Université Virtuelle du Burkina Faso

Sciences fondamentales

Filière : Modélisation mathématique-Calcul scientifique

Projet Tutore

Modélisation Mathématique de l'équation de la chaleur et Son implémentation Informatique dans une Base de Données Relationnelle

Membre du groupe :

TIENDREBEOGO Karim

Sawadogo Emilie

Enseignant :

M. Cheik Amed Traore

Année Universitaire : 2023 – 2024

Table des matières

Introduction

L'étude des phénomènes naturels par le biais de modélisations mathématiques constitue un outil fondamental dans de nombreuses disciplines scientifiques. Elle permet de mieux comprendre les comportements complexes des systèmes physiques et d'anticiper leur évolution dans des conditions diverses. Dans ce cadre, l'équation de la diffusion de la chaleur, également connue sous le nom d'équation de conduction thermique, représente un cas typique et essentiel. Utilisée pour décrire la propagation de la chaleur dans un milieu donné, elle trouve des applications dans des domaines variés allant de l'ingénierie thermique à la climatologie en passant par les sciences des matériaux.

Dans ce projet, nous nous proposons d'explorer l'équation de la chaleur en déployant une approche structurée en cinq parties : modélisation mathématique, génération de données simulées, conception d'une base de données relationnelle, exploitation des données, et rédaction d'un rapport scientifique. Chaque étape contribuera à une meilleure compréhension de ce phénomène tout en mettant en œuvre des compétences multidisciplinaires, allant des mathématiques appliquées à la programmation et à la gestion de bases de données.

I. Présentation du projet

I.1 Problématique

Dans de nombreux domaines scientifiques et industriels, il est crucial de modéliser des phénomènes naturels pour prédire leur comportement ou tester des scénarios hypothétiques. Cependant, une simple modélisation mathématique ne suffit pas. La simulation et le stockage des données dans une structure permettant une exploitation optimale constituent des étapes également essentielles. Ce projet vise à répondre à la problématique suivante : Comment intégrer la modélisation mathématique, la simulation et la gestion de données pour analyser un phénomène donné ? En répondant à cette question, ce projet ambitionne non seulement de fournir des outils pour comprendre la diffusion thermique mais aussi de développer une méthodologie applicable à d'autres phénomènes physiques similaires.

I.2 Objectifs

Les objectifs du projet sont :

1. Modéliser mathématiquement un phénomène naturel.
2. Générer des données à partir de simulations basées sur le modèle mathématique.
3. Concevoir une base de données relationnelle sous PostgreSQL pour stockeret structurer les données.
4. Insérer les données dans la base et réaliser des requêtes pour en extrairedes informations pertinentes.

Méthodologie

Le projet suit une approche structurée en plusieurs étapes :

1. Modélisation mathématique : formulation et résolution de l'équation de la chaleur.
2. Génération de données : écriture d'un programme pour simuler le transfert de chaleur et enregistrer les résultats.
3. Conception de base de données : création d'un schéma relationnel pour stocker les données simulées.
4. Insertion et exploitation des données : alimentation de la base de données et interrogation des données via des requêtes SQL.

II. Phase de réalisation

Modélisation Mathématique du phénomène

— Présentation du phénomène

On étudie une tige homogène de longueur *L*, suffisamment mince pour que la température soit uniforme sur toute la section transversale. Les extrémités de la tige sont maintenues à une température constante nulle. La distribution de température dans la tige est décrite par $u(x,t)$, qui satisfait l'équation de la chaleur suivante :

$$\frac{\partial u}{\partial t}(x,t) - c^2 \frac{\partial^2 u}{\partial x^2}(x,t) = 0$$

Avec les conditions aux limites :

$u(0,t)=0$, $u(L,t)=0$, et une condition initiale : $u(x,0)=f(x)$.

— Hypothèses

La tige est isolée thermiquement sur sa surface.

Les extrémités sont fixées à une température nulle.

La chaleur se propage uniquement en une dimension.

