Missing_data_detection

September 3, 2024

1 Loading packages

[86]:
```
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
import plotly.express as px
import plotly.graph_objects as go
from sklearn.cluster import DBSCAN
from scipy.stats import linregress
from sklearn.ensemble import IsolationForest
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPRegressor
```

2 Data exploration

[48]:
```
directory_path1 = '/Users/Ilyasse/Python projects/Equipement TA/7709.csv'
directory_path2 = '/Users/Ilyasse/Python projects/Equipement TS/37488.xlsx'
```

[49]:
```
ta_data =  pd.read_csv(directory_path1)
ts_data = pd.read_excel(directory_path2)
```

[50]:
```
ta_data.head()
```

[50]:

	Measure ID	NGV	Date of Creation	Point Name
0	4211208	1.35	2022-01-22T06:11:43.000000Z	1RV-0
1	4061956	0.42	2022-01-16T05:25:21.000000Z	1RV-0
2	4055996	0.42	2022-01-15T21:08:32.000000Z	1RV-0
3	4130532	1.90	2022-01-19T21:08:55.000000Z	1RV-0
4	4100892	0.34	2022-01-18T07:49:33.000000Z	1RV-0

[51]:
```
ts_data.head()
```

[51]:

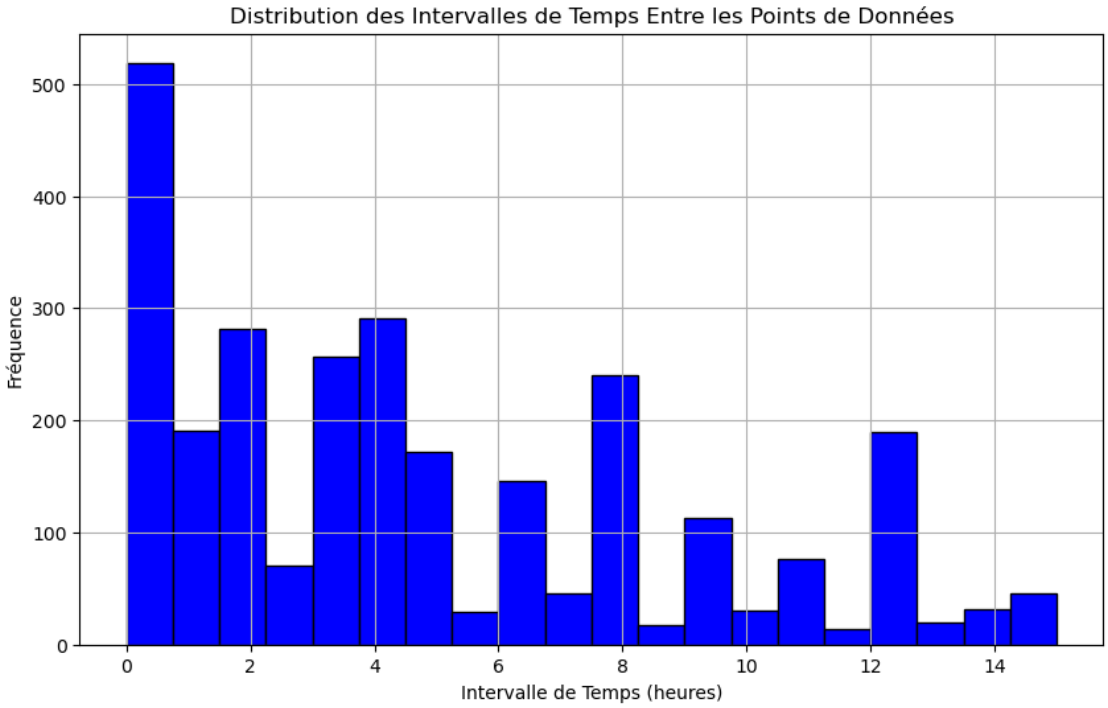
	Measure ID	NGV	Date of Creation	Point Name
0	12583124	0.23777	2023-06-21T18:40:00.000000Z	7RV-0
1	12591077	0.22020	2023-06-22T08:40:00.000000Z	7RV-0
2	12581880	0.23846	2023-06-21T16:40:00.000000Z	7RV-0

```
3    12583990  0.23766  2023-06-21T20:10:00.000000Z      7RV-0
4    12582106  0.23813  2023-06-21T17:10:00.000000Z      7RV-0
```

[52]:
```
ta_data['Date of Creation'] = pd.to_datetime(ta_data['Date of Creation'])
# Calculer la différence de temps entre les enregistrements successifs
ta_data['Time Diff'] = ta_data['Date of Creation'].diff()

print(ta_data['Time Diff'])
```

```
0                      NaT
1        -7 days +23:13:38
2        -1 days +15:43:11
3          4 days 00:00:23
4        -2 days +10:40:38
                 ...
10551      0 days 04:30:00
10552    -1 days +15:00:00
10553      0 days 07:30:00
10554    -1 days +22:30:00
10555    -1 days +21:00:00
Name: Time Diff, Length: 10556, dtype: timedelta64[ns]
```

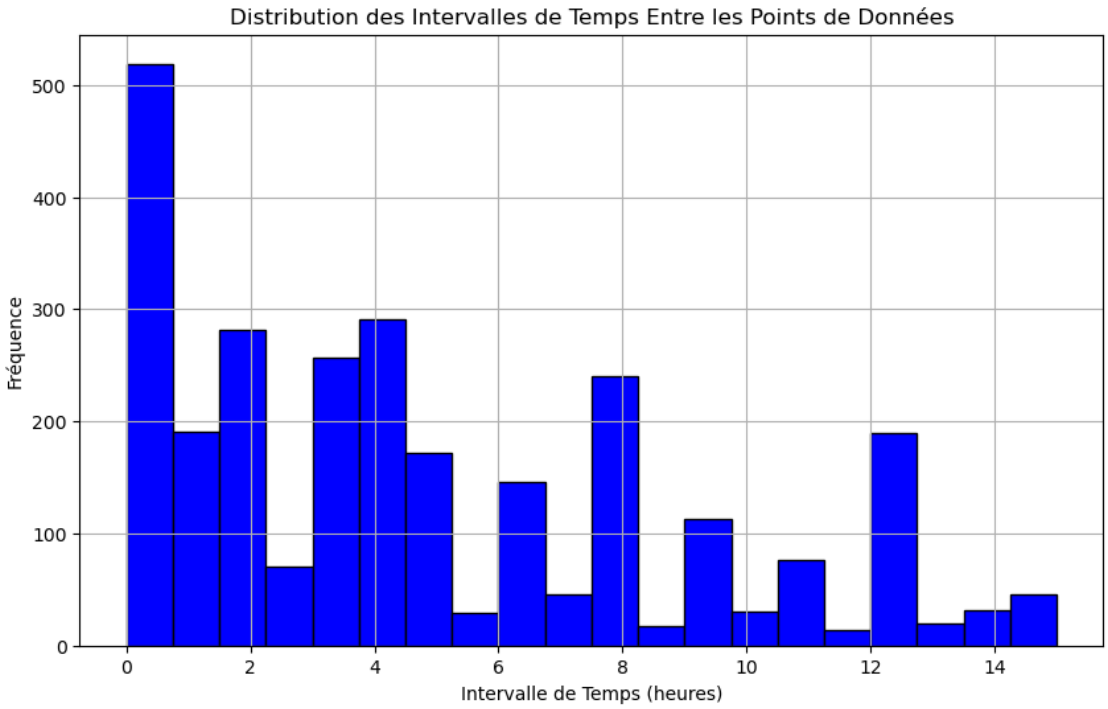
[53]:
```
# Calculer les intervalles de temps entre les points de données consécutifs
ta_data['time Diff'] = ta_data['Date of Creation'].diff().dt.total_seconds() /␣
↪3600  # Différence en heures

# Visualiser la distribution des intervalles de temps


plt.figure(figsize=(10, 6))
plt.hist(ta_data['time Diff'].dropna(), bins=20, range=(0,15), color='blue',␣
↪edgecolor='black')

plt.xlabel('Intervalle de Temps (heures)')
plt.ylabel('Fréquence')
plt.title('Distribution des Intervalles de Temps Entre les Points de Données')
plt.grid(True)
plt.show()

# Analyser les statistiques des intervalles de temps
print(ta_data['time Diff'].describe())
```




```
count    10555.000000
mean         1.827171
std       4491.150638
min     -11353.824444
25%        -15.073472
50%          0.500000
75%         20.000000
max      11311.891944
Name: time Diff, dtype: float64
```

[55]:
```
ts_data['Date of Creation'] = pd.to_datetime(ts_data['Date of Creation'])
# Calculer les intervalles de temps entre les points de données consécutifs
ts_data['time Diff'] = ts_data['Date of Creation'].diff().dt.total_seconds() /␣
 ↪3600  # Différence en heures

# Visualiser la distribution des intervalles de temps


plt.figure(figsize=(10, 6))
plt.hist(ta_data['time Diff'].dropna(), bins=20, range=(0,15), color='blue',␣
 ↪edgecolor='black')

plt.xlabel('Intervalle de Temps (heures)')
plt.ylabel('Fréquence')
```

```
plt.title('Distribution des Intervalles de Temps Entre les Points de Données')
plt.grid(True)
plt.show()


# Analyser les statistiques des intervalles de temps
print(ts_data['time Diff'].describe())
```




```
count    34723.000000
mean         0.016257
std         39.024456
min      -7110.848611
25%         -0.333333
50%          0.166667
75%          0.666667
max        620.833333
Name: time Diff, dtype: float64
```

[56]:
```
# Compter le nombre d'occurrences de chaque horodatage
timestamp_counts = ta_data['Date of Creation'].value_counts()

# Trouver les horodatages qui apparaissent plus d'une fois
simultaneous_measurements = timestamp_counts[timestamp_counts > 1]
```

```
print("Nombre de mesures simultanées:", len(simultaneous_measurements))
print(simultaneous_measurements)
```

```
Nombre de mesures simultanées: 2707
Date of Creation
2024-01-12 14:18:11+00:00    4
2022-01-22 06:11:43+00:00    2
2022-09-06 02:01:29+00:00    2
2022-09-06 18:01:29+00:00    2
2022-09-06 22:01:29+00:00    2
                            ..
2022-04-21 06:54:43+00:00    2
2022-04-21 07:39:07+00:00    2
2022-04-21 06:57:43+00:00    2
2022-04-21 07:01:45+00:00    2
2022-04-21 07:41:09+00:00    2
Name: count, Length: 2707, dtype: int64
```

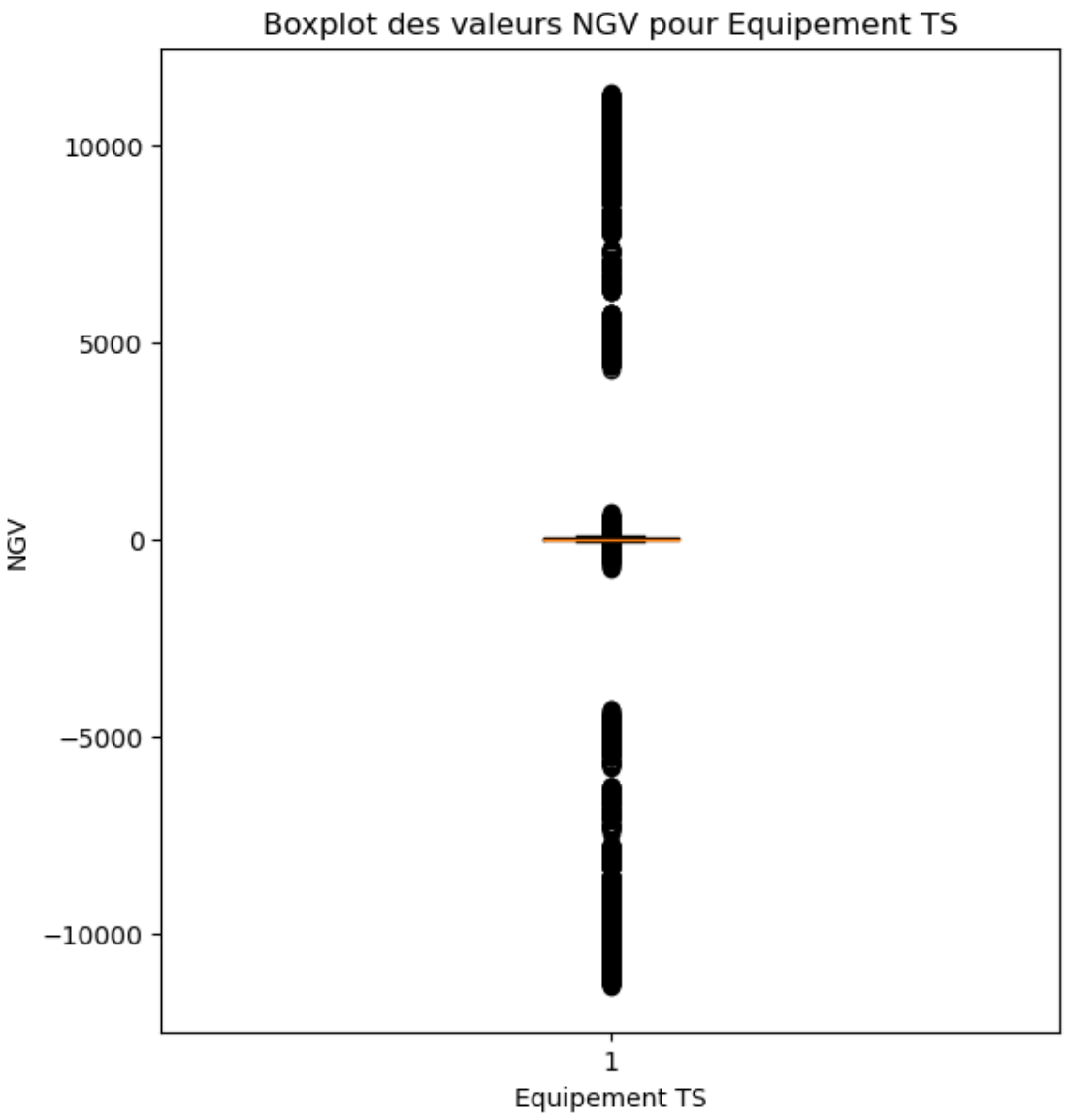
[57]:
```
# Compter le nombre d'occurrences de chaque horodatage
timestamp_counts = ts_data['Date of Creation'].value_counts()

# Trouver les horodatages qui apparaissent plus d'une fois
simultaneous_measurements = timestamp_counts[timestamp_counts > 1]

print("Nombre de mesures simultanées:", len(simultaneous_measurements))
print(simultaneous_measurements)
```

```
Nombre de mesures simultanées: 1182
Date of Creation
2023-06-22 05:40:00+00:00    4
2023-06-22 07:10:00+00:00    4
2023-12-29 00:20:00+00:00    3
2023-07-07 20:40:00+00:00    2
2023-07-08 00:40:00+00:00    2
                            ..
2023-06-29 09:10:00+00:00    2
2023-06-29 14:10:00+00:00    2
2023-06-29 07:10:00+00:00    2
2023-06-29 14:40:00+00:00    2
2023-06-29 15:10:00+00:00    2
Name: count, Length: 1182, dtype: int64
```

[58]:
```
plt.figure(figsize=(14, 7))

plt.subplot(1, 2, 1)
plt.boxplot(ta_data['time Diff'].dropna())
plt.title('Boxplot des valeurs NGV pour Equipement TS')
plt.ylabel('NGV')
```

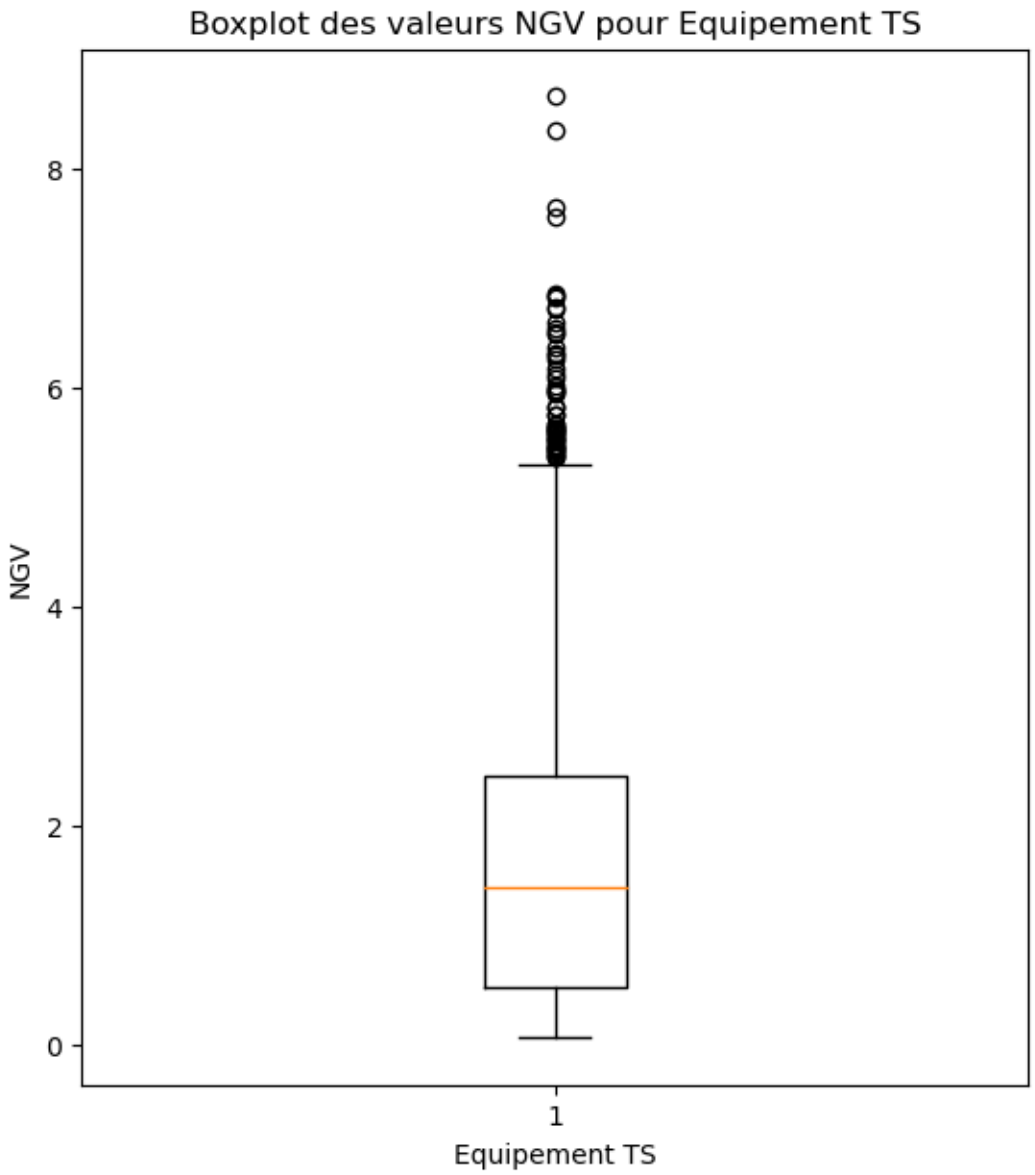
```
plt.xlabel('Equipement TS')
```

[58]: Text(0.5, 0, 'Equipement TS')



[59]:
```
plt.figure(figsize=(14, 7))

plt.subplot(1, 2, 1)
plt.boxplot(ta_data['NGV'].dropna())
plt.title('Boxplot des valeurs NGV pour Equipement TS')
plt.ylabel('NGV')
plt.xlabel('Equipement TS')
```

[59]: Text(0.5, 0, 'Equipement TS')



[61]:
```
# Function to analyze equipment data
def analyze_equipment(data, equipment_name):
    # Ensure the 'Date of Creation' column is in datetime format
    data['Date of Creation'] = pd.to_datetime(data['Date of Creation'])
    data = data.sort_values(by='Date of Creation')


    # Create an interactive plot with Plotly
    fig = go.Figure()
```

