replace("\u00a0", " ", regex=False)
    .str.replace(r"\s", "", regex=True)
    .str.replace(",", ".", regex=False)
    .str.replace("km2", "", flags=re.I, regex=True)
    .str.replace("kmÂ²", "", flags=re.I, regex=True)
)
surf_num = pd.to_numeric(surf, errors="coerce")


bins   = [0, 10, 25, 50, 100, 2500, 5000, 10000, np.inf]
labels = ["]0,10]", "]10,25]", "]25,50]", "]50,100]", "]100,2500]",
"]2500,5000]", "]5000,10000]", "]10000,+â^ž["]

cats = pd.cut(surf_num, bins=bins, labels=labels, right=True,
include_lowest=False)
compte = cats.value_counts().reindex(labels, fill_value=0)

print("\n=== DÃ©nombrement des Ã®les par classe de surface (kmÂ²) ===")
for classe, n in compte.items():
    print(f"- {classe} : {n}")


df["Classe_Surface_km2"] = cats
```

Séance 4. Les distributions statistiques

Questions de cour :

Question 1. Je dégage 3 critères pour choisir entre une distribution statistique avec variables discrètes ou avec variables continues :

- Il faut d'abord regarder la nature du phénomène étudié. Si celui-ci est dénombrable et distinct, alors on choisira la distribution statistique avec des variables discrètes.
- En interprétant les principales caractéristiques des données fournies. Veut-on lisser les données (variables continues) ou bien veut-on montrer une fréquence absolue (variables discrètes) ?
- À la quantité de paramètres des lois ayant forcément une influence sur celle que l'on va choisir pour notre distribution statistique.

Question 2. Selon moi, on retrouve deux grandes lois mathématiques très fréquemment utilisées en géographie pour aider au traitement statistique à grande échelle. La première est la loi de Zipf Mandelbrot car, en mettant en relation le classement des unités et leur décroissement au fil de ce rang, cette loi nous permet de modéliser des systèmes régionaux tout en tenant compte de la diversité des villes. La loi simple de Zipf permet quant à elle de détecter les cas de croissance macrocéphalique et donc d'y palier si la croissance est trop concentrée.

Deuxièmement, la loi normale ou courbe de Gauss est aussi très utilisée en géographie car énormément de phénomènes suivent une distribution normale ou quasi-normale. Elle permet donc d'analyser la distribution spatiale, elle mesure la probabilité d'un événement, ou encore d'établir des classes ou intervalles statistiques.

Exercice Python :

```python
#La Loi de Dirac
import numpy as np
import matplotlib.pyplot as plt

# === Fonction pour tracer une loi de Dirac ===
def plot_dirac(a=0.0, weight=1.0):
    """
    Visualise une loi de Dirac centrÃ©e en a (masse ponctuelle).
    a : position du pic
    weight : hauteur de la masse (probabilitÃ© totale)
    """
    # Axe des x autour du point a
    x_min, x_max = a - 3, a + 3
    xs = np.linspace(x_min, x_max, 400)

    # CrÃ©ation de la figure
    plt.figure(figsize=(7, 4))
     plt.plot(xs, np.zeros_like(xs), color="black", linewidth=1)   # axe
horizontal

    # Tige (stem) reprÃ©sentant la masse ponctuelle
    plt.stem([a], [weight], linefmt='r-', markerfmt='ro', basefmt='k-')

    # Titres et mise en forme
    plt.title(f"Loi de Dirac centrÃ©e en a = {a}")
    plt.xlabel("x")
    plt.ylabel("ProbabilitÃ© (masse)")
    plt.xlim(x_min, x_max)
    plt.ylim(0, max(1.1 * weight, 1))
    plt.grid(True, linestyle="--", alpha=0.6)
    plt.tight_layout()

    # Affiche le graphique
    plt.show()

#La loi uniforme discrÃ¨te
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import uniform, norm


# === Fonction pour tracer une loi uniforme ===
```

```
def plot_uniform(a=0, b=1):
    """
    Visualise la loi uniforme U(a, b)
    a : borne infÃ©rieure
    b : borne supÃ©rieure
    """
    x = np.linspace(a - (b - a) * 0.2, b + (b - a) * 0.2, 400)
    y = uniform.pdf(x, loc=a, scale=b - a)

    plt.figure(figsize=(7, 4))
    plt.plot(x, y, color='green', linewidth=2)
    plt.fill_between(x, y, color='lightgreen', alpha=0.4)
    plt.title(f"Loi uniforme U({a}, {b})")
    plt.xlabel("x")
    plt.ylabel("DensitÃ© de probabilitÃ©")
    plt.grid(True, linestyle="--", alpha=0.6)
    plt.tight_layout()
    plt.show()