— Méthode de résolution

Pour résoudre cette équation, nous utilisons la méthode de séparation des variables en posant $u(x,t) = \phi(x)\psi(t)$. Cette approche permet de transformer l'équation en deux équations différentielles indépendantes. La solution générale est obtenue sous la forme d'une série de Fourier :

$$u(t,x) = \sum_{n=1}^{\infty} a_n sin(\tfrac{n\pi}{L}) e^{-(\frac{n\pi c}{L})^2 t}$$

où les coefficients a_n sont déterminés à partir des conditions initiales

— Interprétation de la solution

Les modes $\sin(\frac{n\pi x}{L})$ décrivent les profils spatiaux de température

Le terme exponentiel $e^{-(\frac{n\pi c}{L})^2 t}$ montre que les températures décroissent avec le temps

Simulation numérique et génération des données

Une simulation numérique a été mise en œuvre pour calculer l'évolution de la température dans le temps et l'espace. La méthode des différences finies explicite a été utilisée. Les paramètres suivants ont été choisis :

- Longueur de la tige : $L=1m$
- Nombre de points en espace : $N_x = 50$.
- Nombre de points en temps : $N_t = 1000$.

Le programme, développé en Python, génère un fichier CSV contenant les températures pour chaque position et instant.

Création de la base de données relationnelle

Une base de données relationnelle *simulation_chaleur* a été mise en place pour organiser et stocker les données générées par la simulation. Cette base est structurée autour de trois tables principales :

1. *Simulations* : Enregistre les informations générales de chaque simulation, telles que la description et la date de création. La clé primaire est id_simulation.
2. *Paramètres* : Détaille les paramètres d'entrée de la simulation (longueur de la tige, coefficient de diffusion, etc.), chaque enregistrement étant lié à une simulation via une clé étrangère id_simulation.
3. *Résultats* : Contient les températures calculées pour chaque position et instant de temps. Les résultats sont reliés à leurs paramètres associés grâce à une clé étrangère id_simulation.

Les scripts SQL nécessaires à la création de ces tables ont été développés, et les données issues de la simulation ont été insérées dans la base à l'aide d'un script Python.

Insertion des données dans la base de données

Les résultats de la simulation ont été organisés dans un fichier CSV, puis insérés dans la base PostgreSQL. Le script Python utilise la bibliothèque psycopg2 pour automatiser cette étape.

Une fois les données insérées, elles peuvent être interrogées via des requêtes SQL pour analyse.

Exploitation des données

Pour analyser les données, plusieurs requêtes SQL ont été développées. Par exemple :

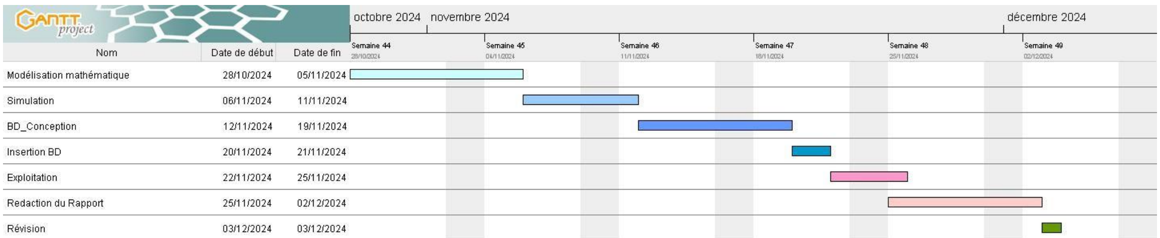
— Identifier la position de la température maximale à un instant donné.

— Calculer la température moyenne sur la tige au fil du temps. Les résultats de ces requêtes ont été exportés sous forme de tableaux ou graphiques pour une analyse plus approfondie.

En somme, La phase de réalisation a permis de modéliser et de simuler le phénomène de diffusion de la chaleur avec succès. Les résultats ont été intégrés dans une base de données relationnelle, facilitant leur exploitation. L'utilisation de requêtes SQL et de visualisations graphiques a permis de vérifier les hypothèses et de répondre aux objectifs initiaux du projet.