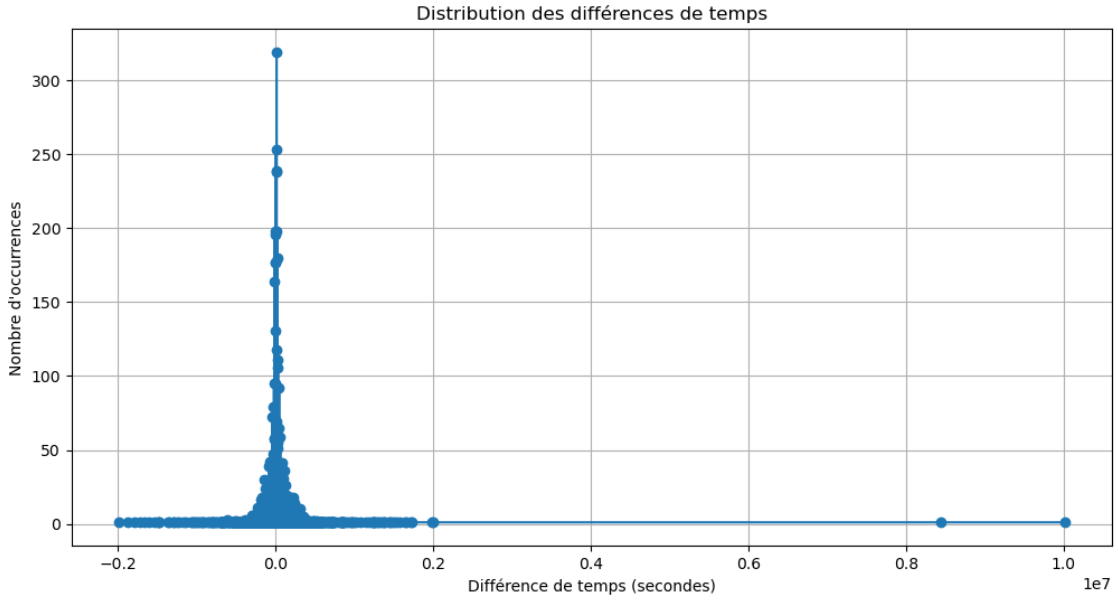
```
    # Add trace for NGV data
    fig.add_trace(go.Scatter(x=data['Date of Creation'],
                             y=data['NGV'],
                             mode='markers+lines',
                             name='NGV Équipement TA',
                             marker=dict(size=5),
                             line=dict(width=2)))

    # Update layout of the plot
    fig.update_layout(title='NGV en fonction de la Date de Creation',
                      xaxis_title='Date de Creation',
                      yaxis_title='NGV',
                      template='plotly_white',
                      legend_title='Legend')

    # Show the plot
    fig.show()
```

[62]:
```
def analyse_dossier(dossier, equipement):

    for root, dirs, files in os.walk(dossier):
        for file in files:
            if file.endswith('.xlsx'):
                chemin_fichier = os.path.join(root, file)
                print(f'Traitement du fichier: {chemin_fichier}')
                data = pd.read_excel(chemin_fichier)
                analyze_equipment(data, equipement)

            elif file.endswith('.csv'):
                chemin_fichier = os.path.join(root, file)
                print(f'Traitement du fichier: {chemin_fichier}')
                data = pd.read_csv(chemin_fichier)
                analyze_equipment(data, equipement)



    return
# Chemins des dossiers contenant les fichiers Excel pour les équipements TA et TS
dossier_ta = '/Users/Ilyasse/Python projects/Equipement TA'
dossier_ts = '/Users/Ilyasse/Python projects/Equipement TS'

# Traiter les fichiers pour chaque équipement
analyse_dossier(dossier_ta, 'TA')
analyse_dossier(dossier_ts, 'TS')
```

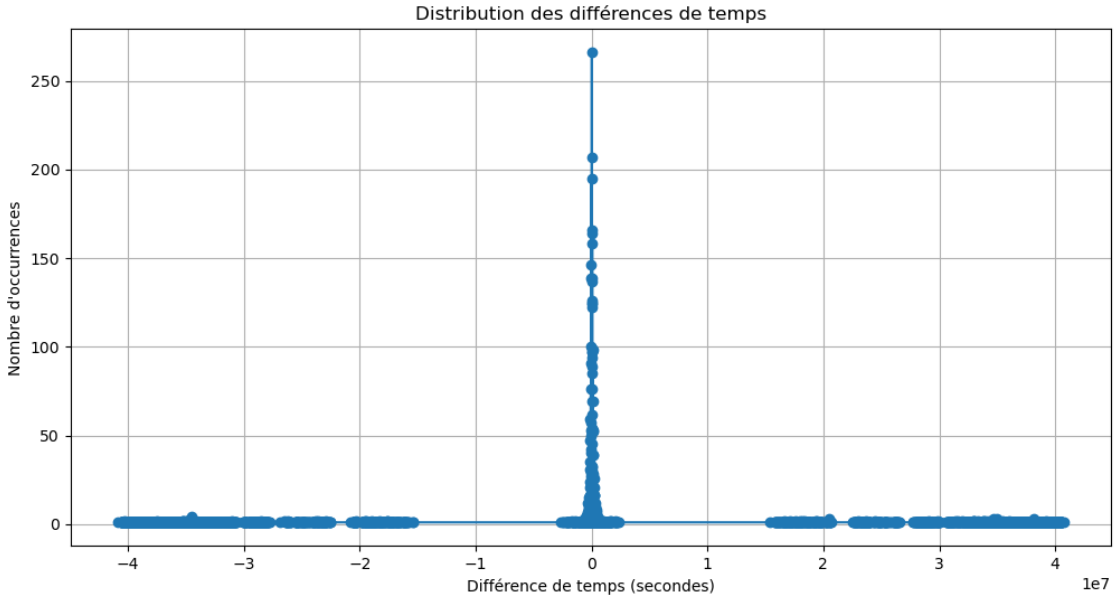
```
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7709.csv
```

```
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7710.csv
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7711.csv
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7712.csv
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7715.csv
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7718.csv
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7721.csv
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7724.csv
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7727.csv
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7728.csv
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7729.csv
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7730.csv
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TS\37482.xlsx
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TS\37483.xlsx
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TS\37484.xlsx
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TS\37485.xlsx
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TS\37486.xlsx
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TS\37487.xlsx
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TS\37488.xlsx
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TS\37489.xlsx
```

[63]:
```python
folder_path = '/Users/Ilyasse/Python projects/Equipement TA/7721.csv'
data = pd.read_csv(folder_path)
data['Date of Creation'] = pd.to_datetime(data['Date of Creation'])

# Calculer les différences de temps en secondes
data['time_diff'] = data['Date of Creation'].diff().dt.total_seconds()

# Compter le nombre d'occurrences de chaque différence
time_diff_counts = data['time_diff'].value_counts().sort_index()

# Tracer le graphique
plt.figure(figsize=(12, 6))
plt.plot(time_diff_counts.index, time_diff_counts.values, marker='o')
plt.title('Distribution des différences de temps')
plt.xlabel('Différence de temps (secondes)')
plt.ylabel('Nombre d\'occurrences')
plt.grid(True)
plt.show()
```



[64]:
```
folder_path = '/Users/Ilyasse/Python projects/Equipement TA/7709.csv'
data = pd.read_csv(folder_path)
data['Date of Creation'] = pd.to_datetime(data['Date of Creation'])

# Calculer les différences de temps en secondes
data['time_diff'] = data['Date of Creation'].diff().dt.total_seconds()

# Compter le nombre d'occurrences de chaque différence
time_diff_counts = data['time_diff'].value_counts().sort_index()

# Tracer le graphique
plt.figure(figsize=(12, 6))
plt.plot(time_diff_counts.index, time_diff_counts.values, marker='o')
plt.title('Distribution des différences de temps')
plt.xlabel('Différence de temps (secondes)')
plt.ylabel('Nombre d\'occurrences')
plt.grid(True)
plt.show()
```



3 Data traitement

4 Statistic approach

[65]:
```
# Définir une période au-delà de laquelle nous considérons qu'il y a une coupure␣
↪(par exemple, 0.5 heure)
threshold = pd.Timedelta(hours=1)

# Identifier les périodes où la différence de temps dépasse le seuil
missing_data = ta_data[ta_data['Time Diff'] > threshold]

# Afficher les périodes d'absence de données
print("Périodes d'absence de données :")
print(missing_data[['Date of Creation', 'Time Diff']])
```

```
Périodes d'absence de données :
              Date of Creation       Time Diff
3     2022-01-19 21:08:55+00:00 4 days 00:00:23
5     2022-01-23 18:11:43+00:00 5 days 10:22:10
8     2022-01-25 02:20:31+00:00 7 days 20:55:10
10    2022-01-20 05:08:55+00:00 2 days 09:31:40
11    2022-01-20 18:11:43+00:00 0 days 13:02:48
...                         ...             ...
10545 2024-04-04 15:29:02+00:00 0 days 04:30:00
10547 2024-04-04 18:29:02+00:00 0 days 10:30:00
10550 2024-04-04 21:29:02+00:00 0 days 09:00:00
```

```
10551 2024-04-05 01:59:02+00:00 0 days 04:30:00
10553 2024-04-05 00:29:02+00:00 0 days 07:30:00

[5010 rows x 2 columns]
```

[66]:
```python
# Utiliser l'IQR pour identifier les outliers
Q1 = np.percentile(ta_data['Time Diff'].dropna(), 25)
Q3 = np.percentile(ta_data['Time Diff'].dropna(), 75)
IQR = Q3 - Q1
seuil = Q3 + 1.5 * IQR  # Seuil basé sur l'IQR
print(f'Seuil proposé basé sur IQR pour l\'équipement TA', seuil)

# Afficher les données avec anomalies
anomalies = ta_data[ta_data['Time Diff'] > seuil]
print(f'Nombre d\'anomalies détectées pour l\'équipement TA:', len(anomalies))
```

```
Seuil proposé basé sur IQR pour l'équipement TA 261396750000000 nanoseconds
Nombre d'anomalies détectées pour l'équipement TA: 1690
```

[67]:
```python
# Utiliser l'IQR pour identifier les outliers
Q1 = np.percentile(ts_data['time Diff'].dropna(), 25)
Q3= np.percentile(ts_data['time Diff'].dropna(), 75)
IQR = Q3 - Q1

seuil = Q3 + 1.5 * IQR  # Seuil basé sur l'IQR
print(f'Seuil proposé basé sur IQR pour l\'équipement TS', seuil)

# Afficher les données avec anomalies
anomalies = ts_data[ts_data['time Diff'] > seuil]
print(f'Nombre d\'anomalies détectées pour l\'équipement TS:', len(anomalies))
```

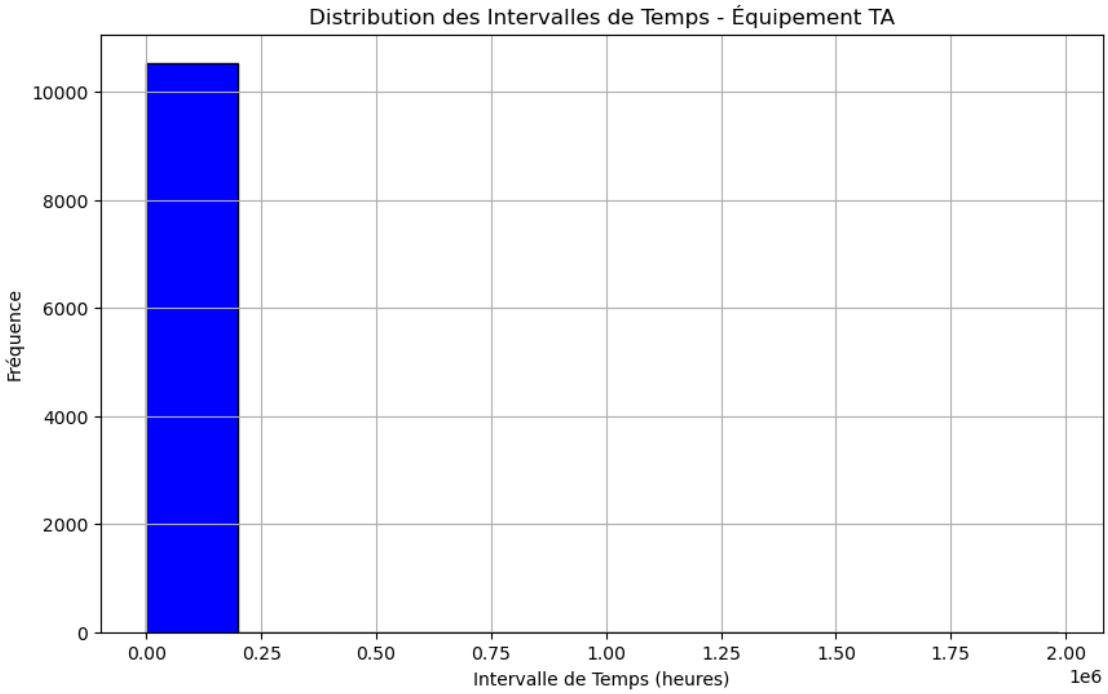
```
Seuil proposé basé sur IQR pour l'équipement TS 2.1666666666666665
Nombre d'anomalies détectées pour l'équipement TS: 1382
```

[49]:
```python
# Fonction pour analyser les intervalles de temps et déterminer un seuil
def analyse_intervalles(data, equipement):
    # Conversion de la colonne 'Date of Creation' en type datetime
    data['Date of Creation'] = pd.to_datetime(data['Date of Creation'])
    # Trier les données par date
    data.sort_values(by='Date of Creation', inplace=True)
    # Calculer les intervalles de temps entre chaque point de données consécutif
    data['time_diff'] = data['Date of Creation'].diff().dt.total_seconds()   #␣
 ↪Différence en heures
    time_difference=[]
    for i in range(0,200):

        time_difference.append(data['time_diff'][i])
    print(time_difference)
```

```
    # Visualiser la distribution des intervalles de temps
    plt.figure(figsize=(10, 6))
    plt.hist(data['time_diff'].dropna(), bins=10, color='blue',␣
↪edgecolor='black')
    plt.xlabel('Intervalle de Temps (heures)')
    plt.ylabel('Fréquence')
    plt.title(f'Distribution des Intervalles de Temps - Équipement {equipement}')
    plt.grid(True)
    plt.show()

    # Afficher les statistiques des intervalles de temps
    stats = data['time_diff'].describe()
    print(stats)

    # Utiliser l'IQR pour identifier les outliers
    Q1 = np.percentile(data['time_diff'].dropna(), 25)
    Q3 = np.percentile(data['time_diff'].dropna(), 75)
    IQR = Q3 - Q1

    seuil = Q3 + 1.5 * IQR  # Seuil basé sur l'IQR
    print(f'Seuil proposé basé sur IQR pour l\'équipement {equipement}:', seuil)

    # Afficher les données avec anomalies
    anomalies = data[data['time_diff'] > seuil]
    print(f'Nombre d\'anomalies détectées pour l\'équipement {equipement}:',␣
↪len(anomalies))

    return seuil

# Parcourir les fichiers Excel dans le dossier spécifié et analyser les données
def traiter_dossier(dossier, equipement):
    seuils = []
    for root, dirs, files in os.walk(dossier):
        for file in files:
            if file.endswith('.xlsx'):
                chemin_fichier = os.path.join(root, file)
                print(f'Traitement du fichier: {chemin_fichier}')
                data = pd.read_excel(chemin_fichier)
                seuil = analyse_intervalles(data, equipement)
                seuils.append(seuil)
            elif file.endswith('.csv'):
                chemin_fichier = os.path.join(root, file)
                print(f'Traitement du fichier: {chemin_fichier}')
                data = pd.read_csv(chemin_fichier)
                seuil = analyse_intervalles(data, equipement)
                seuils.append(seuil)
```

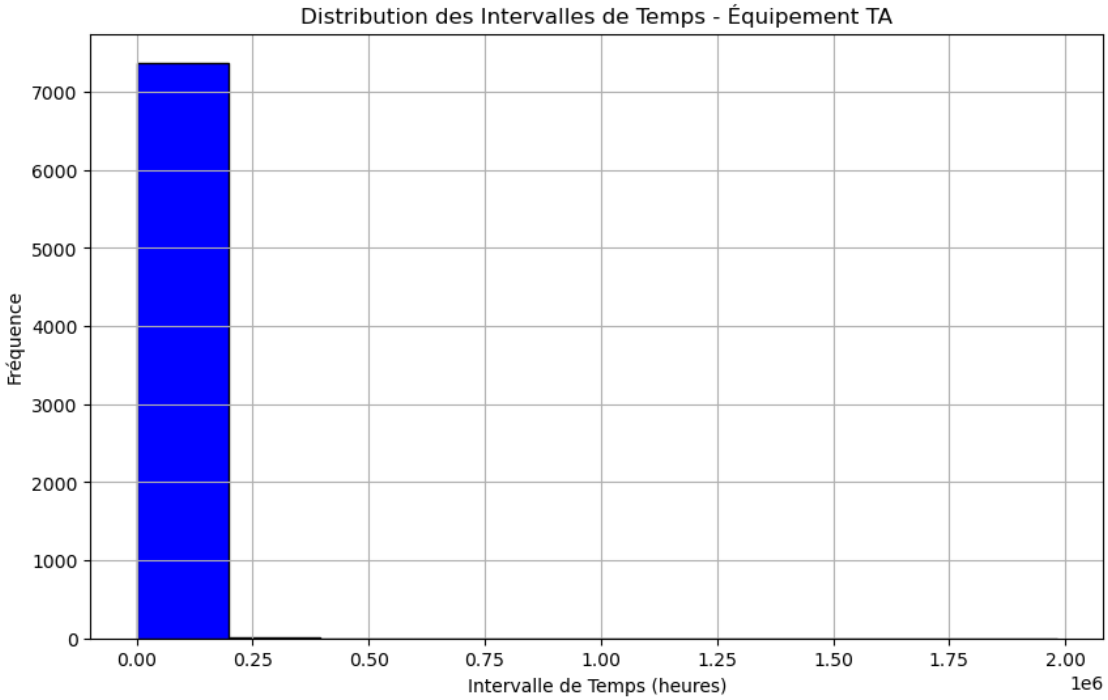
```
    # Calculer un seuil moyen pour tous les fichiers
    seuil_moyen = np.mean(seuils)
    print(f'Seuil moyen pour l\'équipement {equipement}:', seuil_moyen)
    return seuil_moyen

# Chemins des dossiers contenant les fichiers Excel pour les équipements TA et TS
dossier_ta = '/Users/Ilyasse/Python projects/Equipement TA'
dossier_ts = '/Users/Ilyasse/Python projects/Equipement TS'

# Traiter les fichiers pour chaque équipement
seuil_ta = traiter_dossier(dossier_ta, 'TA')
seuil_ts = traiter_dossier(dossier_ts, 'TS')

print(f'Seuil final pour TA: {seuil_ta}')
print(f'Seuil final pour TS: {seuil_ts}')
```

```
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7709.csv
[0.0, 14400.0, nan, 0.0, 14400.0, 0.0, 0.0, 0.0, 0.0, 14400.0, 14400.0, 0.0,
0.0, 14400.0, 1331.0, 14400.0, 0.0, 0.0, 0.0, 14400.0, 0.0, 14400.0, 0.0, 0.0,
0.0, 14400.0, 0.0, 0.0, 14593.0, 14400.0, 0.0, 14400.0, 0.0, 14400.0, 0.0, 0.0,
5388.0, 14400.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1009.0, 0.0, 0.0, 0.0, 0.0, 0.0,
14400.0, 14400.0, 0.0, 0.0, 0.0, 14400.0, 0.0, 14400.0, 0.0, 14400.0, 14400.0,
0.0, 0.0, 14400.0, 14400.0, 14400.0, 14401.0, 0.0, 14400.0, 14400.0, 0.0,
14400.0, 0.0, 0.0, 14400.0, 0.0, 14400.0, 14400.0, 0.0, 14400.0, 0.0, 14400.0,
0.0, 0.0, 0.0, 14400.0, 14400.0, 14400.0, 0.0, 14400.0, 0.0, 0.0, 14400.0,
14400.0, 0.0, 14400.0, 0.0, 14400.0, 14400.0, 0.0, 14400.0, 2797.0, 14401.0,
14400.0, 0.0, 0.0, 14400.0, 0.0, 0.0, 14400.0, 0.0, 0.0, 0.0, 14400.0, 14400.0,
0.0, 0.0, 14400.0, 0.0, 14400.0, 4243.0, 0.0, 0.0, 0.0, 14400.0, 0.0, 0.0, 0.0,
0.0, 14400.0, 14400.0, 14400.0, 0.0, 0.0, 0.0, 10762.0, 14400.0, 14400.0, 0.0,
14400.0, 14400.0, 14400.0, 0.0, 0.0, 0.0, 14400.0, 0.0, 0.0, 14400.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 14400.0, 0.0, 0.0, 0.0, 14400.0, 14400.0, 0.0, 14400.0,
14400.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 14400.0, 0.0, 0.0, 0.0, 0.0,
14400.0, 14400.0, 0.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 14400.0, 0.0, 0.0, 14400.0, 14400.0, 14400.0, 0.0, 0.0,
14400.0, 14400.0, 0.0]
```