# === Fonction pour tracer une loi normale ===
def plot_normale(mu=0, sigma=1):
    """
    Visualise la loi normale N(mu, sigmaÂ²)
    mu : moyenne
    sigma : Ã©cart-type
    """
    x = np.linspace(mu - 4 * sigma, mu + 4 * sigma, 400)
    y = norm.pdf(x, mu, sigma)

    plt.figure(figsize=(7, 4))
    plt.plot(x, y, color='blue', linewidth=2)
    plt.fill_between(x, y, color='skyblue', alpha=0.4)
    plt.title(f"Loi normale N({mu}, {sigma}Â²)")
    plt.xlabel("x")
    plt.ylabel("DensitÃ© de probabilitÃ©")
    plt.grid(True, linestyle="--", alpha=0.6)
    plt.tight_layout()
    plt.show()


    #La loi binomiale
```

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import binom

def plot_binomiale(n=10, p=0.5):
    """
    Visualise la loi binomiale B(n, p)
    n : nombre d'expÃ©riences
    p : probabilitÃ© de succÃ¨s Ã  chaque essai
    """
    # Axe des abscisses (valeurs possibles : 0, 1, 2, ..., n)
    x = np.arange(0, n + 1)
    # ProbabilitÃ©s associÃ©es (fonction de masse)
    y = binom.pmf(x, n, p)

    # CrÃ©ation du graphique
    plt.figure(figsize=(7, 4))
    plt.stem(x, y, linefmt='r-', markerfmt='ro', basefmt='k-')
    plt.title(f"Loi binomiale B({n}, {p})")  # â†  ligne corrigÃ©e
    plt.xlabel("Nombre de succÃ¨s (k)")
    plt.ylabel("ProbabilitÃ© P(X = k)")
    plt.grid(True, linestyle="--", alpha=0.6)
    plt.tight_layout()
    plt.show()

#La loi de Poisson
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import poisson


# === Fonction pour tracer une loi de Poisson ===
def plot_poisson(lam=3):
    """
    Visualise la loi de Poisson P(Î»)
       lam : valeur moyenne (Î»), c'est-Ã -dire le nombre moyen
d'Ã©vÃ©nements
    """
    # Axe des abscisses (valeurs discrÃ¨tes possibles)
    x = np.arange(0, lam * 3 + 1)
    # ProbabilitÃ©s associÃ©es (fonction de masse de probabilitÃ©)
    y = poisson.pmf(x, lam)
```

```
    # CrÃ©ation du graphique
    plt.figure(figsize=(7, 4))
    plt.stem(x, y, linefmt='b-', markerfmt='bo', basefmt='k-')
    plt.title(f"Loi de Poisson P(Î» = {lam})")
    plt.xlabe

#La loi Zipf-Mandelbrot
import numpy as np
import matplotlib.pyplot as plt


def zipf_mandelbrot_pmf(N=50, s=1.1, q=1.0):
    """
    Calcule la PMF (probabilitÃ©) de la loi de Zipfâ€“Mandelbrot sur les
rangs 1..N.
    P(k) = C / (k + q)^s  avec normalisation  sum_k P(k) = 1
    ParamÃ¨tres
    ----------
    N : int
        Nombre total d'objets/rangs (support fini).
    s : float
        Exposant (> 1 recommandÃ© pour une somme bien comportÃ©e).
    q : float
        DÃ©calage (>= 0). q=0 redonne la Zipf "pure".
    Retour
    ------
    k : np.ndarray shape (N,)
        Rangs 1..N
    p : np.ndarray shape (N,)
        ProbabilitÃ©s normalisÃ©es associÃ©es Ã  chaque rang.
    """
    k = np.arange(1, N + 1, dtype=float)
    weights = 1.0 / np.power(k + q, s)
    Z = weights.sum()  # constante de normalisation
    p = weights / Z
    return k, p


def plot_zipf_mandelbrot(N=50, s=1.1, q=1.0, loglog=True):
    """