Planning du projet (digramme de Gant)

Le diagramme de Gantt ci-dessous illustre la planification détaillée des différentes étapes du projet, comprenant la modélisation, la simulation, et la gestion des données.



Bilan du projet

Ce projet a permis de combiner des compétences en modélisation mathématique, en programmation et en gestion de base de données. Les résultats obtenus montrent que la structure relationnelle adoptée facilite l'interrogation et l'analyse des données. Les requêtes SQL conçues ont permis de répondre à des hypothèses précises et de valider des tendances. Malgré quelques défis rencontrés, notamment dans la mise en œuvre de la base de donnés, le projet a globalement atteint ses objectifs.

Conclusion

Ce projet illustre l'utilité des outils de modélisation mathématique combinés à des systèmes relationnels pour le traitement et l'analyse de données issues de simulations scientifiques.

Les étapes de modélisation, de simulation et de gestion des données ont démontré l'importance de la collaboration entre disciplines pour résoudre des problématiques complexes. L'intégration réussie des résultats dans une base relationnelle, associée à des analyses pertinentes via SQL, montre une approche solide et extensible.

En somme, ce projet ouvre la voie à des applications plus complexes et variées dans des domaines tels que la physique, la biologie et l'ingénierie. Les compétences acquises tout au long de ce projet offrent une base solide pour aborder des problèmes similaires à l'avenir.

Annexe

Résolution de l'équation de la chaleur par la méthode de la séparation des variables

Soit une tige homogène de longueur *L* suffisamment mince de façon que la chaleur soit distribuée également sur toute la section transversale : la surface de cette dernière est isolée donc il n'y a donc pas aucune perte de chaleur à travers la surface de la tige. Si nous notons par *u(t,x)* la température dans la tige au temps *t* au point *x* alors u(t,x) satisfait à l'équation de la chaleur :

$$u :(x,t) \rightarrow u(x,t),\ \forall\, x \in \mathrm{R}$$

$$\frac{\partial u}{\partial t}(x,t) - c^2 \frac{\partial^2 u}{\partial x^2}(x,t) = 0$$

- Condition aux limites

La tige étant isolé et fixé au deux (2) extrémités la température x = 0 et x=L est maintenue constante à 0

D'où $u(0,L)=0\ ;\ u(L,t)=0\ ,\ \forall t \geq 0$

- Condition initiale

Il y'a pas d'échange de chaleur de chaleur aux extrémités.

- Domaine de définition

Pour t = 0 la température de la tige est donné par une distribution initial *f(x) :*

$$u(x,0) = f(x)\ \forall\, x \in [0,L]$$

Pour $\forall\, x \in [0, L],\ t \geq 0$ on *u(t, x)* Par regroupement

notre équation devient :

$$\begin{cases} u:(x,t)\mapsto u(x,t) & \forall x\in\mathbb{R},\ \forall t\in\mathbb{R}^+ \\ \frac{\partial u}{\partial t}(x,t)-c^2\frac{\partial^2 u}{\partial x^2}(x,t)=0 & \\ u(0,t)=0, & u(L,t)=0 \\ u(x,0)=f(x). & \end{cases}$$

Nous allons procéder par la méthode des séparations des variables pour la résolution de l'équation

Posons $u(t, x) = \phi(x)\psi(t)$,

$$u(t, x) = \frac{\phi''(x)}{\phi(x)} = \frac{1}{c^2}\frac{\psi'(t)}{\psi(t)}$$

le membre de gauche et de droite, chacun ne dépend pas de l'autre donc on peut écrire :

$$\phi''(x) - \lambda\phi = 0, (1)$$

$$\psi'(t) - \lambda c^2\psi(t) = 0, (2)$$

— En fonction de λ la solution de la première équation s'écrit :

$$\phi(x) = \begin{array}{c} A\exp^{\sqrt{\lambda}x} + B\exp^{\sqrt{-\lambda}x}, si\lambda > 0 \\ Acos(\sqrt{-\lambda}x + Bsin(\sqrt{-\lambda}x), si\lambda < 0 \\ Ax + b, si\lambda = 0 \end{array}$$