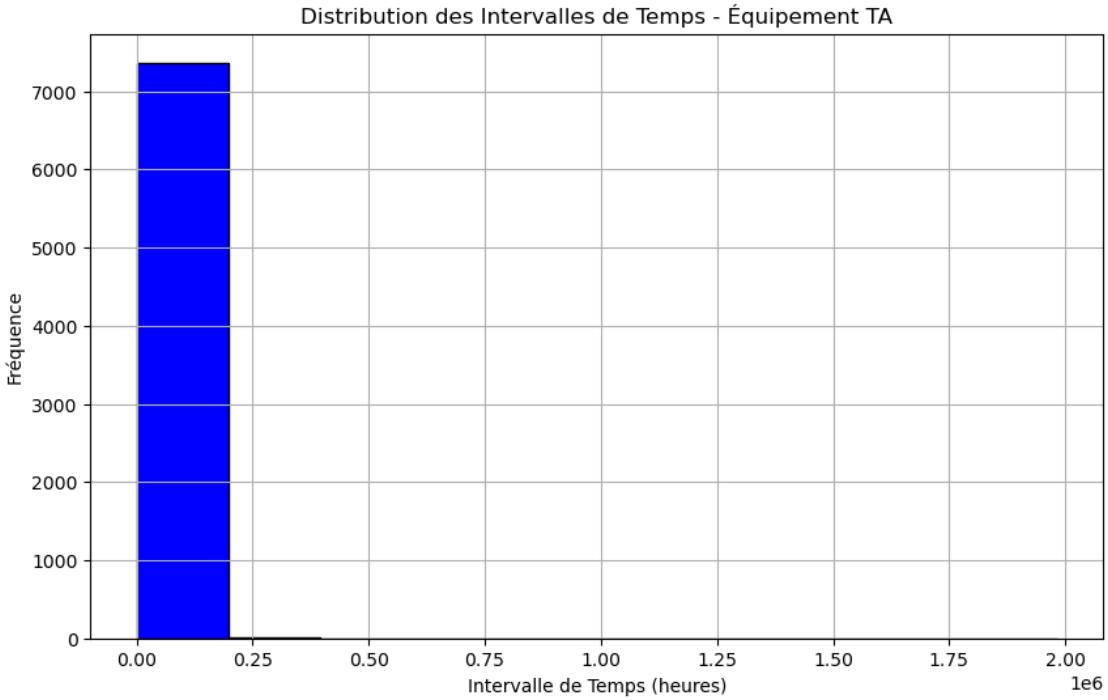

```
count    1.055500e+04
mean     6.632063e+03
std      3.336866e+04
min      0.000000e+00
25%      0.000000e+00
50%      3.600000e+03
75%      1.440000e+04
max      1.982545e+06
Name: time_diff, dtype: float64
Seuil proposé basé sur IQR pour l'équipement TA: 36000.0
Nombre d'anomalies détectées pour l'équipement TA: 31
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7710.csv
[1009.0, 14400.0, 101328.0, 14400.0, 14400.0, 14400.0, 14593.0, 1278.0, 14400.0,
14400.0, 1248.0, 14400.0, 14400.0, 15138.0, 3575.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 104231.0, 1331.0, 604.0, 33670.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 5388.0, 14400.0, 14400.0,
nan, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 8029.0, 14400.0, 14400.0,
14400.0, 14400.0, 14401.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 13380.0,
14400.0, 14400.0, 14400.0, 14400.0, 2797.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14401.0, 14400.0, 707.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
```

```
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 4243.0, 14400.0, 14400.0, 10762.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14401.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 10265.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0]
```



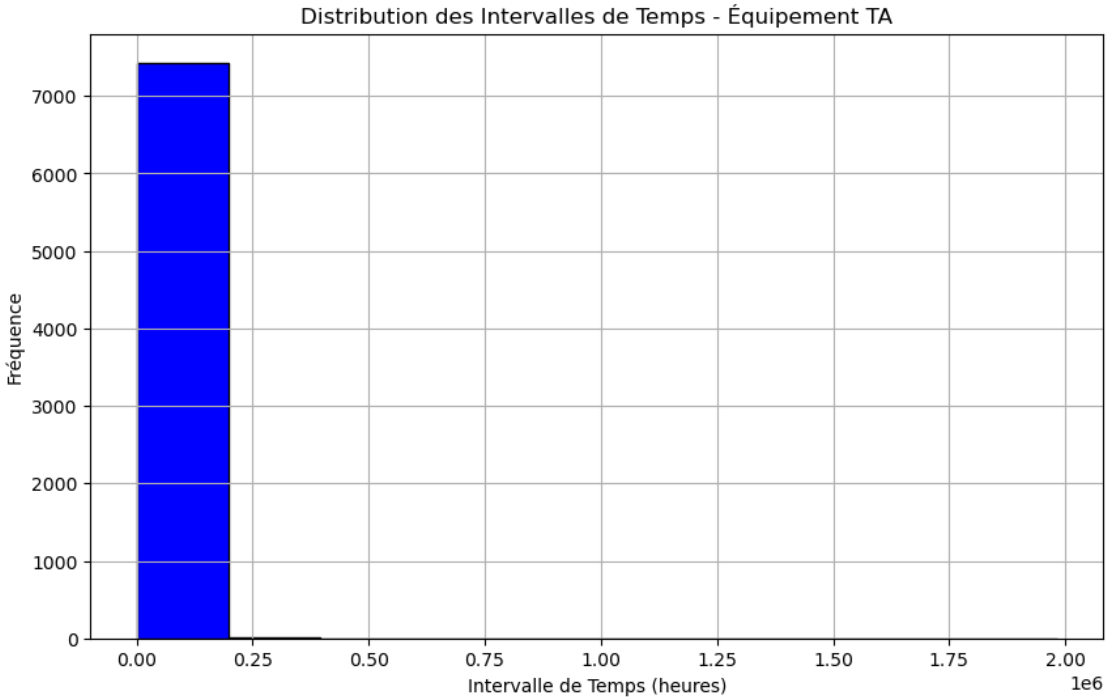

```
count    7.388000e+03
mean     9.479657e+03
std      3.929021e+04
min      0.000000e+00
25%      3.600000e+03
50%      5.400000e+03
75%      1.440000e+04
max      1.982545e+06
Name: time_diff, dtype: float64
Seuil proposé basé sur IQR pour l'équipement TA: 30600.0
Nombre d'anomalies détectées pour l'équipement TA: 36
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7711.csv
[1331.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 1248.0, 5388.0, 14400.0, 14400.0, 1278.0, 14400.0, 104231.0,
```

```
14400.0, 14400.0, 3575.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14593.0, 14400.0, nan, 14400.0, 14400.0, 101328.0, 14400.0, 738.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 8029.0, 14400.0, 14400.0,
14400.0, 14401.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 13380.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 2797.0,
14400.0, 14401.0, 14400.0, 14400.0, 14400.0, 14400.0, 4243.0, 14400.0, 14400.0,
707.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14401.0, 14400.0, 10762.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 10265.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0]
```



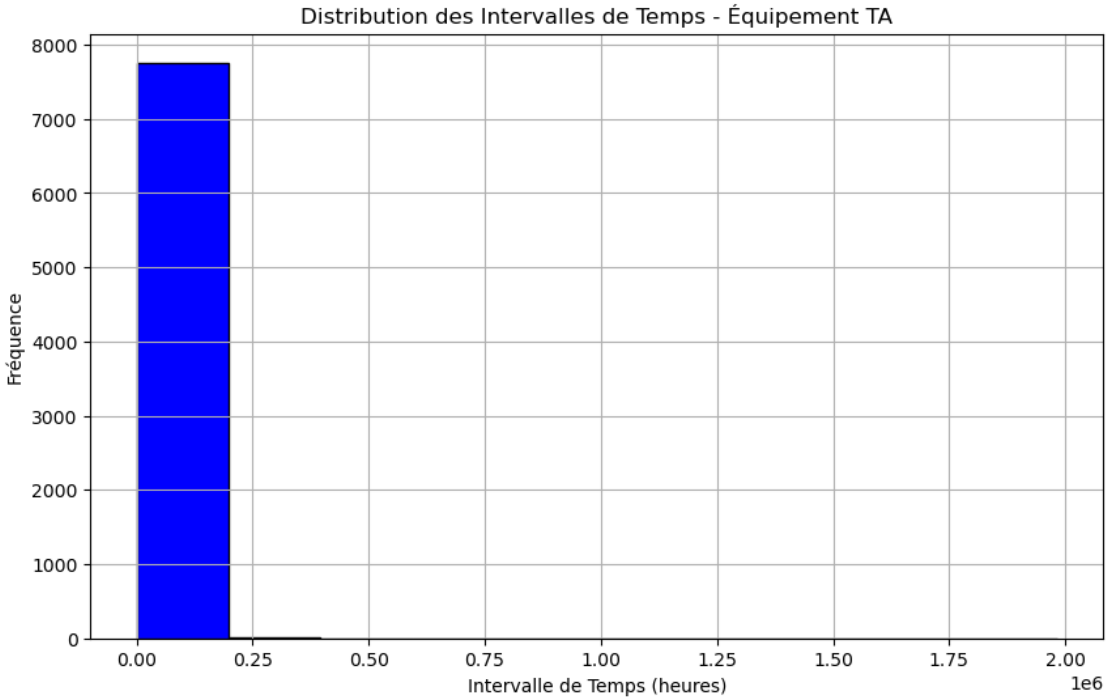

```
count    7.379000e+03
mean     9.487169e+03
std      4.085757e+04
```

```
min      0.000000e+00
25%      3.600000e+03
50%      5.400000e+03
75%      1.440000e+04
max      1.982545e+06
Name: time_diff, dtype: float64
Seuil proposé basé sur IQR pour l'équipement TA: 30600.0
Nombre d'anomalies détectées pour l'équipement TA: 35
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7712.csv
[3575.0, 14400.0, 14400.0, 14400.0, 14400.0, 34274.0, 14400.0, 14400.0, 5388.0,
14400.0, 14400.0, 14593.0, 14400.0, 14400.0, 14400.0, 28800.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 101328.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, nan, 738.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
1331.0, 104231.0, 1278.0, 14400.0, 14400.0, 14400.0, 1248.0, 1009.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 8029.0,
14401.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 13380.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 2797.0, 14400.0, 14400.0, 14400.0, 14400.0, 14401.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 707.0,
4243.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 10762.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14401.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 10265.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0]
```



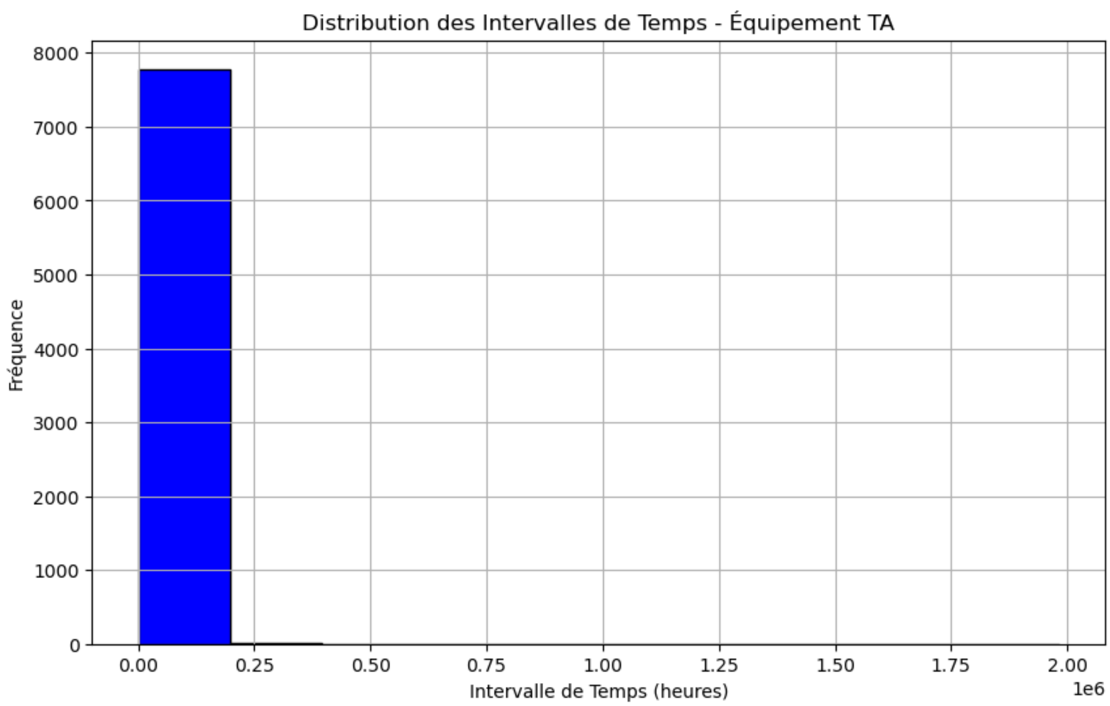

```
count    7.444000e+03
mean     9.409794e+03
std      4.071600e+04
min      0.000000e+00
25%      3.600000e+03
50%      5.400000e+03
75%      1.440000e+04
max      1.982545e+06
Name: time_diff, dtype: float64
Seuil proposé basé sur IQR pour l'équipement TA: 30600.0
Nombre d'anomalies détectées pour l'équipement TA: 29
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7715.csv
[14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14593.0, 14400.0, 14400.0, 104231.0, 14400.0, 101328.0,
14400.0, 14400.0, 738.0, 14400.0, 14400.0, 1009.0, 14400.0, 14400.0, 5388.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 28800.0, 3575.0, 1248.0,
14400.0, 14400.0, 14400.0, nan, 14400.0, 14400.0, 14400.0, 1331.0, 14400.0,
14400.0, 33670.0, 14400.0, 1278.0, 14400.0, 14400.0, 14400.0, 604.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 8030.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 13380.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 2797.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14401.0, 14401.0, 707.0, 14400.0, 14400.0, 14399.0, 14399.0,
```

```
14400.0, 14400.0, 14401.0, 4242.0, 14400.0, 14401.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 10762.0, 14400.0, 14400.0, 14400.0, 14400.0, 14401.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 10265.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0]
```



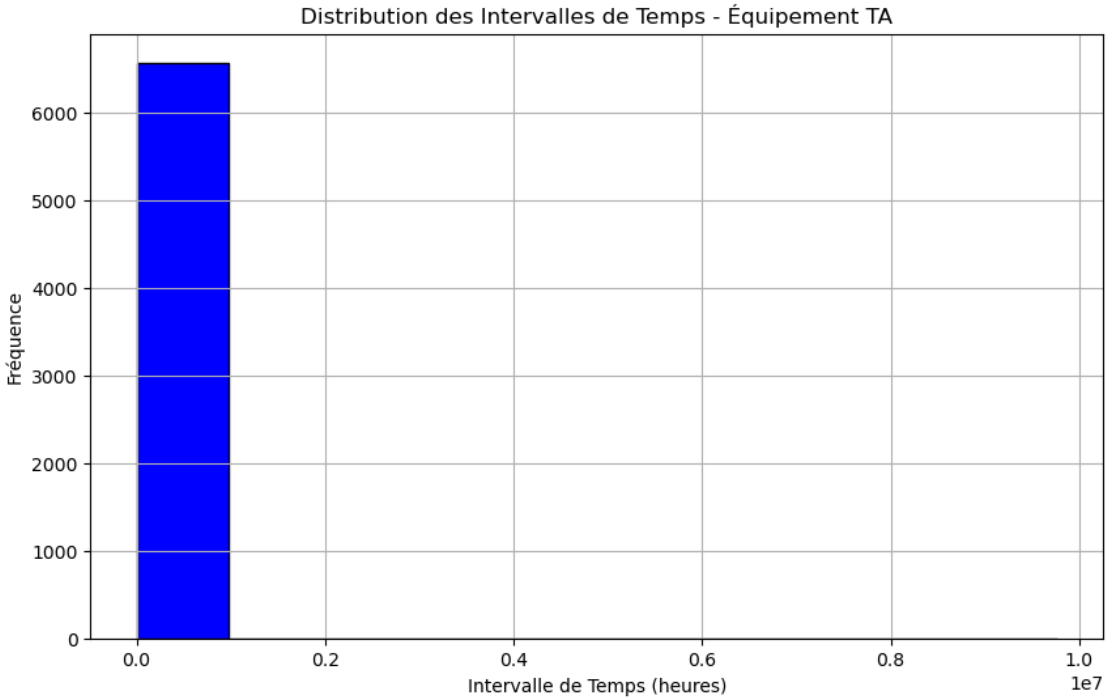

```
count    7.768000e+03
mean     9.016620e+03
std      3.589990e+04
min      0.000000e+00
25%      3.600000e+03
50%      5.400000e+03
75%      1.440000e+04
max      1.982545e+06
Name: time_diff, dtype: float64
Seuil proposé basé sur IQR pour l'équipement TA: 30600.0
Nombre d'anomalies détectées pour l'équipement TA: 26
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7718.csv
[14400.0, 33670.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14593.0, 5388.0,
101328.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 104231.0,
```

```
3575.0, 16926.0, 14400.0, 604.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
1009.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 1331.0, 14400.0,
14400.0, 14400.0, 738.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, nan, 14400.0, 43200.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 8030.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 13380.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14399.0, 14400.0, 14400.0, 14400.0, 14401.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14401.0, 707.0, 14400.0, 14399.0, 14400.0, 14401.0, 2797.0, 4242.0,
14401.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 10762.0, 14401.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 10265.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0]
```



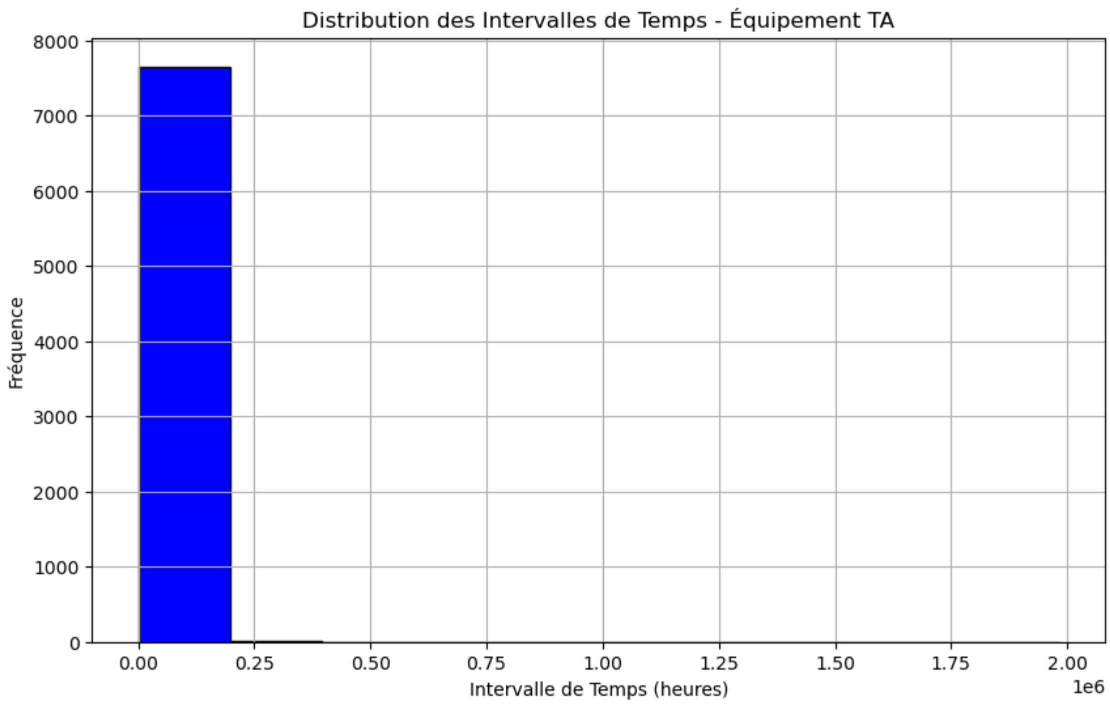

```
count    7.790000e+03
mean     8.993235e+03
std      3.594233e+04
```

```
min      0.000000e+00
25%      3.600000e+03
50%      5.400000e+03
75%      1.440000e+04
max      1.982545e+06
Name: time_diff, dtype: float64
Seuil proposé basé sur IQR pour l'équipement TA: 30600.0
Nombre d'anomalies détectées pour l'équipement TA: 29
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7721.csv
[14400.0, 14400.0, 14400.0, 43200.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 101328.0, 14400.0, 14400.0, 14400.0, 14400.0, nan, 14400.0,
14400.0, 14400.0, 14593.0, 3575.0, 14400.0, 14400.0, 14400.0, 5388.0, 738.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 15731.0, 14400.0,
14400.0, 16926.0, 14400.0, 104231.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 8030.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 13380.0, 14400.0, 14400.0, 14400.0, 14400.0, 14401.0, 14400.0, 2797.0,
14399.0, 14400.0, 14400.0, 14400.0, 14400.0, 707.0, 14400.0, 14401.0, 14400.0,
14400.0, 14401.0, 4242.0, 14400.0, 14400.0, 14399.0, 14400.0, 14401.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 10762.0, 14400.0,
14400.0, 14401.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 10265.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0]
```



