Approximation de fonctions

LOBANDJI LOPAKA CHARLES

March 6, 2024

1 Introduction

Dans le cadre de l'examen d'algorithmique et d'approximation du cours de L3 Informatique/Gestion à l'Université Nouveaux Horizons, notre mission était d'explorer différentes approches d'approximation de fonctions. Notre objectif était de modéliser un ensemble de données représentant une relation complexe entre une variable d'entrée (x) et une variable de sortie (y). Dans ce rapport, nous présenterons l'énoncé du problème ainsi que la solution que nous avons proposée.

2 Enoncé du problème

L'´enonc´e du probl`eme consistait `a effectuer les tˆaches suivantes :

2.1 Données à modéliser

Dans le cadre de notre travail, nous avons simulé au moins 20 observations de type (x, y) en respectant les consignes. Chaque étudiant devait choisir des données différentes afin d'éviter les duplications. De plus, nous avons fourni une justification ou une explication de la relation sous-jacente correspondant à ces données.

2.2 Estimation de la fonction

Nous devions appliquer la méthode de Gauss (Directe), la méthode de Lagrange et la méthode de Newton pour estimer la fonction à partir des données fournies.

2.3 Calcul de l'erreur

Dans le cadre de notre étude, nous avons calculé l'erreur d'approximation pour chaque méthode en comparant les valeurs prédites avec les valeurs réelles de la variable de sortie (y). Pour évaluer la précision de chaque approche, nous avons utilisé l'erreur quadratique moyenne (RMSE), une mesure recommandée dans ce contexte.

2.4 Visualisation des données

Dans le cadre de notre étude, nous avons présenté graphiquement les données originales (x, y) sur un graphique. Ensuite, nous avons superposé les courbes obtenues à partir de chaque méthode d'approximation sur le même graphique, ce qui nous a permis de les comparer visuellement**.**

3 Solution

Pour résoudre le problème nous avons suivi les étapes suivantes :

3.1 Génération des données

Nous avons généré 20 observations de type (x, y) en respectant les consignes de l'énoncé. De plus, nous avons également fourni une explication de la relation sous-jacente pour ces données.

3.2 Estimation de la fonction

Nous avons implémenté les méthodes de Gauss (Directe), de Lagrange et de Newton pour estimer la fonction à partir des données fournies. Chaque méthode a été appliquée séparément en utilisant les formules et les algorithmes appropriés.

3.3 Calcul de l'erreur

Nous avons calculé l'erreur d'approximation pour chaque méthode en comparant les valeurs prédites avec les valeurs réelles de la variable de sortie (y). Pour évaluer la précision de chaque approche, nous avons utilisé l'erreur quadratique moyenne (RMSE) comme métrique.

3.4 Visualisation des données

Nous avons créé un graphique pour représenter les données originales (x, y). Ensuite, nous avons superposé les courbes obtenues à partir de chaque méthode d'approximation sur le même graphique. Cela nous a permis de comparer visuellement les différentes approches.

4 Images du code et du résultat

4.1 Gauss et Lagrange

```python
# Méthode de Gauss
coefficients_gauss = np.polyfit(x_data, y_data, deg=5)

# Méthode de Lagrange
tabnine: test | explain | document | ask
def lagrange_interpolation(x, x_data, y_data):
    n = len(x_data)
    result = 0
    for i in range(n):
        term = y_data[i]
        for j in range(n):
            if j != i:
                term *= (x - x_data[j]) / (x_data[i] - x_data[j])
        result += term
    return result
```

Figure 1: Gauss et Lagrange

4.2 Newton

```python
# Méthode de Newton
tabnine: test | explain | document | ask
def divided_difference(x_data, y_data):
    n = len(x_data)
    F = np.zeros((n, n))
    F[:,0] = y_data
    for j in range(1,n):
        for i in range(n-j):
            F[i,j] = (F[i+1,j-1] - F[i,j-1]) / (x_data[i+j] - x_data[i])
    return F[0]

tabnine: test | explain | document | ask
def newton_interpolation(x, x_data, y_data):
    a = divided_difference(x_data, y_data)
    n = len(x_data) - 1
    p = a[n]
    for k in range(1, n + 1):
        p = a[n - k] + (x - x_data[n - k]) * p
    return p
```