    Trace la loi de Zipfâ€“Mandelbrot :
      - en barres (linÃ©aire)
```

```
        - en log-log (rang-frÃ©quence), pratique pour vÃ©rifier la
dÃ©croissance en loi de puissance
    """
    k, p = zipf_mandelbrot_pmf(N=N, s=s, q=q)

    # --- Graphique 1 : barres (linÃ©aire)
    plt.figure(figsize=(7, 4))
    plt.bar(k, p)
    plt.xlabel("Rang k")
    plt.ylabel("ProbabilitÃ© P(k)")
    plt.title(f"Loi de Zipfâ€“Mandelbrot (N={N}, s={s}, q={q})")
    plt.tight_layout()

    # --- Graphique 2 : log-log (optionnel)
    if loglog:
        plt.figure(figsize=(7, 4))
        plt.loglog(k, p, marker="o", linestyle="none")
        # Ligne guide ~ (k+q)^(-s) (proportionnelle), pour visualisation
            # On ajuste une droite en log-log pour montrer la pente
approchÃ©e -s
        coeffs = np.polyfit(np.log10(k + q), np.log10(p), 1)
        pente = coeffs[0]
             # Courbe de rÃ©fÃ©rence (reconstruite Ã  partir de la
rÃ©gression)
        pref_line = 10 ** (coeffs[1]) * (k + q) ** pente
        plt.loglog(k, pref_line, linewidth=1)

        plt.xlabel("log10(rang k+q)")
        plt.ylabel("log10(P(k))")
                plt.title(f"Zipfâ€“Mandelbrot (log-log) â€¢ pente â‰ˆ
{pente:.2f}")
        plt.tight_layout()

plt.savefig("zipf-mandelbrot.png", dpi=150)
plt.close()

if __name__ == "__main__":
    # Exemples rapides : ajuste N, s, q selon ton cas
    plot_zipf_mandelbrot(N=50, s=1.1, q=1.0)
    # plot_zipf_mandelbrot(N=100, s=1.3, q=2.0)


#Loi Poisson
```

```
import numpy as np
import matplotlib.pyplot as plt

# ParamÃ¨tre lambda (taux moyen dâ€™Ã©vÃ©nements)
lam = 2.0

# Domaine continu pour t â‰¥ 0
t = np.linspace(0, 5, 200)
# DensitÃ© de probabilitÃ© (loi exponentielle associÃ©e au processus de
Poisson)
f = lam * np.exp(-lam * t)

# Graphique
plt.figure(figsize=(7, 4))
plt.plot(t, f, color="darkorange", lw=2)
plt.title(f"Distribution continue associÃ©e Ã  la loi de Poisson (Î» =
{lam})")
plt.xlabel("Temps entre deux Ã©vÃ©nements (t)")
plt.ylabel("DensitÃ© de probabilitÃ© f(t)")
plt.grid(True, linestyle="--", alpha=0.6)
plt.tight_layout()
plt.savefig("poisson_continue.png", dpi=150)
plt.close()

#La loi normale
import numpy as np
import matplotlib.pyplot as plt
from math import sqrt, pi, exp

# ParamÃ¨tres de la loi normale
mu = 0       # moyenne
sigma = 1    # Ã©cart type

# Domaine des valeurs continues
x = np.linspace(mu - 4*sigma, mu + 4*sigma, 400)
# DensitÃ© de probabilitÃ© f(x)
f = (1 / (sigma * sqrt(2 * pi))) * np.exp(-((x - mu)**2) / (2 *
sigma**2))

# CrÃ©ation du graphique
plt.figure(figsize=(7, 4))
plt.plot(x, f, color="royalblue", lw=2)
plt.title(f"Loi normale (Î¼ = {mu}, Ïƒ = {sigma})")
```

```
plt.xlabel("Valeurs de la variable X")
plt.ylabel("DensitÃ© de probabilitÃ© f(x)")
plt.grid(True, linestyle="--", alpha=0.6)
plt.tight_layout()

# Enregistrer l'image (au lieu de plt.show())
plt.savefig("loi_normale.png", dpi=150)
plt.close()

print("â… Image enregistrÃ©e : loi_normale.png")

#la Loi log-normale
import numpy as np
import matplotlib.pyplot as plt