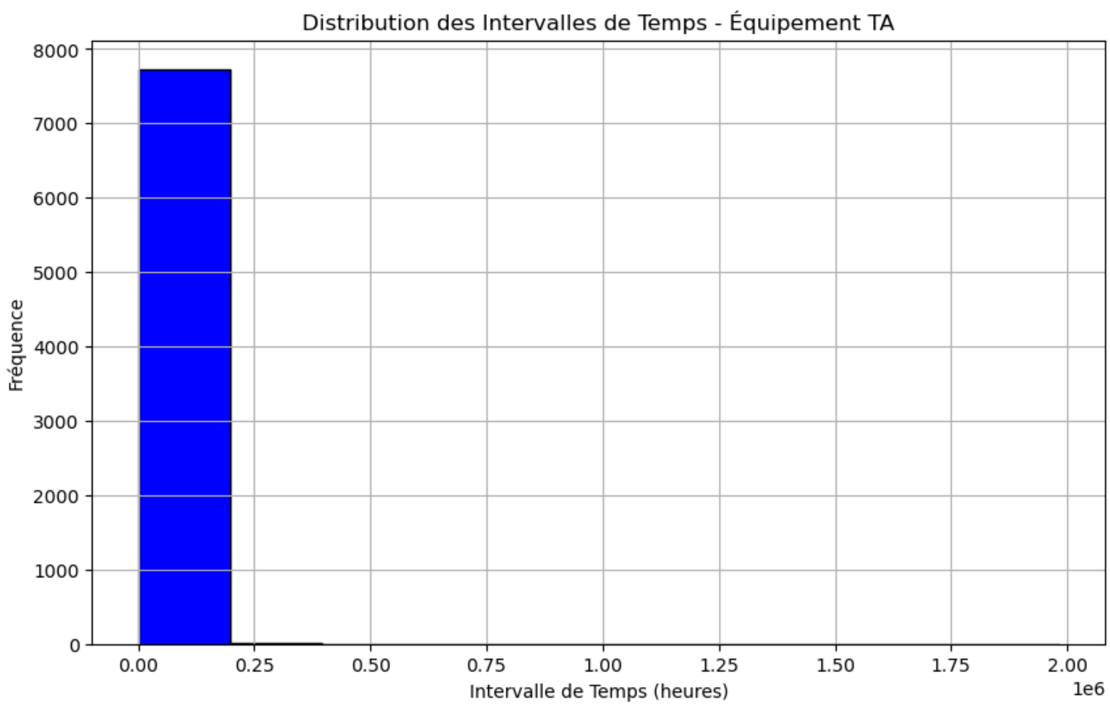

```
count    6.573000e+03
mean     1.065380e+04
std      1.612672e+05
min      0.000000e+00
25%      1.801000e+03
50%      5.400000e+03
75%      1.440000e+04
max      9.766020e+06
Name: time_diff, dtype: float64
Seuil proposé basé sur IQR pour l'équipement TA: 33298.5
Nombre d'anomalies détectées pour l'équipement TA: 28
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7724.csv
[1009.0, 14400.0, 14400.0, 14400.0, 14400.0, 15731.0, 14400.0, 14400.0, 5388.0,
14400.0, 14400.0, 14400.0, 16926.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 43200.0, 738.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
34274.0, 101328.0, 14400.0, nan, 14400.0, 14400.0, 3575.0, 14400.0, 14400.0,
104231.0, 14400.0, 14400.0, 14400.0, 14400.0, 14593.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 8030.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 13380.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 2797.0,
707.0, 14400.0, 14400.0, 14401.0, 14400.0, 14400.0, 14400.0, 14401.0, 14400.0,
14399.0, 14400.0, 14400.0, 14401.0, 4242.0, 14400.0, 14400.0, 14400.0, 14400.0,
```

```
14400.0, 14400.0, 14400.0, 14401.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14399.0, 14400.0, 14400.0, 14400.0, 14401.0, 14400.0,
10762.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 10265.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0]
```



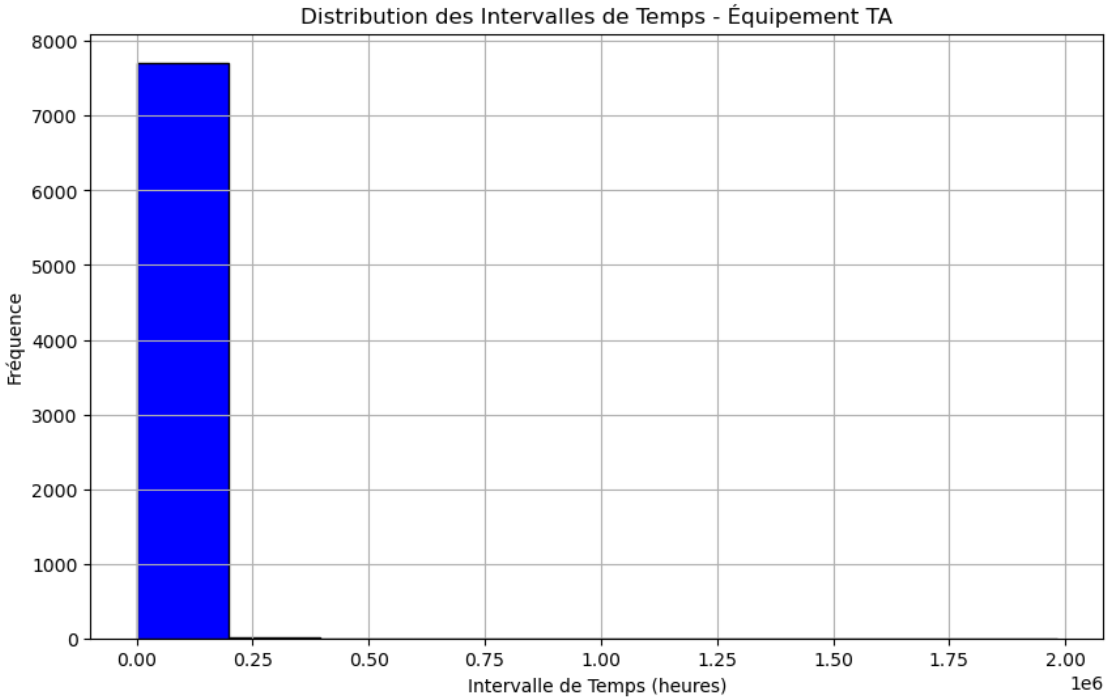

```
count    7.660000e+03
mean     9.145862e+03
std      3.625203e+04
min      0.000000e+00
25%      3.600000e+03
50%      5.400000e+03
75%      1.440000e+04
max      1.982545e+06
Name: time_diff, dtype: float64
Seuil proposé basé sur IQR pour l'équipement TA: 30600.0
Nombre d'anomalies détectées pour l'équipement TA: 29
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7727.csv
[1278.0, 14400.0, 33670.0, 43200.0, 14400.0, 14400.0, 3575.0, 14400.0, 14400.0,
14400.0, 14400.0, 1248.0, 14400.0, 14400.0, 14400.0, 104231.0, 14400.0, 14400.0,
```

```
1331.0, 14400.0, 14400.0, 14400.0, 14400.0, 14399.0, 14400.0, nan, 604.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14401.0, 14401.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 1009.0, 14593.0, 14400.0,
101328.0, 5388.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 737.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 8030.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
2797.0, 14400.0, 13380.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14401.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 4242.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 10763.0, 14400.0, 14400.0, 707.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14401.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0]
```



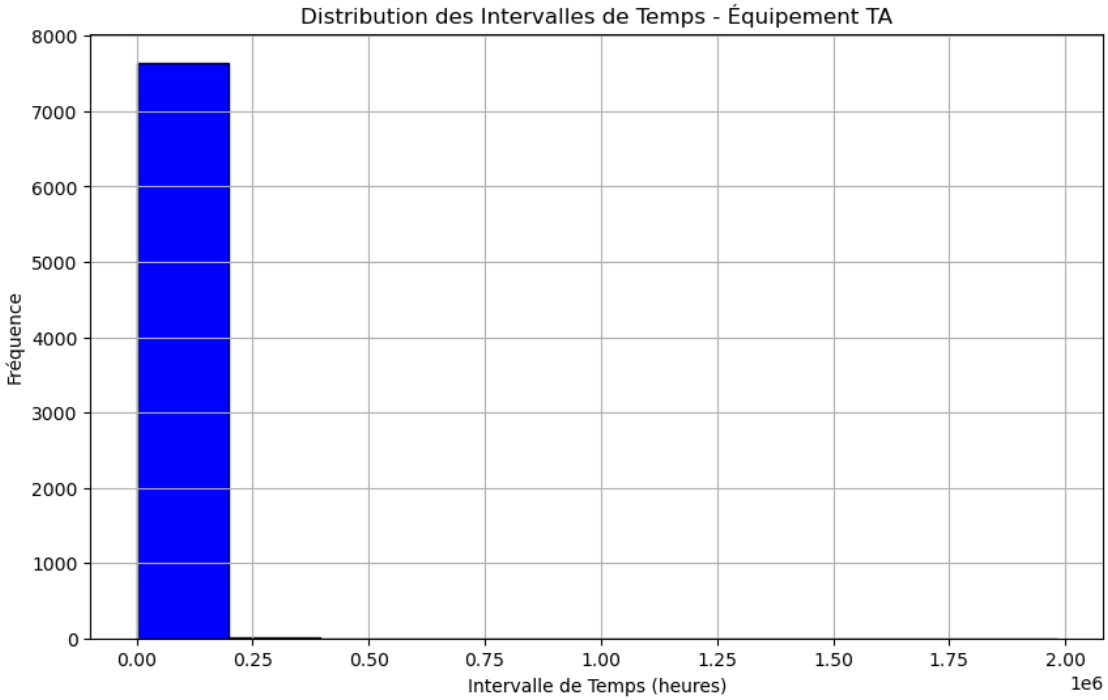

```
count    7.743000e+03
mean     9.047824e+03
std      3.657896e+04
```

```
min      0.000000e+00
25%      3.599000e+03
50%      5.400000e+03
75%      1.440000e+04
max      1.982545e+06
Name: time_diff, dtype: float64
Seuil proposé basé sur IQR pour l'équipement TA: 30601.5
Nombre d'anomalies détectées pour l'équipement TA: 37
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7728.csv
[14400.0, 33670.0, 14400.0, 14400.0, 1248.0, 3575.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, nan, 14401.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
1009.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14401.0, 14400.0, 14400.0,
1278.0, 14400.0, 14400.0, 14400.0, 5388.0, 14400.0, 14400.0, 14400.0, 14593.0,
604.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 737.0, 14399.0, 28800.0,
14400.0, 101328.0, 104231.0, 1331.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 8030.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14401.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14401.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 13380.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 4242.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 10763.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 2797.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 707.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 10265.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0]
```



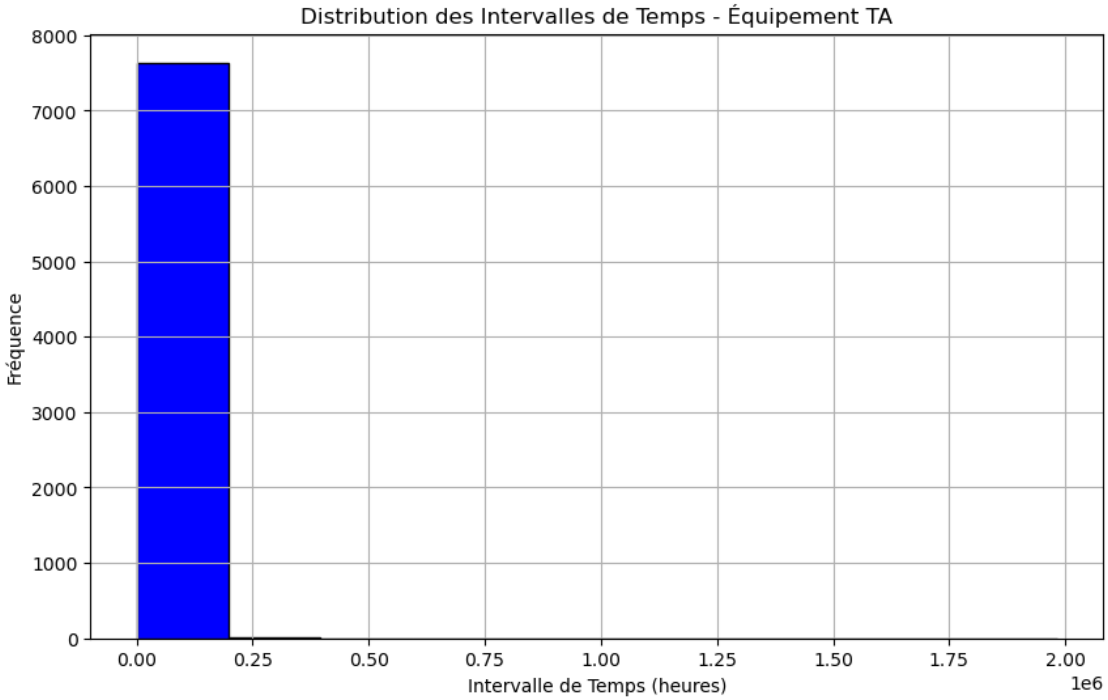

```
count    7.723000e+03
mean     9.103419e+03
std      3.620923e+04
min      0.000000e+00
25%      3.600000e+03
50%      5.400000e+03
75%      1.440000e+04
max      1.982545e+06
Name: time_diff, dtype: float64
Seuil proposé basé sur IQR pour l'équipement TA: 30600.0
Nombre d'anomalies détectées pour l'équipement TA: 34
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7729.csv
[0.0, 5388.0, 14400.0, 14400.0, 3575.0, 14400.0, 14400.0, 14400.0, 104231.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 0.0, 14400.0, 14400.0,
14400.0, 1278.0, 14400.0, 14400.0, 14401.0, 14400.0, 1331.0, nan, 14400.0,
14400.0, 14593.0, 14401.0, 737.0, 14400.0, 1248.0, 14400.0, 14400.0, 14400.0,
14400.0, 14399.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 101328.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
8030.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 13380.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
707.0, 14400.0, 14400.0, 14400.0, 2797.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 4242.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14401.0,
```

```
14400.0, 14400.0, 14401.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 10763.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 10265.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0]
```



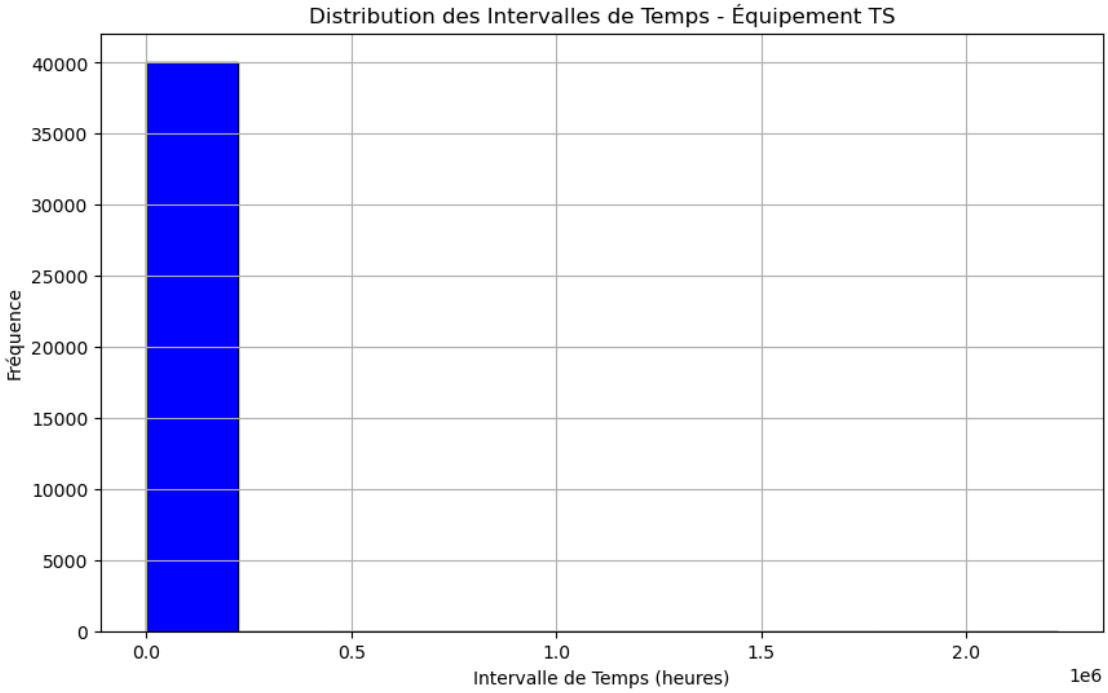

```
count    7.657000e+03
mean     9.145543e+03
std      3.635287e+04
min      0.000000e+00
25%      3.600000e+03
50%      5.400000e+03
75%      1.440000e+04
max      1.982545e+06
Name: time_diff, dtype: float64
Seuil proposé basé sur IQR pour l'équipement TA: 30600.0
Nombre d'anomalies détectées pour l'équipement TA: 35
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TA\7730.csv
[14400.0, 14400.0, 104231.0, nan, 101328.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14401.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 3575.0,
```

```
14400.0, 14400.0, 14400.0, 5388.0, 14400.0, 14400.0, 8030.0, 737.0, 1009.0,
14400.0, 14400.0, 14400.0, 1331.0, 14400.0, 34274.0, 14400.0, 14400.0, 14400.0,
14400.0, 14399.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
1278.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14401.0, 14400.0, 14400.0,
14401.0, 14400.0, 14400.0, 14401.0, 14400.0, 14593.0, 14400.0, 14400.0, 14400.0,
4242.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 13380.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 10763.0, 14400.0, 1248.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 707.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 2797.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
10265.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0, 14400.0,
14400.0, 14400.0]
```

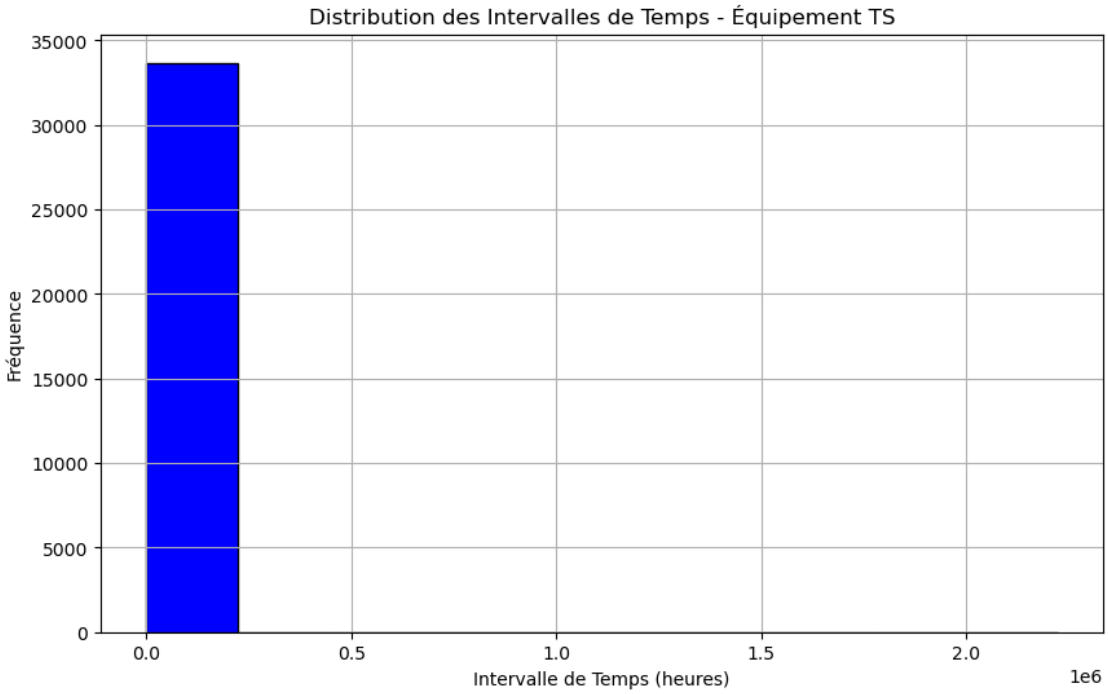



```
count    7.648000e+03
mean     9.161624e+03
std      3.638472e+04
```

```
min       0.000000e+00
25%       3.600000e+03
50%       5.400000e+03
75%       1.440000e+04
max       1.982545e+06
Name: time_diff, dtype: float64
Seuil proposé basé sur IQR pour l'équipement TA: 30600.0
Nombre d'anomalies détectées pour l'équipement TA: 35
Seuil moyen pour l'équipement TA: 31275.0
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TS\37482.xlsx
[1800.0, 1800.0, 0.0, 1800.0, 300.0, 1800.0, 0.0, 1800.0, 0.0, 300.0, 768094.0,
0.0, 1800.0, 1800.0, 0.0, 0.0, 1800.0, 1800.0, 1800.0, 0.0, 1800.0, 0.0, 0.0,
0.0, 1800.0, 1800.0, 1800.0, 0.0, 1800.0, 1691.0, 1800.0, 1800.0, 0.0, 1800.0,
1800.0, 0.0, 1800.0, 0.0, 1800.0, 0.0, 0.0, 0.0, 1800.0, 1800.0, 1800.0, 0.0,
1800.0, 0.0, 0.0, 0.0, 1800.0, 0.0, 0.0, 0.0, 1800.0, 655.0, 0.0, 1800.0, 0.0,
0.0, 0.0, 0.0, 1800.0, 0.0, 1800.0, 1800.0, 1800.0, 1800.0, 0.0, 0.0, 1800.0,
0.0, 0.0, 1800.0, 0.0, 1800.0, 0.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 0.0, 0.0, 1800.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1800.0, 0.0, 1800.0,
0.0, 0.0, 0.0, 0.0, 0.0, 1800.0, 1800.0, 0.0, 1800.0, 1800.0, 0.0, 1800.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 1800.0, 0.0, 1800.0, 0.0, 1800.0, 1800.0, 0.0, 1800.0, 0.0,
1800.0, 1800.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1800.0, 0.0, 0.0, 0.0, 0.0, 0.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 0.0, 1800.0, 1800.0, 1800.0, 1800.0,
0.0, 1800.0, 1800.0, 1800.0, 1800.0, 0.0, 1800.0, 1800.0, 0.0, 0.0, 0.0, 1800.0,
1800.0, 0.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 0.0, 0.0, 1800.0,
0.0, 0.0, 1800.0, 0.0, 0.0, 1800.0, 1800.0, 1800.0, 0.0, 0.0, 0.0, 1800.0, 0.0,
0.0, 0.0, 0.0, 1800.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1800.0, 0.0, 0.0, 1800.0,
1800.0, 1800.0, 0.0, 1800.0, 1800.0, 0.0]
```