La seule solution non nulle est quand $\lambda > 0$,

$$\lambda = -(\frac{n\pi}{L})^2 \text{ et } \phi_n(x) = sin(\frac{n\pi}{L}x), n > 0$$

donc :

$$< \phi_n(x), \phi_m(x) >= \left\{ \begin{array}{l} \frac{L}{2}, sin \neq m \\ 0, sin = m \end{array} \right\}$$

— Pour la deuxième équation :

$$\psi\ (t) - \lambda_n c^2\psi(t) = 0$$

$$\psi'(t) = c_n\exp(\lambda_n c^2 t) = c_n\exp(-\frac{n\pi c^{\acute{e}}}{L}t)$$

— Par conséquent;

$$u(t,x) = \sum_{n=1}^{\infty} a_n sin(\frac{n\pi}{L})e^{-\frac{n\pi c}{2}t}$$

Avec $a_n = \frac{2}{L}\int_0^L sin(\frac{n\pi x}{L})dx\ \forall n \geq 1$

Génération des données

Pour générer les données numériques, nous utiliserons la méthode des différences finies (schéma explicite) . Cette méthode discrétise l'espace *x* et le temps *t*.

- Diviser x en N_x points avec un pas Δx
- Diviser t en N_t points avec un pas Δt

- le schéma explicite donne :

$$u_i^{n+1} = u_i^n + \frac{\alpha \Delta t}{\Delta x^2}(u_{i-1}^n - 2u_i^n + u_{i+1}^n)$$

où $u^n{}_i$ est la température au point x_i et au pas de temps n code python pour générer les données

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

# Paramètres de la simulation

L = 10.0 # Longueur de la barre (m)

T = 5.0 # Durée de la simulation (s)

Nx = 100 # Nombre de points de discrétisation spatiale
Nt = 1000 # Nombre de points de discrétisation temporelle
alpha = 0.01 # Coefficient de diffusion thermique (m^2/s)
# Discrétisation spatiale et temporelle

dx = L / (Nx - 1)

dt = T / Nt

# Stabilité numérique (condition CFL)

 if alpha * dt / dx**2 > 0.5 :

raise ValueError("Condition de stabilité CFL non respectée : réduisez dt ou
augmentez dx.")