from math import sqrt, pi, exp

# === ParamÃ¨tres de la loi log-normale ===
mu = 0       # moyenne du logarithme
sigma = 0.5  # Ã©cart type du logarithme

# Domaine des valeurs (X > 0)
x = np.linspace(0.001, 5, 400)

# DensitÃ© de probabilitÃ© (formule de la loi log-normale)
f = (1 / (x * sigma * sqrt(2 * pi))) * np.exp(-((np.log(x) - mu)**2) /
(2 * sigma**2))

# === CrÃ©ation du graphique ===
plt.figure(figsize=(7, 4))
plt.plot(x, f, color="seagreen", lw=2)
plt.title(f"Loi log-normale (Î¼ = {mu}, Ïƒ = {sigma})")
plt.xlabel("Valeur de la variable X (>0)")
plt.ylabel("DensitÃ© de probabilitÃ© f(x)")
plt.grid(True, linestyle="--", alpha=0.6)
plt.tight_layout()

# === Sauvegarde du graphique (au lieu de plt.show()) ===
plt.savefig("loi_log_normale.png", dpi=150)
plt.close()

print("â… Image enregistrÃ©e : loi_log_normale.png")

#Loi uniforme
```

```
import numpy as np
import matplotlib.pyplot as plt

# === ParamÃ¨tres de la loi uniforme ===
a = 0   # borne infÃ©rieure
b = 10  # borne supÃ©rieure

# Domaine des valeurs continues
x = np.linspace(a - 1, b + 1, 400)

# DensitÃ© de probabilitÃ© f(x)
f = np.where((x >= a) & (x <= b), 1 / (b - a), 0)

# === CrÃ©ation du graphique ===
plt.figure(figsize=(7, 4))
plt.plot(x, f, color="orange", lw=2)
plt.fill_between(x, f, color="orange", alpha=0.3)
plt.title(f"Loi uniforme continue sur [{a}, {b}]")
plt.xlabel("Valeur de la variable X")
plt.ylabel("DensitÃ© de probabilitÃ© f(x)")
plt.grid(True, linestyle="--", alpha=0.6)
plt.tight_layout()

# === Sauvegarde du graphique ===
plt.savefig("loi_uniforme.png", dpi=150)
plt.close()

print("â… Image enregistrÃ©e : loi_uniforme.png")

#loi du xÂ²
import numpy as np
import matplotlib.pyplot as plt
from math import gamma, sqrt, pi, exp

# === ParamÃ¨tre de la loi du Ï‡Â² ===
k = 5   # degrÃ©s de libertÃ© (tu peux changer la valeur : 1, 2, 10,
...)

# Domaine des valeurs continues (X >= 0)
x = np.linspace(0, 30, 400)

# === Fonction de densitÃ© de probabilitÃ© ===
```

```
f = (1 / (2 ** (k / 2) * gamma(k / 2))) * (x ** (k / 2 - 1)) *
np.exp(-x / 2)

# === CrÃ©ation du graphique ===
plt.figure(figsize=(7, 4))
plt.plot(x, f, color="crimson", lw=2)
plt.title(f"Loi du khi-deux (Ï‡Â²) avec k = {k} degrÃ©s de libertÃ©")
plt.xlabel("Valeur de la variable X (â‰¥ 0)")
plt.ylabel("DensitÃ© de probabilitÃ© f(x)")
plt.grid(True, linestyle="--", alpha=0.6)
plt.tight_layout()

# === Sauvegarde du graphique ===
plt.savefig("loi_chi2.png", dpi=150)
plt.close()

print("âœ… Image enregistrÃ©e : loi_chi2.png")



#Loi de Pareto
import numpy as np
import matplotlib.pyplot as plt

# === ParamÃ¨tres de la loi de Pareto ===
alpha = 2.5    # paramÃ¨tre de forme (plus il est grand, plus la queue
est courte)
xm = 1.0       # valeur minimale (borne de dÃ©part)

# Domaine des valeurs continues (x >= xm)
x = np.linspace(xm, 10, 400)

# === Fonction de densitÃ© de probabilitÃ© (PDF) ===
f = alpha * (xm ** alpha) / (x ** (alpha + 1))

# === CrÃ©ation du graphique ===
plt.figure(figsize=(7, 4))