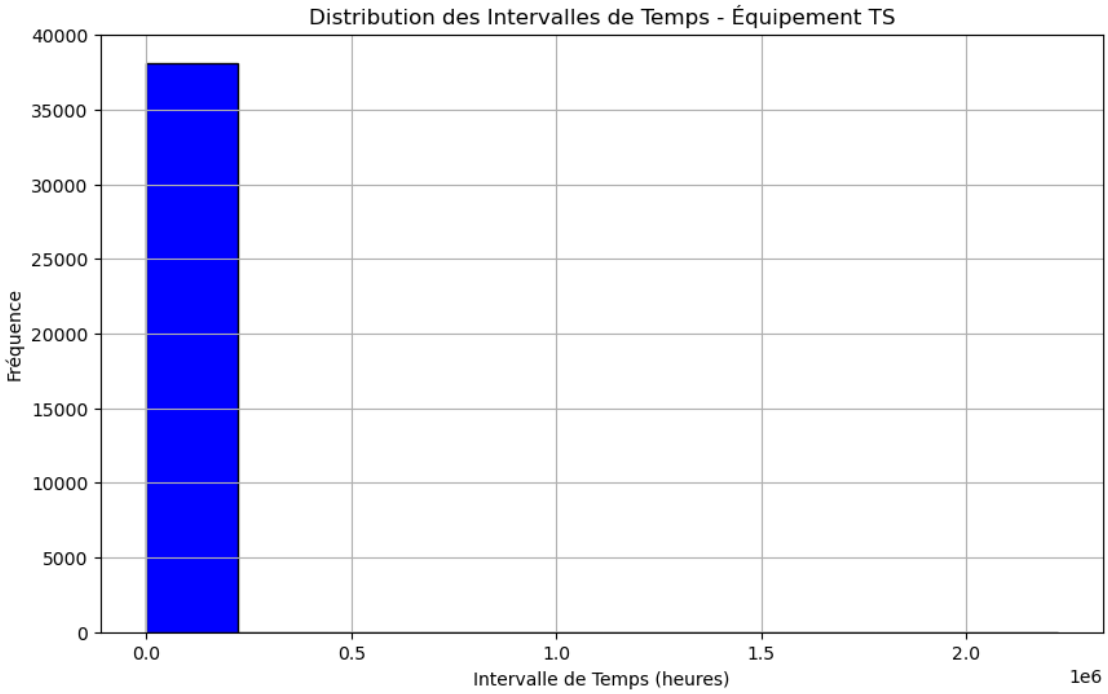


```
count    4.007000e+04
mean     6.380648e+02
std      1.203697e+04
min      0.000000e+00
25%      6.000000e+02
50%      6.000000e+02
75%      6.000000e+02
max      2.223962e+06
Name: time_diff, dtype: float64
Seuil proposé basé sur IQR pour l'équipement TS: 600.0
Nombre d'anomalies détectées pour l'équipement TS: 1475
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TS\37483.xlsx
[1800.0, 1800.0, 1800.0, 1800.0, 768094.0, 203.0, 2703.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1309.0, 1800.0, 1800.0, nan,
300.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 300.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 655.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1691.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0]
```



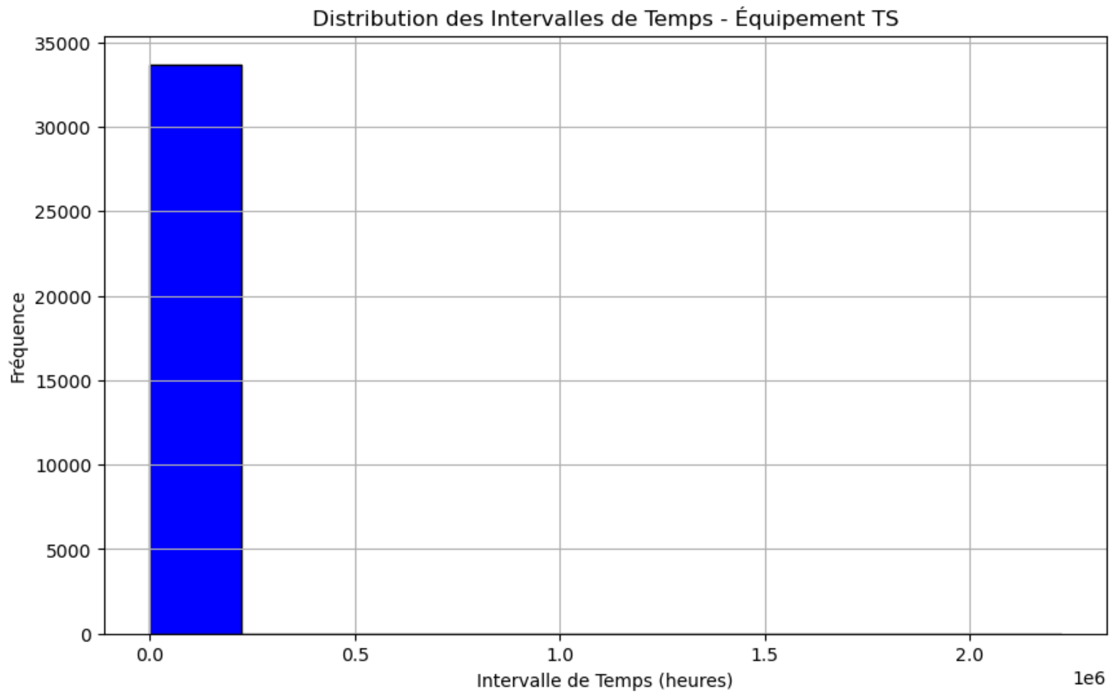

```
count    3.368200e+04
mean     7.591846e+02
std      1.312413e+04
min      0.000000e+00
25%      6.000000e+02
50%      6.000000e+02
75%      6.000000e+02
max      2.223962e+06
Name: time_diff, dtype: float64
Seuil proposé basé sur IQR pour l'équipement TS: 600.0
Nombre d'anomalies détectées pour l'équipement TS: 1478
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TS\37484.xlsx
[0.0, 0.0, 1800.0, 0.0, 1800.0, 1800.0, 1800.0, 0.0, 0.0, 0.0, 1800.0, 0.0, 0.0,
1800.0, 0.0, 0.0, 203.0, 1800.0, 1800.0, 0.0, 0.0, 0.0, 1800.0, 1800.0, 0.0,
1800.0, 0.0, 0.0, 0.0, 1800.0, 1800.0, 1309.0, 1800.0, 1800.0, 0.0, 1800.0, 0.0,
1800.0, 1800.0, 0.0, 1800.0, 1800.0, 1691.0, 0.0, 0.0, 1800.0, 1800.0, 0.0, nan,
0.0, 0.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 0.0, 1800.0, 0.0, 1800.0, 0.0,
0.0, 1800.0, 1800.0, 0.0, 0.0, 1800.0, 1800.0, 1800.0, 0.0, 1800.0, 1800.0,
1800.0, 1800.0, 0.0, 0.0, 1800.0, 0.0, 1800.0, 1800.0, 1800.0, 1800.0, 0.0, 0.0,
0.0, 1800.0, 1800.0, 1800.0, 0.0, 0.0, 1800.0, 0.0, 1800.0, 1800.0, 1800.0, 0.0,
0.0, 1800.0, 1800.0, 1800.0, 1800.0, 0.0, 0.0, 1800.0, 0.0, 0.0, 0.0, 0.0, 0.0,
1800.0, 0.0, 1800.0, 1800.0, 1800.0, 0.0, 1800.0, 0.0, 0.0, 1800.0, 0.0, 1800.0,
0.0, 1800.0, 0.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 0.0, 1800.0, 0.0, 0.0, 0.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 0.0, 1800.0, 1800.0, 1800.0, 0.0, 0.0, 0.0, 0.0, 1800.0, 1800.0,
```

```
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 0.0, 0.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 0.0, 0.0, 0.0, 1800.0, 1800.0, 0.0, 0.0, 0.0, 0.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 0.0, 1800.0, 0.0, 1800.0, 1800.0, 0.0,
1800.0, 0.0, 1800.0, 0.0, 1800.0, 1800.0, 0.0, 1800.0, 0.0, 1800.0, 1800.0,
1800.0, 0.0]
```



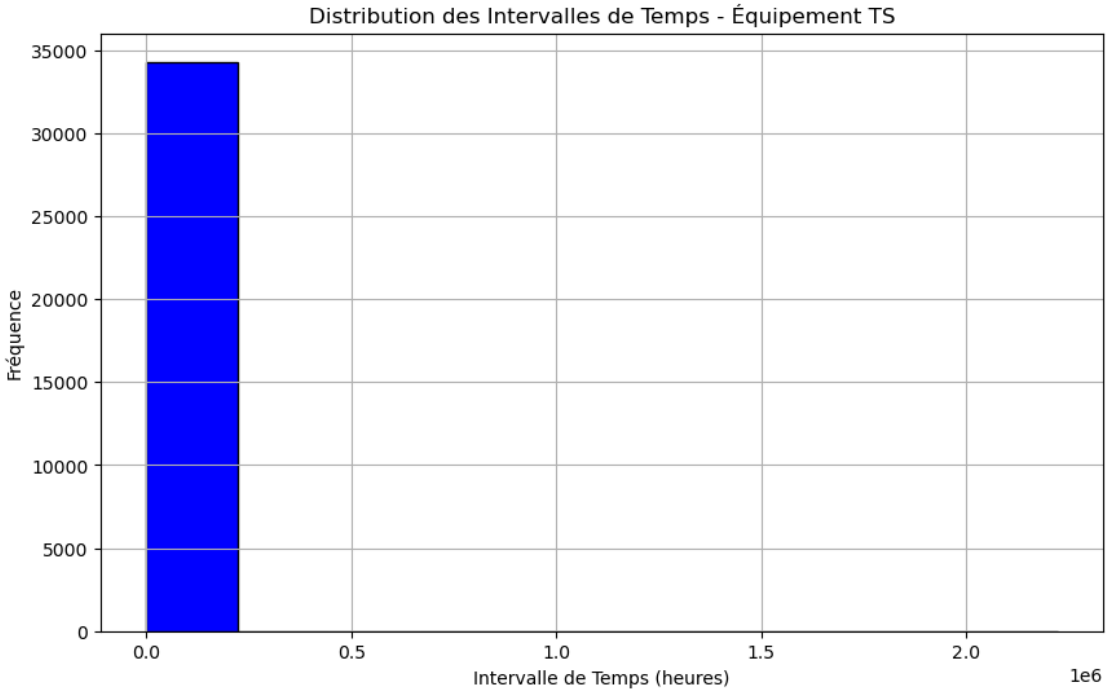

```
count    3.813800e+04
mean     6.704823e+02
std      1.233715e+04
min      0.000000e+00
25%      6.000000e+02
50%      6.000000e+02
75%      6.000000e+02
max      2.223962e+06
Name: time_diff, dtype: float64
Seuil proposé basé sur IQR pour l'équipement TS: 600.0
Nombre d'anomalies détectées pour l'équipement TS: 1478
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TS\37485.xlsx
[1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 2703.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
768094.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 300.0, nan, 1800.0,
1800.0, 1800.0, 1800.0, 1691.0, 1309.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 300.0, 1800.0, 1800.0, 1800.0, 655.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 203.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
```

```
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0]
```



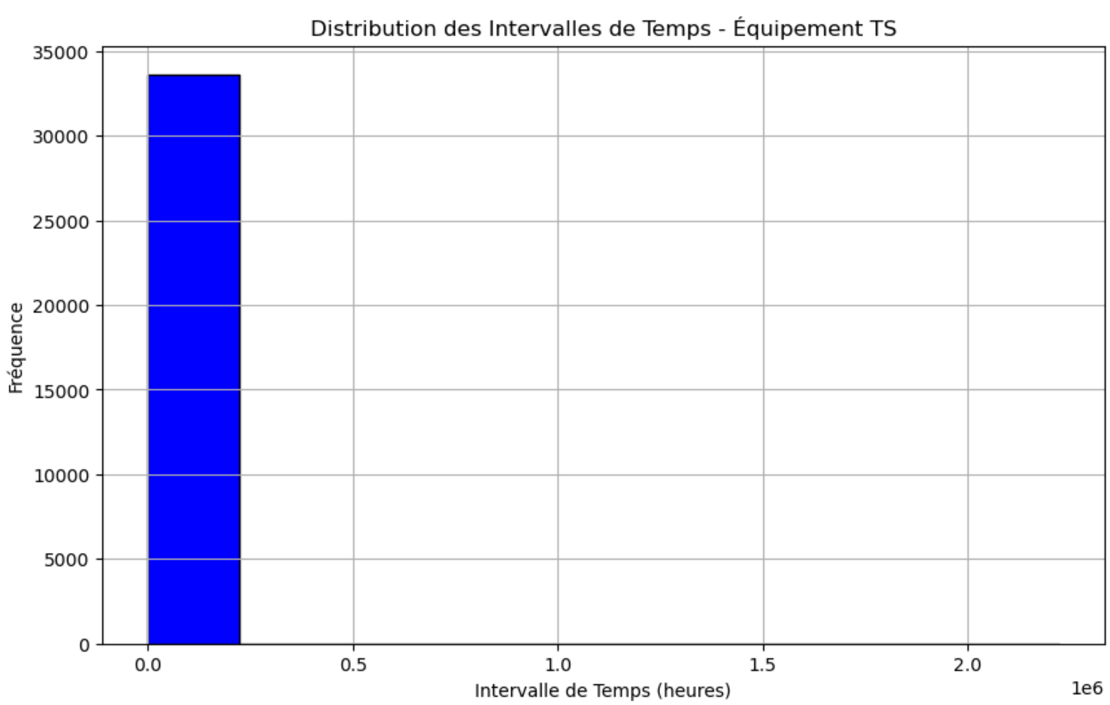

```
count    3.370100e+04
mean     7.621749e+02
std      1.325037e+04
min      0.000000e+00
25%      6.000000e+02
50%      6.000000e+02
75%      6.000000e+02
max      2.223962e+06
Name: time_diff, dtype: float64
Seuil proposé basé sur IQR pour l'équipement TS: 600.0
```

```
Nombre d'anomalies détectées pour l'équipement TS: 1491
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TS\37486.xlsx
[1800.0, 1800.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1800.0, 1800.0, 1800.0, 1800.0,
0.0, 1800.0, 1800.0, 1800.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1800.0, 1800.0, 1800.0,
0.0, 768094.0, 1800.0, 0.0, 1800.0, 1800.0, 0.0, 1800.0, 0.0, 0.0, 0.0, 655.0,
1800.0, 0.0, 1800.0, 1800.0, 0.0, 0.0, 1800.0, 1800.0, 1691.0, 1800.0, 1800.0,
0.0, 0.0, 1800.0, 0.0, 1800.0, 1800.0, 0.0, 0.0, 0.0, 0.0, 1800.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 1800.0, 1800.0, 0.0, 1800.0, 0.0, 1800.0, 0.0, 1800.0, 0.0,
0.0, 1800.0, 0.0, 0.0, 1800.0, 1800.0, 0.0, 1800.0, 0.0, 0.0, 1800.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1800.0, 0.0, 1800.0, 1800.0, 1800.0, 0.0, 1800.0,
1800.0, 0.0, 1800.0, 1800.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1800.0, 1800.0, 0.0,
1800.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1800.0, 1800.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
1800.0, 1800.0, 0.0, 0.0, 1800.0, 1800.0, 1800.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
1800.0, 0.0, 0.0, 1800.0, 1800.0, 1800.0, 0.0, 1800.0, 1800.0, 0.0, 0.0, 0.0,
0.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 0.0, 0.0, 0.0, 1800.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
1800.0, 0.0, 0.0, 0.0, 0.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 0.0, 1800.0,
0.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 0.0, 0.0, 0.0, 0.0, 1800.0, 0.0,
1800.0, 1800.0]
```

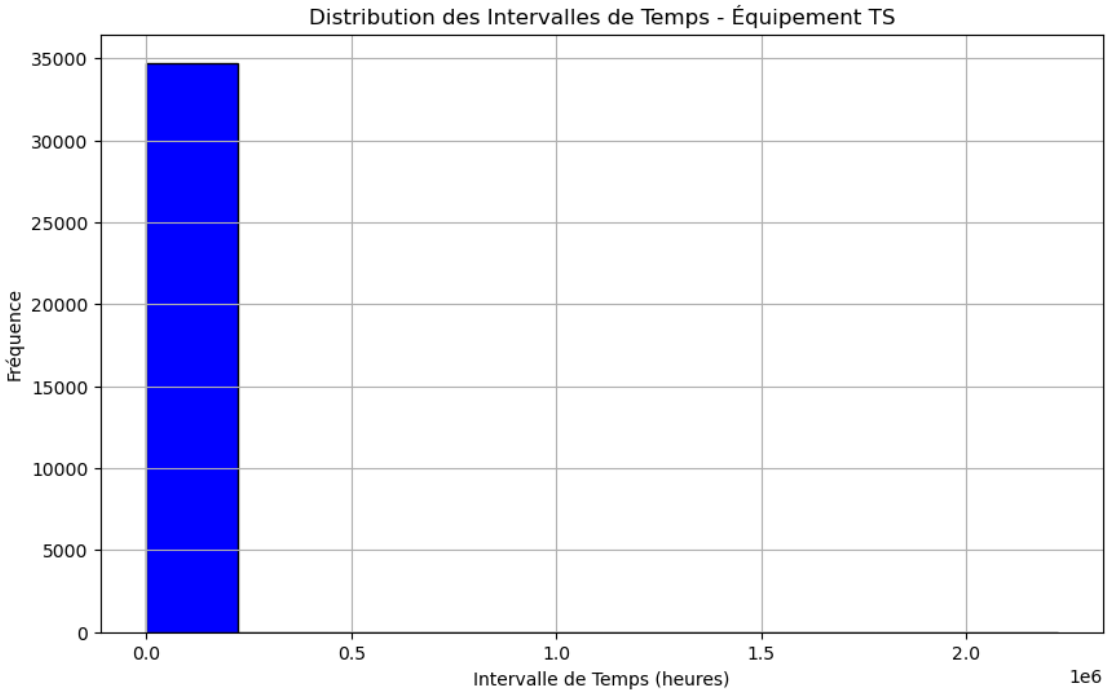



```
count    3.431400e+04
mean     7.485590e+02
std      1.313108e+04
min      0.000000e+00
25%      6.000000e+02
```

```
50%      6.000000e+02
75%      6.000000e+02
max      2.223962e+06
Name: time_diff, dtype: float64
Seuil proposé basé sur IQR pour l'équipement TS: 600.0
Nombre d'anomalies détectées pour l'équipement TS: 1473
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TS\37487.xlsx
[0.0, 1800.0, 0.0, 300.0, 1800.0, 1800.0, 300.0, 0.0, 1800.0, 1800.0, 0.0,
1800.0, 1800.0, 1800.0, 0.0, 0.0, 0.0, 1800.0, 203.0, 0.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 0.0, 1800.0, 0.0, 1800.0, 1800.0, nan,
0.0, 1800.0, 0.0, 1800.0, 2703.0, 0.0, 1800.0, 0.0, 1800.0, 0.0, 0.0, 1800.0,
0.0, 0.0, 0.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 0.0, 0.0, 1691.0, 0.0,
1800.0, 1800.0, 1800.0, 1800.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
0.0, 1800.0, 1800.0, 0.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 0.0, 1800.0, 1800.0, 0.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0]
```