# Initialisation des matrices
 x = np.linspace(0, L, Nx) # Coordonnées spatiales
 t = np.linspace(0, T, Nt) # Temps
u = np.zeros((Nt, Nx)) # Matrice des températures
 # Condition initiale : exemple d'une distribution sinusoidale u[0, :] =
np.sin(np.pi * x / L)
# Conditions aux limites : Dirichlet
 u[ :, 0] = 0 # Bord gauche
 u[ :, -1] = 0 # Bord droit
# Boucle de simulation (méthode explicite)
for n in range(0, Nt - 1) :
for i in range(1, Nx - 1) :
u[n + 1, i] = u[n, i] + alpha * dt / dx**2 * (u[n, i + 1] - 2 * u[n, i] + u[n, i- 1])
```

Sauvegarde des données

np.savetxt("temperature_data.csv", u, delimiter=",") print("Simulation terminée. Les données sont enregistrées dans ’temperature_data.csv’.")

Visualisation

plt.figure(figsize=(8, 6))

for i in range(0, Nt, Nt // 10) : # Affiche quelques étapes pour éviter une surcharge visuelle
plt.plot(x, u[i, :], label=f"t = {t[i]} :.2f} s")

plt.xlabel("Position x (m)")

plt.ylabel("Température u(x, t)")

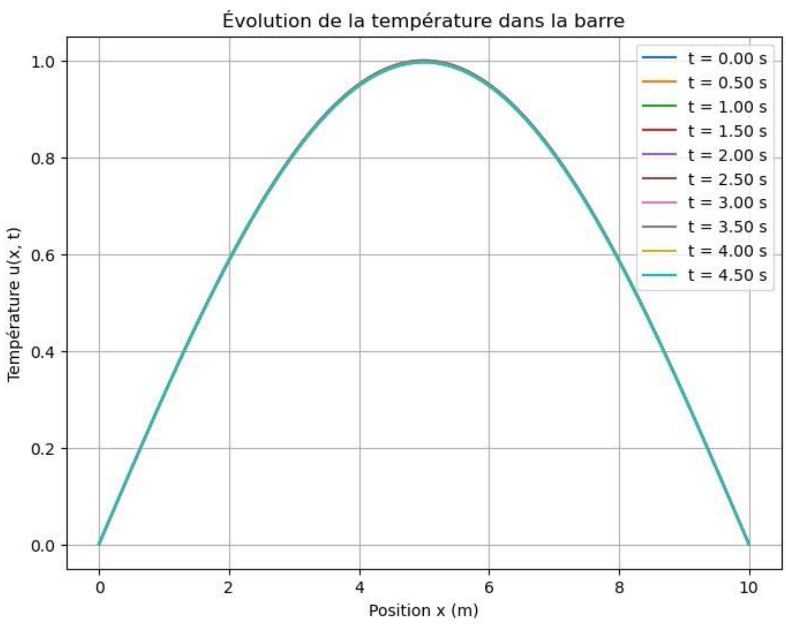
plt.title("Évolution de la température dans la barre")

plt.legend()

plt.grid()

plt.show()

graphe généré après la simulation



Création de la Base de Données Relationnelle

- Une base de données nommé *Simulation chaleur* a été créé comprenant trois tables :

1. Table *simulations*

Attributs :

id_simulation (clé primaire) : Identifiant unique de la simulation.

description : les conditions initiales.

longueur : Longueur de la tige (en mètres).

alpha : Coefficient de diffusion thermique.

N_X : Nombre de points de discrétisation spatiale.

N_t : Nombre de points de discrétisation temporelle.

dt : Pas de temps utilisé dans la simulation.

dx : Pas spatial utilisé dans la simulation.

2. Table *resultats*

Attributs :

id_resultat (clé primaire) : pour résultat

id_simulation (clé sécondaire) : lien avec la table simulations.

temps : L'instant temporel correspondant à ce résultat (en secondes).

position : La position spatiale sur la tige (en mètres).

temperature : La température calculée à cet instant et à cette position.

3. Table *parametres* Attributs :

id_condition (clé primaire) : Identifiant unique pour chaque paramètre initiale.

id_simulation (clé sécondaire) : lien avec la table simulations.

nom_du paramètres : Type de condition initiale.

valeur_parètres : Valeurs spécifiques associées à la condition initiale.

- Les scripts SQL nécessaires pour la création de la structure de la base de données : CREATE DATABSE simulation_chaleur ;

1. CREATE TABLE simulations (id_simulation SERIAL PRIMARY KEY, description TEXT NOT NULL,
   date_simulation TIMESTAMP DEFAULT CURRENT_TIMESTAMP

   );
2. CREATE TABLE parametres (id_parametre SERIAL PRIMARY KEY, id_simulation INTEGER NOT NULL, nom_parametre TEXT NOT NULL, valeur_parametre

FLOAT NOT NULL, FOREIGN KEY (id_simulation)
REFERENCES simulations (id_simulation)
ON DELETE CASCADE

);

3. CREATE TABLE resultats (id_resultat SERIAL PRIMARY KEY, id_simulation INTEGER NOT NULL, temps FLOAT NOT NULL, position FLOAT NOT NULL, temperature FLOAT NOT NULL,

   FOREIGN KEY (id_simulation) REFERENCES simulations (id_simulation)

   ON DELETE CASCADE

   );

Insertion et Exploitation des Données

- le code python pour l'insertion des résultats

```
import psycopg2
import pandas as pd
# Charger les données de température depuis le fichier CSV

u = pd.read_csv("temperature_data.csv", header=None)