plt.plot(x, f, color="purple", lw=2)
plt.fill_between(x, f, color="purple", alpha=0.3)
plt.title(f"Loi de Pareto (Î± = {alpha}, xâ‚˜ = {xm})")
plt.xlabel("Valeur de la variable X (â‰¥ xâ‚˜)")
plt.ylabel("DensitÃ© de probabilitÃ© f(x)")
plt.grid(True, linestyle="--", alpha=0.6)
plt.tight_layout()
```

```
# === Sauvegarde du graphique ===
plt.savefig("loi_pareto.png", dpi=150)
plt.close()

print("â… Image enregistrÃ©e : loi_pareto.png")

#Question 2
import math

# === 1ï¸ âƒ£ Fonction : moyenne (espÃ©rance mathÃ©matique) ===
def moyenne(distribution, *params):
    """
    Calcule la moyenne (espÃ©rance) d'une loi donnÃ©e.
      distribution : nom de la loi ("dirac", "uniforme", "normale",
"binomiale", "poisson", "pareto")
    *params : paramÃ¨tres selon la loi
    """
    if distribution == "dirac":
        # Loi de Dirac centrÃ©e sur une valeur a
        a, = params
        return a

    elif distribution == "uniforme":
        a, b = params
        return (a + b) / 2

    elif distribution == "normale":
        mu, sigma = params
        return mu

    elif distribution == "binomiale":
        n, p = params
        return n * p

    elif distribution == "poisson":
        lam, = params
        return lam

    elif distribution == "pareto":
        alpha, xm = params
        if alpha > 1:
            return (alpha * xm) / (alpha - 1)
```

```
        else:
            return math.inf  # moyenne non dÃ©finie si Î± â‰¤ 1

    else:
         raise ValueError("Loi inconnue. Choisis parmi : dirac, uniforme,
normale, binomiale, poisson, pareto.")


# === 2ï¸âƒ£ Fonction : Ã©cart type (Ïƒ) ===
def ecart_type(distribution, *params):
    """
    Calcule l'Ã©cart type d'une loi donnÃ©e.
    """
    if distribution == "dirac":
        return 0

    elif distribution == "uniforme":
        a, b = params
        return (b - a) / math.sqrt(12)

    elif distribution == "normale":
        mu, sigma = params
        return sigma

    elif distribution == "binomiale":
        n, p = params
        return math.sqrt(n * p * (1 - p))

    elif distribution == "poisson":
        lam, = params
        return math.sqrt(lam)

    elif distribution == "pareto":
        alpha, xm = params
        if alpha > 2:
             return xm * math.sqrt(alpha / ((alpha - 1) ** 2 * (alpha -
2)))
        else:
            return math.inf  # Ïƒ non dÃ©fini si Î± â‰¤ 2

    else:
         raise ValueError("Loi inconnue. Choisis parmi : dirac, uniforme,
normale, binomiale, poisson, pareto.")
```

```
# === 3ï¸ âƒ£ DÃ©monstration ===
if __name__ == "__main__":
    lois = [
        ("dirac", (5,)),
        ("uniforme", (0, 10)),
        ("normale", (0, 2)),
        ("binomiale", (20, 0.3)),
        ("poisson", (4,)),
        ("pareto", (3, 1))
    ]

    print("=== Moyenne et Ã©cart type pour chaque loi ===")
    for nom, params in lois:
        m = moyenne(nom, *params)
        s = ecart_type(nom, *params)
          print(f"Loi {nom.capitalize()} {params} â†’ Moyenne = {m:.3f},
Ïƒ = {s:.3f}
```