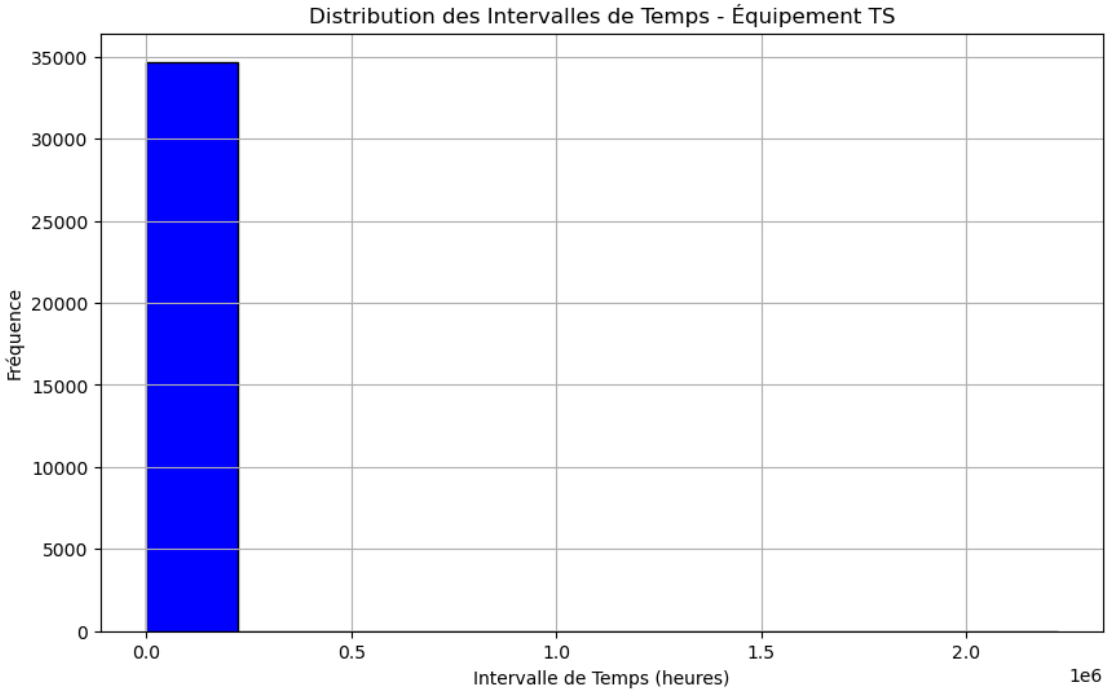


```
count    3.363100e+04
mean     7.611922e+02
std      1.326037e+04
min      0.000000e+00
25%      6.000000e+02
50%      6.000000e+02
75%      6.000000e+02
max      2.223962e+06
Name: time_diff, dtype: float64
Seuil proposé basé sur IQR pour l'équipement TS: 600.0
Nombre d'anomalies détectées pour l'équipement TS: 1477
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TS\37488.xlsx
[0.0, 1800.0, 0.0, 1800.0, 1800.0, 0.0, 1800.0, 1800.0, 1800.0, 0.0, 1800.0,
0.0, 1800.0, 0.0, 1800.0, 0.0, 1800.0, 0.0, 0.0, 0.0, 1800.0, 0.0, 1800.0, 0.0,
nan, 0.0, 0.0, 1800.0, 1800.0, 0.0, 0.0, 1800.0, 0.0, 1800.0, 0.0, 0.0, 1800.0,
0.0, 1800.0, 0.0, 1800.0, 300.0, 0.0, 0.0, 300.0, 1800.0, 0.0, 1800.0, 0.0, 0.0,
0.0, 1800.0, 1800.0, 0.0, 0.0, 1800.0, 1800.0, 1800.0, 0.0, 1800.0, 1800.0, 0.0,
0.0, 1800.0, 1800.0, 1800.0, 0.0, 1800.0, 0.0, 0.0, 0.0, 1800.0, 1800.0, 1800.0,
0.0, 1800.0, 0.0, 1800.0, 1800.0, 0.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0,
0.0, 0.0, 0.0, 0.0, 1800.0, 0.0, 1800.0, 1800.0, 1800.0, 0.0, 0.0, 0.0, 1800.0,
0.0, 0.0, 0.0, 1800.0, 0.0, 1800.0, 1800.0, 1800.0, 0.0, 1800.0, 0.0, 1800.0,
1800.0, 1800.0, 0.0, 0.0, 0.0, 0.0, 1800.0, 0.0, 0.0, 1800.0, 1800.0, 0.0,
1800.0, 0.0, 1800.0, 1800.0, 1800.0, 0.0, 1800.0, 1800.0, 1800.0, 1800.0, 0.0,
1800.0, 0.0, 0.0, 1800.0, 0.0, 0.0, 1800.0, 0.0, 1800.0, 1800.0, 1800.0, 1800.0,
1800.0, 0.0, 0.0, 1800.0, 1800.0, 1800.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1800.0,
0.0, 1800.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1800.0, 0.0, 0.0, 0.0, 1800.0, 1800.0,
1800.0, 1800.0, 0.0, 1800.0, 0.0, 0.0, 1800.0, 0.0, 0.0, 1800.0, 1800.0, 1800.0,
1800.0, 0.0, 1800.0, 1800.0, 1800.0, 1800.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
1800.0, 0.0, 1800.0, 0.0, 1800.0]
```




```
count    3.472300e+04
mean     7.372535e+02
std      1.304933e+04
min      0.000000e+00
25%      6.000000e+02
50%      6.000000e+02
75%      6.000000e+02
max      2.223962e+06
Name: time_diff, dtype: float64
Seuil proposé basé sur IQR pour l'équipement TS: 600.0
Nombre d'anomalies détectées pour l'équipement TS: 1478
Traitement du fichier: /Users/Ilyasse/Python projects/Equipement TS\37489.xlsx
[0.0, 203.0, 2703.0, 1800.0, 1800.0, 1800.0, 1800.0, 1800.0, 0.0, 0.0, 1800.0,
1800.0, 1800.0, 0.0, 0.0, 1800.0, 1800.0, 1309.0, 0.0, 0.0, 1800.0, 0.0, 1800.0,
0.0, 1800.0, 768094.0, 1800.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1800.0, 1800.0, 1800.0,
0.0, 1800.0, 1800.0, 1800.0, 0.0, 1800.0, 0.0, 1800.0, 0.0, 1800.0, 1800.0, 0.0,
1800.0, 1800.0, 0.0, 1800.0, 1800.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1800.0, 0.0, 0.0,
0.0, 1800.0, 0.0, 0.0, 1800.0, 1800.0, 0.0, 1800.0, 0.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 0.0, 0.0, 0.0, 0.0, 1800.0, 1800.0, 0.0, 1800.0, 0.0, 1800.0,
1800.0, 1800.0, 0.0, 1800.0, 1800.0, 0.0, 0.0, 0.0, 1800.0, 1800.0, 0.0, 0.0,
0.0, 1800.0, 0.0, 1800.0, 1800.0, 0.0, 1800.0, 1800.0, 0.0, 0.0, 1800.0, 1800.0,
0.0, 1800.0, 0.0, 1800.0, 0.0, 0.0, 0.0, 1800.0, 0.0, 1800.0, 1800.0, 1800.0,
1800.0, 1800.0, 0.0, 0.0, 0.0, 1800.0, 0.0, 1800.0, 1800.0, 1800.0, 1800.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1800.0, 1800.0, 0.0, 1800.0, 1800.0, 0.0,
1800.0, 1800.0, 0.0, 1800.0, 0.0, 1800.0, 1800.0, 0.0, 0.0, 0.0, 1800.0, 0.0,
```

```
0.0, 1800.0, 1800.0, 1800.0, 0.0, 1800.0, 1800.0, 0.0, 1800.0, 1800.0, 0.0,
1800.0, 1800.0, 1800.0, 0.0, 1800.0, 1800.0, 1800.0, 0.0, 1800.0, 0.0, 0.0, 0.0,
1800.0, 1800.0, 0.0, 0.0, 0.0, 0.0, 1800.0, 1800.0, 0.0, 1800.0, 0.0, 0.0,
1800.0, 0.0, 0.0, 0.0, 0.0, 1800.0, 1800.0, 1800.0]
```




```
count    3.468300e+04
mean     7.386055e+02
std      1.305549e+04
min      0.000000e+00
25%      6.000000e+02
50%      6.000000e+02
75%      6.000000e+02
max      2.223962e+06
Name: time_diff, dtype: float64
Seuil proposé basé sur IQR pour l'équipement TS: 600.0
Nombre d'anomalies détectées pour l'équipement TS: 1472
Seuil moyen pour l'équipement TS: 600.0
Seuil final pour TA: 31275.0
Seuil final pour TS: 600.0
```

5 Interquartile range

[73]:
```
# Function to analyze equipment data
def analyze_anomalie(data, equipment_name):
    # Conversion de la colonne 'Date of Creation' en type datetime
    data['Date of Creation'] = pd.to_datetime(data['Date of Creation'])
    # Trier les données par date
    data.sort_values(by='Date of Creation', inplace=True)
    # Calculer les intervalles de temps entre chaque point de données consécutif
    data['time_diff'] = data['Date of Creation'].diff().dt.total_seconds() /␣
↪3600  # Différence en heures

    # Utiliser l'IQR pour identifier les outliers
    Q1 = np.percentile(data['time_diff'].dropna(), 25)
    Q3 = np.percentile(data['time_diff'].dropna(), 75)
    IQR = Q3 - Q1

    seuil = Q3 + 1.5*IQR
    #euil basé sur l'IQR
    print(f'Seuil proposé basé sur IQR pour l\'équipement {equipment_name}:',␣
↪seuil)

    # Afficher les données avec anomalies
    # Ensure the 'Date of Creation' column is in datetime format
    data['Date of Creation'] = pd.to_datetime(data['Date of Creation'])
    data = data.sort_values(by='Date of Creation')

    anomalies = data[data['time_diff'] > seuil]

    # Visualize NGV evolution over time with anomalies and interpolations
    plt.figure(figsize=(14, 7))

    plt.plot(data['Date of Creation'], data['NGV'], marker='o', linestyle='-',␣
↪color='b', label='Original Data')


    for anomaly_index in anomalies.index:
        if anomaly_index in data.index:
            plt.axvline(data.loc[anomaly_index, 'Date of Creation'],␣
↪color='red', linestyle='--', linewidth=1, label='Début de Coupure' if␣
↪anomaly_index == anomalies.index[0] else "")
            next_index = data.index.get_loc(anomaly_index) + 1
        if next_index < len(data):
            plt.axvline(data.iloc[next_index]['Date of Creation'], color='red',␣
↪linestyle='-', linewidth=1, label='Fin de Coupure' if anomaly_index ==␣
↪anomalies.index[0] else "")
```

```
    plt.xlabel('Date et Heure')
    plt.ylabel('NGV')
    plt.title(f"Évolution de NGV pour {equipment_name}")
    plt.legend()
    plt.grid(True)
    plt.show()
```

6 Slope changement

[74]:
```
def detect_anomalies_slope(data, equipment_name):
    # Assurez-vous que la colonne 'Date of Creation' est au format datetime
    data['Date of Creation'] = pd.to_datetime(data['Date of Creation'],␣
 ↪errors='coerce')

    # Supprimez les lignes avec des 'Date of Creation' invalides
    data = data.dropna(subset=['Date of Creation'])

    # Calculez les différences successives dans NGV
    data['NGV Diff'] = data['NGV'].diff()
    data['Time Diff'] = data['Date of Creation'].diff().dt.total_seconds() /␣
 ↪3600  # Différence en heures

    # Calculez la pente (changement dans NGV par unité de temps)
    data['Slope'] = data['Time Diff']



    # Détectez les anomalies
    anomalies = data[(data['Slope'] > 400)]


    # Visualisez l'évolution de NGV au fil du temps avec les anomalies et les␣
 ↪interpolations
    # Create a scatter plot for all data points
    fig = px.scatter(data, x=data['Date of Creation'], y=data['NGV'], title='NGV␣
 ↪Over Time')

# Add red color for anomalies
    fig.add_scatter(x=anomalies['Date of Creation'], y=anomalies['NGV'],␣
 ↪mode='markers', marker=dict(color='red'), name='Anomalies')



    fig = go.Figure()

    # Add trace for NGV data
    fig.add_trace(go.Scatter(x=data['Date of Creation'],
```

```
                            y=data['NGV'],
                            mode='markers+lines',
                            name='NGV Équipement TA',
                            marker=dict(size=5),
                            line=dict(width=2)))
    # Ajouter des points rouges pour les anomalies
    fig.add_trace(go.Scatter(
        x=anomalies['Date of Creation'],
        y=anomalies['NGV'],
        mode='markers',
        marker=dict(color='red'),
        name='Anomalies'
    ))

    # Update layout of the plot
    fig.update_layout(title='NGV en fonction de la Date de Creation',
                      xaxis_title='Date de Creation',
                      yaxis_title='NGV',
                      template='plotly_white',
                      legend_title='Legend')

    # Show the plot
    fig.show()

    # Imprimez les statistiques des pentes
    print(f"Statistiques des Pentes pour {equipment_name}:")
    print(data['Slope'].describe())

    print(f"Nombre d'anomalies détectées pour {equipment_name}:␣
↪{len(anomalies)}")


# Chemins des dossiers contenant les fichiers Excel pour les équipements TA et TS
dossier_ta = '/Users/Ilyasse/Python projects/Equipement TA/7710.csv'
dossier_ts = '/Users/Ilyasse/Python projects/Equipement TS'
donn= pd.read_csv(dossier_ta)
detect_anomalies_slope(donn, '7710')
```

```
Statistiques des Pentes pour 7710:
count    7388.000000
mean        2.631708
std        53.454673
min      -640.133056
25%        -6.000000
50%         1.500000
75%         9.000278
max       616.133056
Name: Slope, dtype: float64
```

```
Nombre d'anomalies détectées pour 7710: 13
```

[75]:
```
dossier_ta = '/Users/Ilyasse/Python projects/Equipement TA/7712.csv'
donn= pd.read_csv(dossier_ta)
detect_anomalies_slope(donn, '7712')
```

```
Statistiques des Pentes pour 7712:
count    7444.000000
mean        2.597365
std        62.659984
min      -658.706944
25%        -6.000000
50%         1.500000
75%         8.000278
max       664.133056
Name: Slope, dtype: float64
Nombre d'anomalies détectées pour 7712: 21
```

[76]:
```
dossier_ta = '/Users/Ilyasse/Python projects/Equipement TA/7721.csv'
donn= pd.read_csv(dossier_ta)
detect_anomalies_slope(donn, '7721')
```

```
Statistiques des Pentes pour 7721:
count    6573.000000
mean        2.952360
std        68.283860
min      -552.497222
25%        -4.000000
50%         1.500000
75%         6.500000
max      2780.781111
Name: Slope, dtype: float64
Nombre d'anomalies détectées pour 7721: 13
```

[77]:
```
dossier_ts = '/Users/Ilyasse/Python projects/Equipement TS/37485.xlsx'
donn= pd.read_excel(dossier_ts)
detect_anomalies_slope(donn, '37482')
```

```
Statistiques des Pentes pour 37482:
count    33701.000000
mean         0.205305
std          7.285421
min       -619.500000
25%         -0.500000
50%          0.166667
75%          0.666667
max        619.333333
Name: Slope, dtype: float64
Nombre d'anomalies détectées pour 37482: 2
```

7 Slope optimized

[78]:
```
def detect_anomalies_pente(data, equipment_name):
    # Assurez-vous que la colonne 'Date of Creation' est au format datetime
    data['Date of Creation'] = pd.to_datetime(data['Date of Creation'],␣
 ↪errors='coerce')

    # Supprimez les lignes avec des 'Date of Creation' invalides
    data = data.dropna(subset=['Date of Creation'])

    # Calculez les différences successives dans NGV
    data['NGV Diff'] = data['NGV'].diff()
    data['Time Diff'] = data['Date of Creation'].diff().dt.total_seconds() /␣
 ↪3600  # Différence en heures

    # Calculez la pente (changement dans NGV par unité de temps)
    data['Slope'] = data['NGV Diff']/data['Time Diff']


      # Seuil supérieur

    duration_hours=2


    ## Créez une fenêtre glissante pour vérifier la constance de la pente sur la␣
 ↪durée définie
    data['Is Constant Slope'] = data['Slope'].
 ↪rolling(window=int(duration_hours), min_periods=1).apply(lambda x: np.all(np.
 ↪isclose(x, x.mean(), atol=1e-5)), raw=True)

    # Filtrer les périodes où la pente est constante
    anomalies = data[data['Is Constant Slope'] == 1]
    fig = px.scatter(data, x=data['Date of Creation'], y=data['NGV'], title='NGV␣
 ↪Over Time')

# Add red color for anomalies
    fig.add_scatter(x=anomalies['Date of Creation'], y=anomalies['NGV'],␣
 ↪mode='markers', marker=dict(color='red'), name='Anomalies')


    fig = go.Figure()

    # Add trace for NGV data
    fig.add_trace(go.Scatter(x=data['Date of Creation'],
                             y=data['NGV'],
                             mode='markers+lines',
                             name='NGV Équipement TA',
```

```
                            marker=dict(size=5),
                            line=dict(width=2)))
    # Ajouter des points rouges pour les anomalies
    fig.add_trace(go.Scatter(
        x=anomalies['Date of Creation'],
        y=anomalies['NGV'],
        mode='markers',
        marker=dict(color='red'),
        name='Anomalies'
    ))

    # Update layout of the plot
    fig.update_layout(title='NGV en fonction de la Date de Creation',
                      xaxis_title='Date de Creation',
                      yaxis_title='NGV',
                      template='plotly_white',
                      legend_title='Legend')

    # Show the plot
    fig.show()

    # Imprimez les statistiques des pentes
    print(f"Statistiques des Pentes pour {equipment_name}:")
    print(data['Slope'].describe())



# Chemins des dossiers contenant les fichiers Excel pour les équipements TA et TS
dossier_ta = '/Users/Ilyasse/Python projects/Equipement TA/7710.csv'
dossier_ts = '/Users/Ilyasse/Python projects/Equipement TS'
donn= pd.read_csv(dossier_ta)
detect_anomalies_pente(donn, '7710')
```

```
Statistiques des Pentes pour 7710:
count    7380.000000
mean        0.001667
std         0.130060
min        -3.036000
25%        -0.008167
50%         0.000000
75%         0.007500
max         3.716000
Name: Slope, dtype: float64
```

[79]:
```
# Chemins des dossiers contenant les fichiers Excel pour les équipements TA et TS
dossier_ta = '/Users/Ilyasse/Python projects/Equipement TA/7712.csv'
dossier_ts = '/Users/Ilyasse/Python projects/Equipement TS'
```

```
donn= pd.read_csv(dossier_ta)
detect_anomalies_pente(donn, '7712')
```

```
Statistiques des Pentes pour 7712:
count    7437.000000
mean        0.002350
std         0.096495
min        -1.272000
25%        -0.007500
50%         0.000008
75%         0.007750
max         3.128750
Name: Slope, dtype: float64
```

[80]:
```
dossier_ts = '/Users/Ilyasse/Python projects/Equipement TS/37485.xlsx'
donn= pd.read_excel(dossier_ts)
detect_anomalies_pente(donn, '37482')
```

```
Statistiques des Pentes pour 37482:
count    33693.000000
mean        -0.007517
std          7.048958
min      -1034.893656
25%         -0.092640
50%         -0.000547
75%          0.088760
max        396.276720
Name: Slope, dtype: float64
```