Nx = 100 # Nombre de points spatiaux

Nt = 1000 # Nombre de points temporels

L = 10.0 # Longueur de la barre
T = 5.0 # Durée de la simulation
dx = L / (Nx - 1)
dt = T / Nt
x = [i * dx for i in range(Nx)]
t = [i * dt for i in range(Nt)]
# Connexion à PostgreSQL
conn = psycopg2.connect(
dbname="simulation_chaleur",
user="postgres",
password="Bi74ba96-bo!!", host="localhost",
port="5432"

)
```

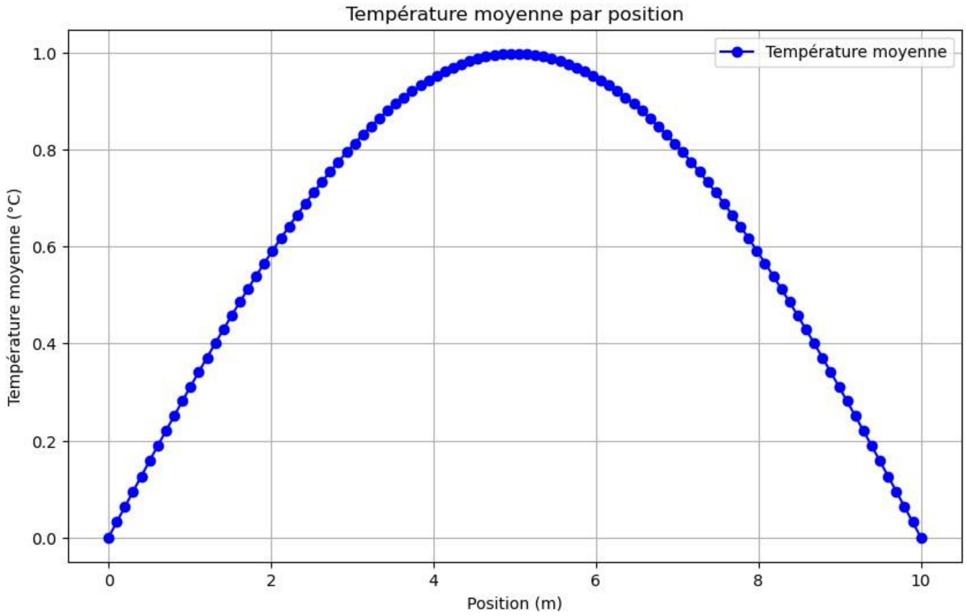
```
cursor = conn.cursor()

# Récupérer l'ID de la simulation insérée

cursor.execute("SELECT id_simulation FROM simulations WHERE

description=%s",

('Simulation de diffusion de la chaleur avec Nx=100, Nt=1000, alpha=0.01',))
id_simulation = cursor.fetchone()[0]

# Insérer les résultats dans la table

 for n, temps in enumerate(t) :

for i, position in enumerate(x) :

temperature = u.iloc[n, i]

cursor.execute(

"INSERT INTO resultats (id_simulation, temps, position, temperature) VALUES
(%s, %s, %s, %s);",

(id_simulation, temps, position, temperature)

)

# Valider les changements et fermer la connexion conn.commit()
cursor.close()
 conn.close()
print("Insertion des résultats terminée.")
```

- des requêtes SQL pour interroger et exploiter les donnée

Calculer la température moyenne sur la tige au fil du temps import psycopg2