Séance 5 : Les statistiques inférentielles

Questions de cour

Question 1 : Un échantillon est un modèle réduit de la population étudiée.udi chantillon est dit représentatif si sa structure (sociale, économique, géographique) reflète celle de la population de rérencer On ne peut souvent pas étudier toute la population, car cela coûterait trop cher, prendrait trop de temps ou serait matériellement impossible (ex : recenser toute une ville chaque semaine).

Question 2 : Un estimateur est une formule ou une règle statistique qui permet d'estimer un paramètre inconnu (comme la moyenne réelle). L'estimation est la valeur numérique concrète obtenue à partir de cet estimateur appliqué à l'échantillon.

Question 3 : L'intervalle de fluctuation décrit l'intervalle de valeur compris entre une borne supérieure et une borne inférieure de valeurs que peut prendre une proportion ou une moyenne d'échantillon dès lors que l'on lequel plusieurs fois. L'intervalle de confiance indique dans quelle interval de valeurs se trouve le vrai paramètre de la population avec une certaine probabilité de 95%

Question 4 : Un biais est une erreur systématique qui fait qu'un estimateur ne donne pas, en moyenne, la vraie valeur du paramètre. Il peut venir d'un mauvais échantillonnage, ou d'un modèle mal adapté.

Question 9 : La statistique inférentielle est une démarche statistique consistant à extrapoler à une population entière les propriétés mises en évidence sur un ensemble d'individus enquêtés, appelé échantillon. De fait les critiques qui lui sont adressées sont plutôt léadresséesen tant que llégitimes clairement remettre en cause l'extrapolation de résultat qui sont justrésultatsn échantiljustesmais qui ne peuvent l'être pleinement juste en élargissant llation étudiée

Exercice de code

Théorie de l'échantillonnage

Question 1 La moyenne du nombre de personnes ayant répondu "Pour" est de 391. Le nombre de personnes ayant répondu "Contre" est de 416 et enfin le nombre de personne ayant répondu opinion est de 193

Question 2

Pour les fréquences cf Excel

Quest, cf. 3

Idem pour l'intervalle de fluctuation cf Excel

Quest, cf. 4

L'intervalle de fluctuation que nous avons calculé avec les moyennes et les effectifs de les échantillons a des de vérifier la représentativité des échantillon si on les échantillons l'ensemble de la population mère. Si la fréquence de l'échantillon se trouve dans l'intervalle de fluctuation, alors on peut en conclure que l'échantillon est bel et bien représentatif et donc que notre raisonnement est cohérent. Le cas échéant, il nous faudrait reconsidérer la population étudiée. En clair, il nous permet de valider ou d'infirmer le tirage effectué initialement.

Théorie de l’estimation

Question 1 : La somme de la ligne est égale à 1000 et les fréquences sont à consulter directement sur l’excel

Question 2 : Pour les fréquences cf Excel

Question 3 : Idem pour l’intervalle de fluctuation cf Excel

Pour	Contre	Sans opinion	
395	396	209	1000
379	432	189	
384	426	190	
395	407	198	
389	413	198	
386	420	194	
371	434	195	
394	397	209	
388	420	192	
401	403	196	
394	409	197	
374	421	205	
395	418	187	
372	431	197	
385	425	190	
397	413	190	
385	431	184	
383	417	200	
399	404	197	
389	412	199	
395	426	179	
394	418	188	
377	434	189	
391	405	204	
397	420	183	
395	419	186	
405	401	194	
409	405	186	
390	418	192	
383	416	201	
394	416	190	
391	415	194	
413	413	174	
384	442	174	
398	411	191	
368	442	190	
387	415	198	
411	396	193	
410	403	187	
420	387	193	
399	414	187	
383	422	195	
405	403	192	
395	409	196	
384	420	196	
400	422	178	
373	442	185	
395	418	187	
384	427	189	

Théorie de l'échantillon			Borne inférieure	Borne supérieure	
Moyenne de "Pour"	390,52	391			
Moyenne de "Contre"	416,06	416			
Moyenne de "Sans opinion"	193,42	193			
Fréquence de "Pour"	0,391	0,39		0,29	0,49
Fréquence de "Contre"	0,41606	0,42		0,32	0,52
Fréquence de "Sans opinion"	0,19342	0,19		0,11	0,27
Fréquence de la population mère "Pour"	0,000178947	0,1787		0,1	0,25
Fréquence de la population mère "Contre"	0,190389016	0,19		0,11	0,27
Fréquence de la population mère "Sans opinion"	0,088329519	0,09		0,03	0,15
Théorie de l'estimation			Intervalle de confiance		
Fréquence de "Pour"	0,395	0,4		0,03	
Fréquence de "Contre"	0,396	0,4		0,03	
Fréquence de "Sans opinion"	0,209	0,21		0,03	