[81]:
```
def calculate_slope(data, window_size):
    slopes = np.full(len(data), np.nan)

    for i in range(window_size, len(data)):
        window = data.iloc[i-window_size:i]
        if window['NGV'].isnull().any() or window['Date of Creation'].isnull().
↪any():
            continue

        x = (window['Date of Creation'] - window['Date of Creation'].iloc[0]).dt.
↪total_seconds() / 3600
        y = window['NGV']

        if len(x) > 1 and len(y) > 1:
            slope, intercept, r_value, p_value, std_err = linregress(x, y)
            slopes[i] = slope
    return slopes
```

```
def detect_constant_slope_periods(data, duration_hours, window_size,␣
↪tolerance=1e-5):
    # Assurez-vous que 'Date of Creation' est au format datetime
    data['Date of Creation'] = pd.to_datetime(data['Date of Creation'])

    # Calculez la pente locale
    data['Slope'] = calculate_slope(data, window_size)

    # Créez une colonne pour vérifier la constance de la pente sur la durée␣
↪définie
    data['Is Constant Slope'] = data['Slope'].rolling(window=int(duration_hours/
↪window_size), min_periods=1).apply(lambda x: np.all(np.isclose(x, x.mean(),␣
↪atol=tolerance)), raw=True)

    # Remplir les NaN avec False (ou 0)
    data['Is Constant Slope'] = data['Is Constant Slope'].fillna(False).
↪astype(int)

    # Créez une colonne pour la longueur des séquences de pente constante
    data['Constant Run Length'] = data['Is Constant Slope'].groupby(data['Is␣
↪Constant Slope'].ne(data['Is Constant Slope'].shift()).cumsum()).cumsum()

    # Filtrer les périodes où la pente est constante pour la durée définie ou␣
↪plus
    anomalies = data[data['Constant Run Length'] >= duration_hours/window_size]

    fig = px.scatter(data, x=data['Date of Creation'], y=data['NGV'], title='NGV␣
↪Over Time')

# Add red color for anomalies
    fig.add_scatter(x=anomalies['Date of Creation'], y=anomalies['NGV'],␣
↪mode='markers', marker=dict(color='red'), name='Anomalies')


    fig = go.Figure()

    # Add trace for NGV data
    fig.add_trace(go.Scatter(x=data['Date of Creation'],
                             y=data['NGV'],
                             mode='markers+lines',
                             name='NGV Équipement TA',
                             marker=dict(size=5),
                             line=dict(width=2)))
    # Ajouter des points rouges pour les anomalies
    fig.add_trace(go.Scatter(
        x=anomalies['Date of Creation'],
```

```
        y=anomalies['NGV'],
        mode='markers',
        marker=dict(color='red'),
        name='Anomalies'
    ))

    # Update layout of the plot
    fig.update_layout(title='NGV en fonction de la Date de Creation',
                      xaxis_title='Date de Creation',
                      yaxis_title='NGV',
                      template='plotly_white',
                      legend_title='Legend')

    # Show the plot
    fig.show()

    # Imprimez les statistiques des pentes
    print(data['Slope'].describe())



# Chemins des dossiers contenant les fichiers Excel pour les équipements TA et TS
dossier_ta = '/Users/Ilyasse/Python projects/Equipement TA/7710.csv'
dossier_ts = '/Users/Ilyasse/Python projects/Equipement TS'
donn= pd.read_csv(dossier_ta)

# Détecter les périodes de pente constante
constant_slope_periods = detect_constant_slope_periods(donn, duration_hours=10,␣
 ↪window_size=5)

# Visualiser les données avec les périodes de pente constante
```

```
count    7384.000000
mean        0.000425
std         0.044512
min        -0.766032
25%        -0.004795
50%        -0.000002
75%         0.004539
max         0.979268
Name: Slope, dtype: float64
```

[82]:
```
dossier_ta = '/Users/Ilyasse/Python projects/Equipement TA/7712.csv'
dossier_ts = '/Users/Ilyasse/Python projects/Equipement TS'
donn= pd.read_csv(dossier_ta)

# Détecter les périodes de pente constante
```

```
constant_slope_periods = detect_constant_slope_periods(donn, duration_hours=10,␣
↪window_size=5)
```

```
count    7440.000000
mean        0.001552
std         0.042332
min        -0.392000
25%        -0.004549
50%         0.000026
75%         0.004642
max         0.600054
Name: Slope, dtype: float64
```

[83]:
```
# Chargement des données
dossier_ta = '/Users/Ilyasse/Python projects/Equipement TA/7710.csv'
data = pd.read_csv(dossier_ta)
data['Date of Creation'] = pd.to_datetime(data['Date of Creation'])

# Check for duplicates
duplicate_dates = data[data.duplicated(subset='Date of Creation', keep=False)]
if not duplicate_dates.empty:
    print("Duplicated Dates Found:")
    print(duplicate_dates)

# Option 1: Remove duplicates


# Option 2: Aggregate duplicates (if applicable)
# data = data.groupby('Date of Creation').agg({'NGV': 'mean'}).reset_index()

# Générer des intervalles de temps réguliers
start_date = data['Date of Creation'].min()
end_date = data['Date of Creation'].max()
date_range = pd.date_range(start=start_date, end=end_date, freq='H')

# Réindexer les données pour avoir des intervalles de temps réguliers

# Interpoler les valeurs manquantes
data['NGV'] = data['NGV'].interpolate(method='linear')

# Fonction pour détecter les anomalies de pente
def detect_anomalies_interpolate(data, equipment_name, window_size_hours=5,␣
↪slope_threshold=0.01, duration_hours=10):

    # Calcul de la pente sur des fenêtres glissantes
    for i in range(len(data) - window_size_hours):
        window = data.iloc[i:i + window_size_hours]
```

```
        # Vérifier s'il y a des valeurs manquantes
        if window['NGV'].isnull().any():
            continue  # Ignore la fenêtre si elle contient des valeurs manquantes

        # Calculez les différences de temps en heures
        x = (window['Date of Creation'] - window['Date of Creation'].iloc[0]).dt.
↪total_seconds() / 3600
        y = window['NGV']

        # Calculez la pente de la régression linéaire
        slope, intercept, r_value, p_value, std_err = linregress(x, y)

        # Assignez la pente calculée à la ligne centrale de la fenêtre
        data.loc[i + window_size_hours // 2, 'Slope'] = slope

    # Supprimez les doublons dans 'Date of Creation'
    data = data.drop_duplicates(subset=['Date of Creation'])

    # Détecter les pentes constantes
    data['Is Constant Slope'] = data['Slope'].rolling(window=duration_hours,␣
↪min_periods=1).apply(
        lambda x: np.all(np.isclose(x, x.mean(), atol=1e-5)), raw=True)
    # Fill NaN values with 0 or another appropriate value before conversion
    data['Is Constant Slope'] = data['Is Constant Slope'].fillna(0).astype(int)

    # Créez une colonne pour la longueur des séquences de pente constante
    data['Constant Run Length'] = data['Is Constant Slope'].astype(int).groupby(
        data['Is Constant Slope'].ne(data['Is Constant Slope'].shift()).
↪cumsum()).cumsum()

    # Filtrer les périodes où la pente est constante pour la durée définie ou␣
↪plus
    constant_slope_periods = data[data['Constant Run Length'] >= duration_hours]

    anomalies = constant_slope_periods[['Date of Creation', 'NGV', 'Slope']]


    fig = px.scatter(data, x=data['Date of Creation'], y=data['NGV'], title='NGV␣
↪Over Time')

# Add red color for anomalies
    fig.add_scatter(x=anomalies['Date of Creation'], y=anomalies['NGV'],␣
↪mode='markers', marker=dict(color='red'), name='Anomalies')
```

```
    fig = go.Figure()

    # Add trace for NGV data
    fig.add_trace(go.Scatter(x=data['Date of Creation'],
                             y=data['NGV'],
                             mode='markers+lines',
                             name='NGV Équipement TA',
                             marker=dict(size=5),
                             line=dict(width=2)))
    # Ajouter des points rouges pour les anomalies
    fig.add_trace(go.Scatter(
        x=anomalies['Date of Creation'],
        y=anomalies['NGV'],
        mode='markers',
        marker=dict(color='red'),
        name='Anomalies'
    ))

    # Update layout of the plot
    fig.update_layout(title='NGV en fonction de la Date de Creation',
                      xaxis_title='Date de Creation',
                      yaxis_title='NGV',
                      template='plotly_white',
                      legend_title='Legend')

    # Show the plot
    fig.show()

    # Imprimez les statistiques des pentes
    print(data['Slope'].describe())
    # Afficher les statistiques des pentes
    print(f'Statistiques des Pentes pour {equipment_name}:')

    print(f'Nombre d\'anomalies détectées pour {equipment_name}:␣
↪{len(anomalies)}')



 # Appliquer la fonction pour détecter les anomalies de pente
anomalies = detect_anomalies_interpolate(data, '7712')
```

```
Duplicated Dates Found:
      Measure ID    NGV          Date of Creation Point Name
3034    13964955  0.107 2023-08-13 09:01:46+00:00      1RH-0
3036    13956450  0.105 2023-08-13 06:01:46+00:00      1RH-0
3041    13956489  0.105 2023-08-13 06:01:46+00:00      1RH-0
3044    13967072  0.107 2023-08-13 09:01:46+00:00      1RH-0
3401    14254240  0.418 2023-08-24 12:54:30+00:00      1RH-0
```

3408	14254435	0.418	2023-08-24 12:54:30+00:00	1RH-0
4571	15149718	0.629	2023-10-03 08:29:26+00:00	1RH-0
4572	15149720	1.352	2023-10-03 08:59:26+00:00	1RH-0
4574	15149717	0.629	2023-10-03 08:29:26+00:00	1RH-0
4575	15149721	1.352	2023-10-03 08:59:26+00:00	1RH-0
4576	15149724	1.372	2023-10-03 09:29:26+00:00	1RH-0
4577	15149727	1.367	2023-10-03 09:59:25+00:00	1RH-0
4578	15149729	1.367	2023-10-03 09:59:25+00:00	1RH-0
4582	15149730	1.366	2023-10-03 10:29:26+00:00	1RH-0
4585	15149725	1.372	2023-10-03 09:29:26+00:00	1RH-0
4586	15149731	1.366	2023-10-03 10:29:26+00:00	1RH-0
5412	15661503	0.385	2023-10-23 06:51:17+00:00	1RH-0
5413	15659451	0.385	2023-10-23 06:51:17+00:00	1RH-0
5842	18086089	1.983	2023-12-19 09:41:50+00:00	1RH-0
5843	18086066	1.983	2023-12-19 09:41:50+00:00	1RH-0
5845	18087189	2.119	2023-12-19 12:41:51+00:00	1RH-0
5848	18087169	2.119	2023-12-19 12:41:51+00:00	1RH-0
5849	18092952	2.340	2023-12-19 15:41:50+00:00	1RH-0
5850	18092902	2.340	2023-12-19 15:41:50+00:00	1RH-0
5851	18107532	2.231	2023-12-19 21:41:51+00:00	1RH-0
5853	18107477	2.231	2023-12-19 21:41:51+00:00	1RH-0
5855	18098321	2.326	2023-12-19 18:41:50+00:00	1RH-0
5857	18098291	2.326	2023-12-19 18:41:50+00:00	1RH-0
5966	18758247	0.221	2024-01-02 14:11:21+00:00	1RH-0
5970	18758774	0.221	2024-01-02 14:11:21+00:00	1RH-0
6094	19225568	3.014	2024-01-12 14:18:11+00:00	1RH-0
6098	19225041	3.014	2024-01-12 14:18:11+00:00	1RH-0
6099	19227330	3.014	2024-01-12 14:18:11+00:00	1RH-0
6102	19226442	3.014	2024-01-12 14:18:11+00:00	1RH-0
6235	19440641	1.389	2024-01-18 07:18:11+00:00	1RH-0
6238	19440225	1.389	2024-01-18 07:18:11+00:00	1RH-0
6308	19571202	1.214	2024-01-21 06:18:11+00:00	1RH-0
6309	19577155	1.214	2024-01-21 06:18:11+00:00	1RH-0
6327	19605268	1.073	2024-01-22 04:18:11+00:00	1RH-0
6330	19614182	1.073	2024-01-22 04:18:11+00:00	1RH-0
6381	19680680	1.095	2024-01-24 06:18:11+00:00	1RH-0
6384	19680692	1.095	2024-01-24 06:18:11+00:00	1RH-0
6423	19742001	1.125	2024-01-25 19:18:11+00:00	1RH-0
6426	19741981	1.125	2024-01-25 19:18:11+00:00	1RH-0
6468	19885366	0.233	2024-01-28 23:56:25+00:00	1RH-0
6485	19881363	0.233	2024-01-28 23:56:25+00:00	1RH-0
6487	19881514	0.186	2024-01-29 08:18:51+00:00	1RH-0
6489	19885383	0.186	2024-01-29 08:18:51+00:00	1RH-0
6671	20338325	1.349	2024-02-09 14:51:56+00:00	1RH-0
6672	20340863	1.349	2024-02-09 14:51:56+00:00	1RH-0
6981	21087450	0.264	2024-02-29 05:51:56+00:00	1RH-0
6985	21087007	0.264	2024-02-29 05:51:56+00:00	1RH-0
6988	21096824	0.290	2024-02-29 11:51:56+00:00	1RH-0

```
6989    21097390  0.290 2024-02-29 11:51:56+00:00      1RH-0
6992    21109728  0.316 2024-02-29 19:21:56+00:00      1RH-0
6996    21109343  0.316 2024-02-29 19:21:56+00:00      1RH-0
6998    21109995  0.316 2024-02-29 19:21:56+00:00      1RH-0
7251    21719309  0.374 2024-03-16 20:51:56+00:00      1RH-0
7254    21715545  0.374 2024-03-16 20:51:56+00:00      1RH-0
```

```
C:\Users\Ilyasse\AppData\Local\Temp\ipykernel_5380\482896588.py:53:
SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

C:\Users\Ilyasse\AppData\Local\Temp\ipykernel_5380\482896588.py:56:
SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

C:\Users\Ilyasse\AppData\Local\Temp\ipykernel_5380\482896588.py:59:
SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy


count    7353.000000
mean        0.000371
std         0.044512
min        -0.766032
25%        -0.004798
50%        -0.000003
75%         0.004484
max         0.979268
Name: Slope, dtype: float64
Statistiques des Pentes pour 7712:
Nombre d'anomalies détectées pour 7712: 0
```

8 Isolation forest

[84]:
```
def algo_forest(data):
# Ensure the 'Date of Creation' column is in datetime format
    data['Date of Creation'] = pd.to_datetime(data['Date of Creation'],␣
 ↪errors='coerce')

# Drop rows with invalid 'Date of Creation'
    data = data.dropna(subset=['Date of Creation'])

# Sort data by 'Date of Creation' to maintain temporal order
    data = data.sort_values(by='Date of Creation').reset_index(drop=True)

# Replace missing NGV values with the mean (for the purpose of the model)
    data['NGV'] = data['NGV'].fillna(data['NGV'].mean())

# Generate a feature set, 'Time' will be used as a feature
    data['Time'] = (data['Date of Creation'] - data['Date of Creation'].min()).
 ↪dt.total_seconds()

    X = data[['NGV', 'Time']]
# Initialize the Isolation Forest model
    iso_forest = IsolationForest(contamination=0.05, random_state=42)

# Fit the model to the data
    iso_forest.fit(X)

# Predict anomalies (-1 indicates an anomaly, 1 indicates normal)
    data['Anomaly'] = iso_forest.predict(X)

# Anomalies are rows where 'Anomaly' is -1
    anomalies = data[data['Anomaly'] == -1]
    fig = px.scatter(data, x=data['Date of Creation'], y=data['NGV'], title='NGV␣
 ↪Over Time')

# Add red color for anomalies
    fig.add_scatter(x=anomalies['Date of Creation'], y=anomalies['NGV'],␣
 ↪mode='markers', marker=dict(color='red'), name='Anomalies')


    fig = go.Figure()

    # Add trace for NGV data
    fig.add_trace(go.Scatter(x=data['Date of Creation'],
                             y=data['NGV'],
                             mode='markers+lines',
                             name='NGV Équipement TA',
```

```
                            marker=dict(size=5),
                            line=dict(width=2)))
    # Ajouter des points rouges pour les anomalies
    fig.add_trace(go.Scatter(
        x=anomalies['Date of Creation'],
        y=anomalies['NGV'],
        mode='markers',
        marker=dict(color='red'),
        name='Anomalies'
    ))

    # Update layout of the plot
    fig.update_layout(title='NGV en fonction de la Date de Creation',
                      xaxis_title='Date de Creation',
                      yaxis_title='NGV',
                      template='plotly_white',
                      legend_title='Legend')

    # Show the plot
    fig.show()
    fig = px.scatter(data, x=data['Date of Creation'], y=data['NGV'], title='NGV␣
↪Over Time')

# Add red color for anomalies
    fig.add_scatter(x=anomalies['Date of Creation'], y=anomalies['NGV'],␣
↪mode='markers', marker=dict(color='red'), name='Anomalies')


    fig = go.Figure()

    # Add trace for NGV data
    fig.add_trace(go.Scatter(x=data['Date of Creation'],
                            y=data['NGV'],
                            mode='markers+lines',
                            name='NGV Équipement TA',
                            marker=dict(size=5),
                            line=dict(width=2)))
    # Ajouter des points rouges pour les anomalies
    fig.add_trace(go.Scatter(
        x=anomalies['Date of Creation'],
        y=anomalies['NGV'],
        mode='markers',
        marker=dict(color='red'),
        name='Anomalies'
    ))

    # Update layout of the plot
```

```
    fig.update_layout(title='NGV en fonction de la Date de Creation',
                      xaxis_title='Date de Creation',
                      yaxis_title='NGV',
                      template='plotly_white',
                      legend_title='Legend')

    # Show the plot
    fig.show()
```

[85]:
```
dossier_ta = '/Users/Ilyasse/Python projects/Equipement TA/7710.csv'
dossier_ts = '/Users/Ilyasse/Python projects/Equipement TS'
donn= pd.read_csv(dossier_ta)
ily = algo_forest(donn)
```

9 MLP regression

[87]:
```
def detect_anomalies_mlp(data, equipment_name):
    # Ensure 'Date of Creation' is in datetime format
    data['Date of Creation'] = pd.to_datetime(data['Date of Creation'],␣
↪errors='coerce')

    # Drop rows with invalid 'Date of Creation'
    data = data.dropna(subset=['Date of Creation'])

    # Sort data by date
    data = data.sort_values('Date of Creation').reset_index(drop=True)

    # Create shifted versions of the sequence for X(t) and X(t-1)
    data['X(t)'] = data['NGV']
    data['X(t-1)'] = data['NGV'].shift(1)

    # Drop rows with NaN values
    data = data.dropna()

    # Define features (X) and target (y)
    X = data[['X(t-1)', 'X(t)']]
    y = data['X(t)']  # Current value as the target

    # Split the data into training and test sets
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪shuffle=False)

    # Initialize the MLP Regressor (Neural Network)
    model = MLPRegressor(hidden_layer_sizes=(50,), activation='relu',␣
↪solver='adam', max_iter=1000, random_state=42)
```

```
    # Train the model
    model.fit(X_train, y_train)

    # Make predictions on the test set
    y_pred = model.predict(X_test)

    # Calculate residuals (difference between actual and predicted values)
    residuals = np.abs(y_test - y_pred)

    # Define a threshold for detecting anomalies (e.g., 2 standard deviations)
    threshold = 2 * np.std(residuals)

    # Flag anomalies
    anomalies = residuals > threshold

    # Create a column 'Anomaly' in the dataset for marking anomalies
    data['Anomaly'] = 0
    data.iloc[-len(y_test):, -1] = anomalies * -1  # Mark anomalies with -1

    # Extract anomalies
    anomalies = data[data['Anomaly'] == -1]

    # Plot the results using Plotly
    fig = go.Figure()

    # Add trace for NGV data
    fig.add_trace(go.Scatter(x=data['Date of Creation'],
                             y=data['NGV'],
                             mode='markers+lines',
                             name=f'NGV {equipment_name}',
                             marker=dict(size=5),
                             line=dict(width=2)))

    # Add red markers for anomalies
    fig.add_trace(go.Scatter(
        x=anomalies['Date of Creation'],
        y=anomalies['NGV'],
        mode='markers',
        marker=dict(color='red', size=10),
        name='Anomalies'
    ))

    # Update layout of the plot
    fig.update_layout(title=f'NGV en fonction de la Date de Creation pour␣
↪{equipment_name}',
                      xaxis_title='Date de Creation',
                      yaxis_title='NGV',
```

```
                      template='plotly_white',
                      legend_title='Legend')

    # Show the plot
    fig.show()

    # Print summary of anomalies
    print(f'Nombre d\'anomalies détectées : {len(anomalies)}')
    print(f'Seuil statistique utilisé : {threshold}')
# Example Usage:
# detect_anomalies_mlp(data, 'Equipment TA')
```

[88]:
```
detect_anomalies_mlp(donn, 'Equipment TA')
```

```
Nombre d'anomalies détectées : 11
Seuil statistique utilisé : 0.06092786955557499
```