```
import pandas as pd

import matplotlib.pyplot as plt

# Connexion à la base de données PostgreSQL
 conn = psycopg2.connect(
dbname="simulation_chaleur",
user="postgres",
 password="Bi74ba96-bo!!",
```

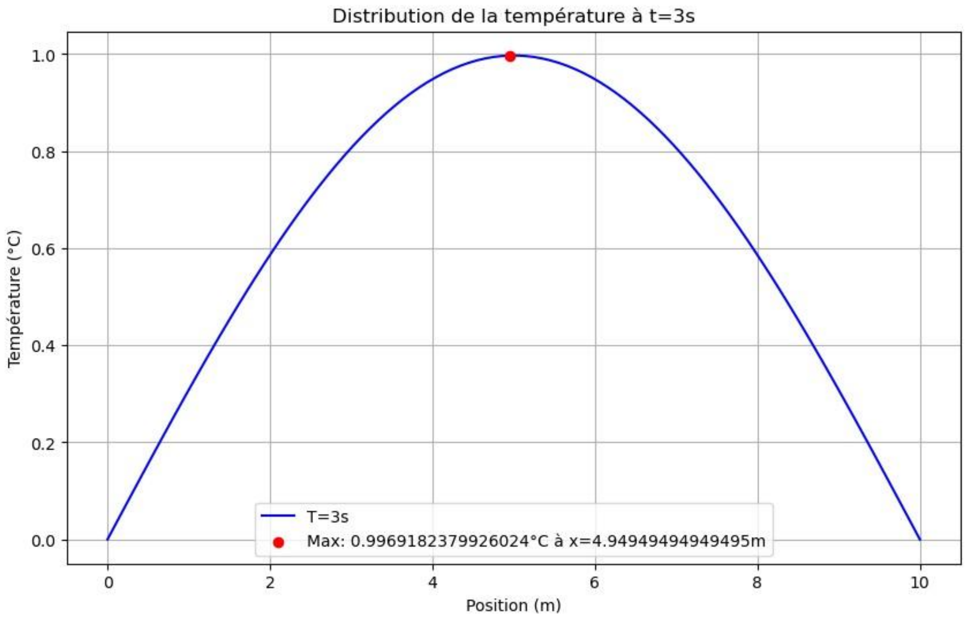
```
host="localhost",
port="5432"
)
# Exécuter la requête pour récupérer les températures moyennes query = """
SELECT position, AVG(temperature) AS temperature_moyenne
FROM resultats
GROUP BY position
ORDER BY position;
"""
data = pd.read_sql_query(query, conn)
# Fermer la connexion
conn.close()
# Créer un graphique
plt.figure(figsize=(10, 6))
plt.plot(data['position'], data['temperature_moyenne'], marker='o', color='blue',
label="Température moyenne")
 plt.xlabel("Position (m)")
plt.ylabel("Température moyenne (°C)")
plt.title("Température moyenne par position")
 plt.legend()
 plt.grid(True)
plt.show()
```



Identifier la position de la température maximale à un instant donné.

```
import psycopg2
 import pandas as pd
 import matplotlib.pyplot as plt
# Connexion à la base de données PostgreSQL
 conn = psycopg2.connect(
 dbname="simulation_chaleur",
 user="postgres",
password="Bi74ba96-bo!!",
 host="localhost",
port="5432"
)
# Spécification de l'instant de temps
 instant_temps = 3
# Requête pour récupérer les données à l'instant donné
query = f"""
SELECT position, temperature
FROM resultats
WHERE temps = {instant_temps}
ORDER BY position;
```

```
""" data = pd.read_sql_query(query, conn)

# Requête pour identifier la position de la température maximale

query_max = f"""

SELECT position, temperature

FROM resultats

WHERE temps = {instant_temps}

ORDER BY temperature DESC

LIMIT 1;

""" max_temp_result = pd.read_sql_query(query_max, conn)

# Fermer la connexion

conn.close()

# Extraire les informations de la température maximale

 position_max = max_temp_result['position'].iloc[0]

temperature_max = max_temp_result['temperature'].iloc[0]

print(f"À l'instant {instant_temps}s, la température maximale est de
{temperature_max}°C à la position {position_max}m.")

# Créer un graphique plt.figure(figsize=(10, 6))
plt.plot(data['position'], data['temperature'], label=f"T={instant_temps}s", color="blue")

plt.scatter(position_max, temperature_max, color="red", label=f"Max :
{temperature_max}°C à x={position_max}m", zorder=5)

plt.xlabel("Position (m)")

plt.ylabel("Température (°C)")

plt.title(f"Distribution de la température à t={instant_temps}s")

plt.legend()

plt.grid(True)

 plt.show()
```


Distribution de la température à t=3s
Température (°C)
Position (m)
T=3s
Max: 0.9969182379926024°C à x=4.94949494949495m
1.0
0.8
0.6
0.4
0.2
0.0
0
2
4
6
8
10