Théorie de décision

```
import pandas as pd
from scipy.stats import shapiro

# Chemins locaux
df1 = pd.read_csv("/home/thomas04dupuy/Desktop/Loi-normale-Test-1
(1).csv")
df2 = pd.read_csv("/home/thomas04dupuy/Desktop/Loi-normale-Test-2.csv")

# Test de Shapiro-Wilk
stat1, p1 = shapiro(df1)
stat2, p2 = shapiro(df2)

print("Test 1 â†’ Stat =", stat1, "p-value =", p1)
print("Test 2 â†’ Stat =", stat2, "p-value =", p2)

# InterprÃ©tation
if p1 > 0.05:
   print( Test 1 suit une distribution normale.")
else:
   print(Test 1 ne suit pas une distribution normale.")

if p2 > 0.05:
   print( Test 2 suit une distribution normale.")
else:
   print( Test 2 ne suit pas une distribution normale.")
```

Question 1 : Test 1 → Stat = 0.9639482021309311 p-value = 6.286744082090187e-22
Test 2 → Stat = 0.2608882349902276 p-value = 7.04938990116743e-67
❌ Test 1 ne suit pas une distribution normale.
❌ Test 2 ne suit pas une distribution normale.
Donc aucun des deux ne suit une distribution normale

Séance 6. La statistique d'ordre des variables qualitatives

Question de cours

Question 1 Une statistique ordinale est une statistique qui s'appuie sur des données ordonnées, c'est-à-dire des données que l'on peut classer selon un ordre, sans que les écarts entre les valeurs aient nécessairement un sens quantitatif précis.

Question 2 Dans les classifications statistiques et spatiales, l'ordre décroissant est à privilégier.L'ordre décroissant permet de placer les éléments les plus importants en premier.

Il facilite la lecture des hiérarchies en mettant en évidence les valeurs dominantes. Il est particulièrement adapté aux analyses de type rang-taille, où l'on étudie la relation entre la taille d'un objet spatial et son rang.

Question 3. La corrélation des rangs mesure l'intensité et le sens de la relation entre deux classements alors que la concordance de classements évalue le degré d'accord global entre plusieurs classements.

Question 4. Les tests de Spearman et de Kendall sont deux tests de corrélation non paramétriques, basés sur les rangs. Ils servent à mesurer l'existence d'une relation monotone entre deux variables ordinales ou quantitatives non normales. L'un mesure la corrélation entre deux séries de rangs et compare les différences de rang entre observations et l'autre repose sur le nombre de paires concordantes et discordantes ainsi que mesure le degré d'accord entre deux classements.

Question 5. Les coefficients de Goodman-Kruskal et de Yule servent à mesurer l'intensité de l'association entre deux variables catégorielles, à partir de tableaux de contingence. Le coefficient de Goodman-Kruskal évalue la réduction d'erreur de prédiction d'une variable par une autre, tandis que le coefficient de Yule mesure la force et le sens de l'association entre deux variables dichotomiques. Ils sont utilisés lorsque les données ne sont ni numériques ni ordinales, mais purement qualitatives.

Mise en oeuvre avec Python

Je n'ai pas pu réaliser l'exercice de cette séance car j'ai un chromebook et qu'il ne lit pas les fichier .csv. Même en ayant réussi à installer excel, celui-ci ne veut pas lire le fichier .csv pour me faire apparaître un tableur lisible avec toutes les données contenues dans le fichier .csv.

Commentaire sur les humanités numériques

Il semble bien qu'aujourd'hui les humanités numériques prennent de plus en plus de place dans la recherche scientifique. Dans un monde où tout semble de plus en plus connecté et où l'industrie de la tech s'impose comme le futur de l'humanité, cette discipline a tout intérêt à se développer au rythme des innovations technologiques qui ne cessent de faire évoluer notre quotidien. Elles sont tant d'outils précieux pour le chercheur qui, s'il ne s'adapte pas aux caractéristiques de son époque, ne peut la comprendre pleinement. Plus encore, ces outils numériques tels que nous avons appris à les manipuler dans le cours doivent servir de support à la recherche scientifique. À l'échelle du travail universitaire, cela ne fait que quelques dizaines d'années que le traitement statistique peut être réalisé dans de telles proportions. Les ordinateurs augmentant chaque année leur puissance de calcul, peut-on vraiment imaginer qu'un jour ils s'arrêtent ? Dans l'intérêt du chercheur, il ne le faudrait pas. Car plus l'ordinateur est puissant, plus le chercheur va pouvoir réaliser des études s'étendant sur une population toujours plus grande. Ainsi, les humanités numériques sont le prolongement de ce constat. Si dans un premier temps on peut trouver le terme oxymorique, humanité et numérique ne s'inscrivant pas du tout dans la même réalité, il semble bien qu'un croisement est possible. Et c'est le domaine de la recherche qui en est le théâtre car seul le chercheur peut également employer au maximum cet outil. Finalement, l'humanité numérique ne peut que se développer dans les prochaines années/décennies, et plus encore, elle peut constituer une forme d'aboutissement dans la recherche avec des traitements statistiques toujours plus importants et brassant une population toujours plus large. L'humanité, si elle cherche sa place dans le futur, la trouvera surement dans le numérique.