Figure 2: Newton

4.3 Calcul de l'erreur, des valeurs prédites et des MSE pour chaque méthode

```python
# Calcul de l'erreur
tabnine: test | explain | document | ask
def calculate_rmse(y_true, y_pred):
    return np.sqrt(np.mean((y_true - y_pred) ** 2))

# Calcul des valeurs prédites pour chaque méthode
y_pred_gauss = np.polyval(coefficients_gauss, x_data)
y_pred_lagrange = lagrange_interpolation(x_data, x_data, y_data)
y_pred_newton = [newton_interpolation(xi, x_data, y_data) for xi in x_data]

# Calcul des RMSE pour chaque méthode
rmse_gauss = calculate_rmse(y_data, y_pred_gauss)
rmse_lagrange = calculate_rmse(y_data, y_pred_lagrange)
rmse_newton = calculate_rmse(y_data, y_pred_newton)
```

Figure 3: Calcul de l'erreur, des valeurs pr´edites et des MSE pour chaque m´ethode

4.4 Résultats

```python
# Affichage des résultats
print("RMSE pour la méthode de Gauss (Directe) :", rmse_gauss)
print("RMSE pour la méthode de Lagrange :", rmse_lagrange)
print("RMSE pour la méthode de Newton :", rmse_newton)

# Comparaison visuelle
plt.figure(figsize=(10, 6))
plt.scatter(x_data, y_data, label='Données originales')
plt.plot(x_data, y_pred_gauss, label=f'Méthode de Gauss (RMSE={rmse_gauss})')
plt.plot(x_data, y_pred_lagrange, label=f'Méthode de Lagrange (RMSE={rmse_lagrange})')
plt.plot(x_data, y_pred_newton, label=f'Méthode de Newton (RMSE={rmse_newton})')
plt.xlabel('x')
plt.ylabel('y')
plt.title('Comparaison des méthodes d\'approximation')
plt.legend()
plt.grid(True)
plt.show()
```

Figure 4: Affichages

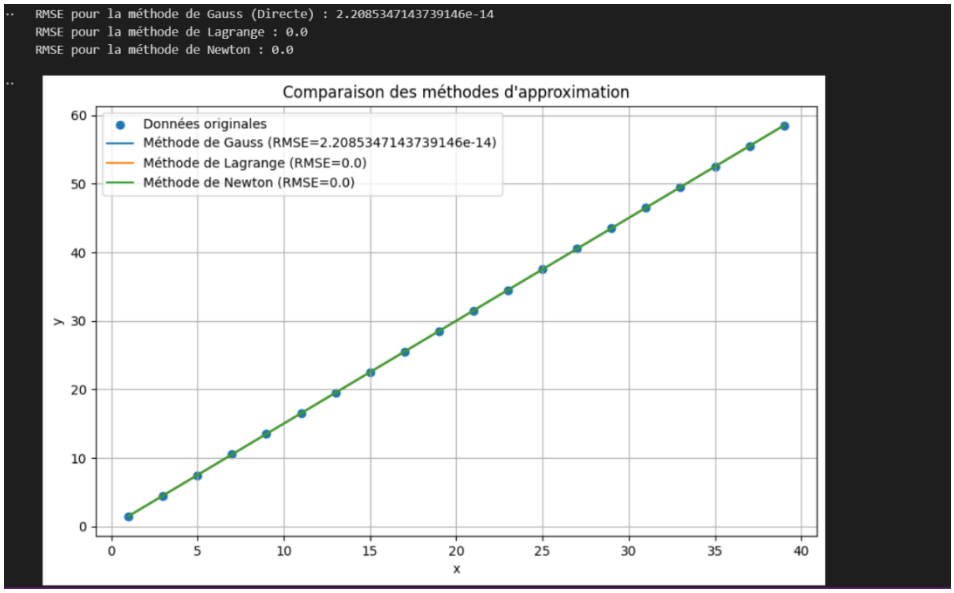

Figure 5: Resultat

5 Conclusion

En conclusion, nous avons réussi à modéliser une relation complexe entre une variable d'entrée et une variable de sortie en utilisant les méthodes d'approximation de Gauss, de Lagrange et de Newton. Nous avons calculé l'erreur d'approximation

pour chaque méthode et avons obtenu des résultats visuels clairs grâce aux graphiques générés. Cette expérience nous a permis de mieux comprendre l'importance de choisir la méthode d'approximation appropriée en fonction des données fournies.