Pour ce qui est du cours qu'il nous a été donné de suivre ce semestre, je l'ai trouvé particulièrement mal fait. Vous ne pouvez pas demander à des étudiants en master d'investir autant de temps et de ressources dans un domaine que très peu, et j'insiste sur le très peu, de personnes n'avaient pratiqué avant. Alors vous avez surement dû créer des vocations chez des personnes qui n'avaient jamais pratiqué Python avant et qui ont adoré le faire pendant vos cours. Pour autant, l'écrasante majorité des personnes ne sont pas dans ce cas. J'entends l'ambition que vous aviez de faire découvrir un langage informatique, aussi important soit-il, à des étudiants en master de géographie. Un outil qui par ailleurs devrait se développer grandement dans les prochaines années. Pour autant, quand on connaît le profil des géographes en France, c'est-à-dire très littéraire, où la plupart d'entre nous ont arrêté les mathématiques au lycée, certains dès la Seconde, on ne va pas reprendre tous les programmes de mathématiques depuis le lycée pour comprendre une seule matière du semestre, qui plus est où la note finale est couplée à une autre matière bien plus facile à comprendre (méthode quantitative). Je n'ai assisté à aucun cours, je n'ai posé aucune question et pourtant j'ai réussi à rendre toutes les séances, ce qui montre que ce n'est pas impossible, même pour quelqu'un qui a arrêté les mathématiques en 3e. Pour autant, je ne vous cache pas mon énervement (le mot

est faible), pendant les dizaines d'heures que j'ai mises3e. dans ce compte rendu. Des dizaines d'heures que j'ai mises sans même savoir si ce que je faisais était bien puisque je n'ai jamais fait de Python. C'est peut-être le pire dans cette histoire, c'est d'y avoir mis autant de temps pour probablement avoir une note médiocre car je suis bien conscient que ce que j'ai rendu l'est. Des heures passées à comprendre votre cours, littéralement des heures, avant que je sois résigné et que j'utilise l'intelligence artificielle pour m'aider car c'était le meilleur outil que j'avais pour avancer. Je pensais bien m'y connaitre en informatique, et c'est ce qui m'a donné une légère avance sur mes camarades, mais je n'imagine pas ceux qui ont de base des difficultés avec les ordinateurs. Vous n'avez presque rien expliqué, vous nous avez donné votre cours de 600 pages et puis on était censés avoir toutes les clés pour réussir. Encore une fois, je ne remets pas en cause votre bonne volonté de nous faire réussir dans votre matière, la preuve vous avez rédigé un cours de 600 pages. Néanmoins, il faudra pour les années suivantes beaucoup plus d'accompagnement pour les élèves, revenir à la base et prendre le temps de tout expliquer. J'ai beaucoup appris, mais dans la souffrance et je pense que le cerveau est fait de telle façon que j'ai associé ces connaissances à un mauvais moment et donc je serai amené à tout oublier (ou en partie). Voilà comme je finirai ce commentaire sur le cours en lui-même, un mauvais moment. Votre bonne volonté est indéniable, maintenant à vous de revoir l'approche avec les étudiants en prenant en considération que la plupart n'y connaissent rien en informatique, et que personne ne va investir tout son temps pour une matière au coefficient secondaire par rapport à d'autres, d'autant plus que c'est une matière que personne n'a choisie et qui nous a été imposée sans notre consentement.

Voilà, j'ai peut-être été dure mais j'ai cherché à être constructive tout en vous montrant la frustration que j'ai eue pendant ce semestre.

Je n'ai pas pu intégrer les images générés par mes codages python car les ayant stockés sur mon environnement Linux de mon chromebook, au moment de rédiger leur commentaire, mon ordinateur les a supprimées de l'espace Linux. Je ne peux pas non plus ouvrir VS Code pour les générer de nouveaux.