10 Algorithme proposed

[89]:
```
def detect_missing_data_with_periods(data):
    data['Date of Creation'] = pd.to_datetime(data['Date of Creation'],␣
 ↪errors='coerce')

# Drop rows with invalid 'Date of Creation'
    data = data.dropna(subset=['Date of Creation'])

# Sort data by 'Date of Creation' to maintain temporal order
    data = data.sort_values(by='Date of Creation').reset_index(drop=True)

# Replace missing NGV values with the mean (for the purpose of the model)

# Generate a feature set, 'Time' will be used as a feature
    data['Period'] = data['Date of Creation'].diff().dt.total_seconds()
    prev_period = data['Period'].iloc[0]
    current_state = 0
    is_missing_data = False
    start_time = None
    last_0_period = prev_period

    missing_data_intervals = []
    anomalies = []
    for i in range(1, len(data)):
        current_period = data['Period'].iloc[i]

        if current_period == prev_period:
            current_state = 0
            is_missing_data = False
```

```
            last_0_period = current_period

        elif current_period > prev_period:

            if current_state == 0:
                start_time = data['Date of Creation'].iloc[i]
            current_state = 1
            is_missing_data = True

        elif current_period < prev_period:

            if current_state == 0:
                start_time = data['Date of Creation'].iloc[i]
            current_state = 2

        next_period = data['Period'].iloc[i+1] if i + 1 < len(data) else None

        if next_period == prev_period:

                current_state = 0
        elif next_period and next_period > current_period:

            if current_state == 1:
                if next_period == last_0_period:
                    current_state = 0
                elif next_period > last_0_period:

                    end_time = data['Date of Creation'].iloc[i] - pd.
↪Timedelta(seconds=1)
                    missing_data_intervals.append((start_time, end_time))

                    anomalies.append({'start_time': start_time, 'end_time':␣
↪end_time})
                    current_state = 0

                else:
                    pass  # Continue to next measurement
            else:
                pass  # Continue to next measurement
        else:
            pass  # Continue to next measurement

        prev_period = current_period
        return missing_data_intervals
dossier_ta = '/Users/Ilyasse/Python projects/Equipement TA/7721.csv'
donn= pd.read_csv(dossier_ta)
```

```
missing_intervals = detect_missing_data_with_periods(donn)
print("Intervals with missing data:", missing_intervals)
```

```
Intervals with missing data: []
```

[90]:
```
def detect_missing_data(data):
    data = data.sort_values(by=['Point Name', 'Date of Creation']).
 ↪reset_index(drop=True)
    data['Date of Creation'] = pd.to_datetime(data['Date of Creation'])

    previous_time = None
    previous_frequency = 0
    last_zero_time = None
    last_zero_frequency = 0
    missing_data_intervals = []
    twelve_hours = 12 * 3600  # 12 heures en secondes
    state = None
    one_count = 0
    first_entry = True

    for i in range(len(data)):
        current_time = data.loc[i, 'Date of Creation']
        current_point = data.loc[i, 'Point Name']

        if previous_time is not None:
            current_frequency = (current_time - previous_time).total_seconds()
        else:
            current_frequency = 0

        if previous_time is not None:
            if current_frequency == previous_frequency:
                state = "0"
                last_zero_time = current_time
                last_zero_frequency = current_frequency
                one_count = 0
            elif current_frequency > previous_frequency:
                if first_entry:
                    state = "0"
                    last_zero_time = current_time
                    last_zero_frequency = current_frequency
                    first_entry = False
                else:
                    one_count += 1
                    if state == "2" and current_frequency <= last_zero_frequency:
                        one_count = 0
                    else:
                        if one_count == 1:
```

```
                            missing_start = previous_time + pd.
↪Timedelta(seconds=last_zero_frequency)
                        missing_end = current_time - pd.Timedelta(seconds=1)
                        state = "1"
            elif current_frequency < previous_frequency:
                state = "2"
                one_count = 0
                if previous_frequency > 0:
                    missing_start = previous_time + pd.
↪Timedelta(seconds=last_zero_frequency)
                    missing_end = current_time - pd.Timedelta(seconds=1)

            if current_frequency > twelve_hours:
                missing_start = previous_time
                missing_end = current_time - pd.Timedelta(seconds=1)
                missing_data_intervals.append((missing_start, missing_end))

        previous_time = current_time
        previous_frequency = current_frequency

        if (current_frequency == previous_frequency or current_frequency <␣
↪previous_frequency) and one_count > 0:
            if state != "2":
                missing_data_intervals.append((missing_start, missing_end))
            one_count = 0

        first_entry = False

    if state == "1":
        missing_start = previous_time + pd.Timedelta(seconds=last_zero_frequency)
        missing_end = current_time - pd.Timedelta(seconds=1)
        missing_data_intervals.append((missing_start, missing_end))

    # Filtrage des intervalles manquants pour éviter les doublons et les␣
↪chevauchements
    if missing_data_intervals:
        filtered_intervals = []
        current_start, current_end = missing_data_intervals[0]

        for start, end in missing_data_intervals[1:]:
            if start <= current_end + pd.Timedelta(seconds=1):
                current_end = max(current_end, end)
            else:
                filtered_intervals.append((current_start, current_end))
                current_start, current_end = start, end

        filtered_intervals.append((current_start, current_end))
```

```
        # Ne garder que les intervalles significatifs et éviter les doublons
        unique_intervals = []
        for start, end in filtered_intervals:
            if start != end and (start, end) not in unique_intervals:
                unique_intervals.append((start, end))

        # Ignorer la première période détectée comme données manquantes
        if len(unique_intervals) > 1:
            unique_intervals = unique_intervals[1:]



    return unique_intervals
```

[91]:
```
# Chargement des données depuis le dossier spécifié
folder_path = '/Users/Ilyasse/Python projects/Equipement TA/7721.csv'
data = pd.read_csv(folder_path)

# Convertir 'Date of Creation' en datetime et s'assurer qu'il est en UTC
data['Date of Creation'] = pd.to_datetime(data['Date of Creation'])

# Si les dates sont naïves, les localiser en UTC
if data['Date of Creation'].dt.tz is None:
    data['Date of Creation'] = data['Date of Creation'].dt.tz_localize('UTC')
else:
    data['Date of Creation'] = data['Date of Creation'].dt.tz_convert('UTC')

# Définir la période d'analyse
start_date = pd.to_datetime('2022-11-04').tz_localize('UTC')
end_date = pd.to_datetime('2022-12-11').tz_localize('UTC')

# Filtrer les données pour la période définie
data_period = data[(data['Date of Creation'] >= start_date) & (data['Date of␣
 ↪Creation'] <= end_date)]

# Analyse des données manquantes pour chaque 'Point Name'
unique_points = data_period['Point Name'].unique()

for point in unique_points:
    point_data = data_period[data_period['Point Name'] == point].copy()
    print(f"\nAnalyse des données manquantes pour le point : {point}")

    # Appeler la fonction de détection des données manquantes
    missing_intervals = detect_missing_data(point_data)

    if not missing_intervals:
```

```
        print("Aucune donnée manquante détectée.")
    else:
        for interval in missing_intervals:
            print(f"Il y a des données manquantes entre {interval[0]} et␣
↪{interval[1]}")
```

```
Analyse des données manquantes pour le point : 5RV-0
Il y a des données manquantes entre 2022-11-08 18:22:31+00:00 et 2022-11-13
00:15:43+00:00
Il y a des données manquantes entre 2022-11-14 16:15:15+00:00 et 2022-11-14
16:15:16+00:00
Il y a des données manquantes entre 2022-11-16 16:15:17+00:00 et 2022-11-16
20:15:16+00:00
Il y a des données manquantes entre 2022-11-17 00:15:17+00:00 et 2022-11-17
19:41:08+00:00
Il y a des données manquantes entre 2022-11-20 07:34:05+00:00 et 2022-11-20
07:34:06+00:00
Il y a des données manquantes entre 2022-11-22 11:34:07+00:00 et 2022-11-22
15:34:07+00:00
Il y a des données manquantes entre 2022-11-26 19:34:07+00:00 et 2022-11-27
02:00:40+00:00
Il y a des données manquantes entre 2022-12-01 02:00:41+00:00 et 2022-12-01
02:00:46+00:00
Il y a des données manquantes entre 2022-12-03 14:00:47+00:00 et 2022-12-04
14:00:46+00:00
Il y a des données manquantes entre 2022-12-05 18:57:59+00:00 et 2022-12-05
18:58:00+00:00
Il y a des données manquantes entre 2022-12-06 02:58:01+00:00 et 2022-12-06
10:58:00+00:00
```

[92]:
```
# Chargement des données depuis le dossier spécifié
folder_path = '/Users/Ilyasse/Python projects/Equipement TA/7721.csv'

data = pd.read_csv(folder_path)
# Analyse des données manquantes pour chaque 'Point Name'
unique_points = data['Point Name'].unique()

for point in unique_points:
    point_data = data[data['Point Name'] == point].copy()
    print(f"\nAnalyse des données manquantes pour le point : {point}")
    missing_intervals = detect_missing_data(point_data)

    if not missing_intervals:
        print("Aucune donnée manquante détectée.")
    else:
        for interval in missing_intervals:
```

```
            print(f"Il y a des données manquantes entre {interval[0]} et␣
↪{interval[1]}")
```

```
Analyse des données manquantes pour le point : 5RV-0
Il y a des données manquantes entre 2022-01-16 13:25:21+00:00 et 2022-01-16
21:25:20+00:00
Il y a des données manquantes entre 2022-01-17 05:25:21+00:00 et 2022-01-17
06:07:26+00:00
Il y a des données manquantes entre 2022-01-18 07:49:33+00:00 et 2022-01-18
08:11:43+00:00
Il y a des données manquantes entre 2022-01-18 12:11:44+00:00 et 2022-01-19
17:08:54+00:00
Il y a des données manquantes entre 2022-01-20 13:08:55+00:00 et 2022-01-20
13:12:07+00:00
Il y a des données manquantes entre 2022-01-23 18:11:43+00:00 et 2022-01-24
22:20:30+00:00
Il y a des données manquantes entre 2022-02-20 16:16:57+00:00 et 2022-02-20
20:16:56+00:00
Il y a des données manquantes entre 2022-03-05 08:16:57+00:00 et 2022-03-05
16:16:56+00:00
Il y a des données manquantes entre 2022-03-05 23:01:34+00:00 et 2022-03-06
15:01:33+00:00
Il y a des données manquantes entre 2022-03-13 07:29:20+00:00 et 2022-03-13
12:29:24+00:00
Il y a des données manquantes entre 2022-03-21 05:24:52+00:00 et 2022-03-23
13:24:51+00:00
Il y a des données manquantes entre 2022-03-23 18:01:58+00:00 et 2022-03-24
14:01:57+00:00
Il y a des données manquantes entre 2022-03-25 06:01:58+00:00 et 2022-03-29
19:37:20+00:00
Il y a des données manquantes entre 2022-04-01 23:17:59+00:00 et 2022-04-02
01:12:45+00:00
Il y a des données manquantes entre 2022-04-03 09:12:46+00:00 et 2022-04-03
15:42:19+00:00
Il y a des données manquantes entre 2022-04-05 01:25:42+00:00 et 2022-04-05
01:26:04+00:00
Il y a des données manquantes entre 2022-04-06 13:26:06+00:00 et 2022-04-06
13:28:42+00:00
Il y a des données manquantes entre 2022-04-20 17:32:42+00:00 et 2022-04-20
17:34:17+00:00
Il y a des données manquantes entre 2022-04-23 23:47:58+00:00 et 2022-04-24
07:08:12+00:00
Il y a des données manquantes entre 2022-04-25 14:32:11+00:00 et 2022-04-27
03:00:42+00:00
Il y a des données manquantes entre 2022-04-28 14:23:39+00:00 et 2022-05-04
12:27:41+00:00
Il y a des données manquantes entre 2022-05-10 10:42:37+00:00 et 2022-05-11
```

```
14:42:36+00:00
Il y a des données manquantes entre 2022-05-19 06:56:01+00:00 et 2022-05-19
08:18:52+00:00
Il y a des données manquantes entre 2022-06-21 03:13:36+00:00 et 2022-06-21
04:49:38+00:00
Il y a des données manquantes entre 2022-06-22 16:18:30+00:00 et 2022-06-22
16:23:36+00:00
Il y a des données manquantes entre 2022-07-04 10:44:52+00:00 et 2022-07-20
11:53:04+00:00
Il y a des données manquantes entre 2022-08-04 05:48:27+00:00 et 2022-08-04
09:48:26+00:00
Il y a des données manquantes entre 2022-08-06 13:18:09+00:00 et 2022-08-06
13:25:23+00:00
Il y a des données manquantes entre 2022-08-12 00:26:31+00:00 et 2022-08-12
00:27:07+00:00
Il y a des données manquantes entre 2022-08-19 11:10:56+00:00 et 2022-08-19
11:10:57+00:00
Il y a des données manquantes entre 2022-09-08 22:01:29+00:00 et 2022-09-15
16:21:44+00:00
Il y a des données manquantes entre 2022-09-19 08:00:32+00:00 et 2022-09-20
18:00:28+00:00
Il y a des données manquantes entre 2022-09-21 16:00:36+00:00 et 2022-09-21
16:00:37+00:00
Il y a des données manquantes entre 2022-10-04 08:00:31+00:00 et 2022-10-04
12:00:30+00:00
Il y a des données manquantes entre 2022-10-04 20:00:31+00:00 et 2022-10-05
00:00:30+00:00
Il y a des données manquantes entre 2022-10-05 14:43:27+00:00 et 2022-10-05
14:48:28+00:00
Il y a des données manquantes entre 2022-10-05 22:48:29+00:00 et 2022-10-06
02:48:28+00:00
Il y a des données manquantes entre 2022-10-08 17:43:50+00:00 et 2022-10-08
17:43:51+00:00
Il y a des données manquantes entre 2022-10-14 00:34:09+00:00 et 2022-10-14
00:34:10+00:00
Il y a des données manquantes entre 2022-10-16 21:50:43+00:00 et 2022-10-16
21:50:44+00:00
Il y a des données manquantes entre 2022-10-17 12:34:30+00:00 et 2022-10-17
12:34:32+00:00
Il y a des données manquantes entre 2022-10-24 18:55:45+00:00 et 2022-10-24
18:55:47+00:00
Il y a des données manquantes entre 2022-10-25 18:05:07+00:00 et 2022-10-25
18:05:08+00:00
Il y a des données manquantes entre 2022-10-27 13:25:55+00:00 et 2022-10-27
13:25:56+00:00
Il y a des données manquantes entre 2022-11-02 14:22:29+00:00 et 2022-11-02
14:22:30+00:00
Il y a des données manquantes entre 2022-11-08 18:22:31+00:00 et 2022-11-13
```

00:15:43+00:00
Il y a des données manquantes entre 2022-11-14 16:15:15+00:00 et 2022-11-14 16:15:16+00:00
Il y a des données manquantes entre 2022-11-16 16:15:17+00:00 et 2022-11-16 20:15:16+00:00
Il y a des données manquantes entre 2022-11-17 00:15:17+00:00 et 2022-11-17 19:41:08+00:00
Il y a des données manquantes entre 2022-11-20 07:34:05+00:00 et 2022-11-20 07:34:06+00:00
Il y a des données manquantes entre 2022-11-22 11:34:07+00:00 et 2022-11-22 15:34:07+00:00
Il y a des données manquantes entre 2022-11-26 19:34:07+00:00 et 2022-11-27 02:00:40+00:00
Il y a des données manquantes entre 2022-12-01 02:00:41+00:00 et 2022-12-01 02:00:46+00:00
Il y a des données manquantes entre 2022-12-03 14:00:47+00:00 et 2022-12-04 14:00:46+00:00
Il y a des données manquantes entre 2022-12-05 18:57:59+00:00 et 2022-12-05 18:58:00+00:00
Il y a des données manquantes entre 2022-12-06 02:58:01+00:00 et 2022-12-06 10:58:00+00:00
Il y a des données manquantes entre 2022-12-18 16:20:33+00:00 et 2022-12-18 16:20:34+00:00
Il y a des données manquantes entre 2022-12-24 04:20:35+00:00 et 2022-12-24 08:20:34+00:00
Il y a des données manquantes entre 2022-12-27 16:20:35+00:00 et 2023-04-03 10:14:51+00:00
Il y a des données manquantes entre 2023-04-10 10:14:52+00:00 et 2023-08-01 11:01:51+00:00
Il y a des données manquantes entre 2023-08-11 19:01:44+00:00 et 2023-08-11 19:31:45+00:00
Il y a des données manquantes entre 2023-08-19 19:31:47+00:00 et 2023-08-19 20:18:43+00:00
Il y a des données manquantes entre 2023-08-19 20:48:44+00:00 et 2023-08-19 22:18:43+00:00
Il y a des données manquantes entre 2023-08-22 10:18:44+00:00 et 2023-08-22 11:12:53+00:00
Il y a des données manquantes entre 2023-08-22 18:54:29+00:00 et 2023-08-22 18:54:30+00:00
Il y a des données manquantes entre 2023-08-30 09:25:54+00:00 et 2023-08-30 09:50:50+00:00
Il y a des données manquantes entre 2023-08-30 10:20:51+00:00 et 2023-08-30 10:50:50+00:00
Il y a des données manquantes entre 2023-08-30 15:12:34+00:00 et 2023-08-30 15:12:36+00:00
Il y a des données manquantes entre 2023-08-30 19:15:03+00:00 et 2023-08-30 19:15:04+00:00
Il y a des données manquantes entre 2023-10-03 10:59:26+00:00 et 2023-10-03

15:28:49+00:00
Il y a des données manquantes entre 2023-10-06 12:58:53+00:00 et 2023-10-06
13:28:52+00:00
Il y a des données manquantes entre 2023-10-12 06:44:42+00:00 et 2023-10-12
06:44:43+00:00
Il y a des données manquantes entre 2023-10-18 11:48:36+00:00 et 2023-10-18
14:40:01+00:00
Il y a des données manquantes entre 2023-10-18 20:10:02+00:00 et 2023-10-18
21:10:01+00:00
Il y a des données manquantes entre 2023-10-18 21:40:02+00:00 et 2023-10-19
00:40:02+00:00
Il y a des données manquantes entre 2023-10-19 09:21:13+00:00 et 2023-10-19
09:51:17+00:00
Il y a des données manquantes entre 2023-10-19 12:21:17+00:00 et 2023-10-19
12:51:17+00:00
Il y a des données manquantes entre 2023-10-19 14:21:17+00:00 et 2023-10-19
14:51:16+00:00
Il y a des données manquantes entre 2023-10-29 17:13:34+00:00 et 2023-10-30
10:13:34+00:00
Il y a des données manquantes entre 2023-10-30 12:13:35+00:00 et 2023-11-03
18:49:04+00:00
Il y a des données manquantes entre 2023-11-10 09:18:42+00:00 et 2023-11-10
16:41:29+00:00
Il y a des données manquantes entre 2023-11-10 17:58:00+00:00 et 2023-11-10
18:58:03+00:00
Il y a des données manquantes entre 2023-11-16 10:49:43+00:00 et 2023-11-16
10:50:09+00:00
Il y a des données manquantes entre 2023-11-21 22:50:10+00:00 et 2023-12-08
09:17:33+00:00
Il y a des données manquantes entre 2023-12-08 17:41:50+00:00 et 2023-12-08
18:41:50+00:00
Il y a des données manquantes entre 2023-12-08 23:41:50+00:00 et 2023-12-09
00:41:50+00:00
Il y a des données manquantes entre 2023-12-09 08:41:50+00:00 et 2023-12-09
09:41:49+00:00
Il y a des données manquantes entre 2023-12-27 18:08:50+00:00 et 2023-12-27
18:08:51+00:00
Il y a des données manquantes entre 2023-12-29 14:11:19+00:00 et 2023-12-29
14:11:20+00:00
Il y a des données manquantes entre 2023-12-31 02:11:21+00:00 et 2023-12-31
05:11:20+00:00
Il y a des données manquantes entre 2024-01-08 17:55:46+00:00 et 2024-01-08
17:55:47+00:00
Il y a des données manquantes entre 2024-01-12 14:18:11+00:00 et 2024-01-12
15:18:10+00:00
Il y a des données manquantes entre 2024-01-26 21:56:04+00:00 et 2024-01-26
22:26:24+00:00
Il y a des données manquantes entre 2024-01-29 09:48:51+00:00 et 2024-01-29

```
10:21:55+00:00
Il y a des données manquantes entre 2024-03-23 11:51:56+00:00 et 2024-04-03
10:58:57+00:00
```

[93]:
```python
 # Convertir les dates
data['Date of Creation'] = pd.to_datetime(data['Date of Creation'])
data = data.sort_values(by='Date of Creation').reset_index(drop=True)

data['Period'] = data['Date of Creation'].diff().dt.total_seconds() / 3600

# Analyse des données manquantes pour chaque 'Point Name'
unique_points = data['Point Name'].unique()

for point in unique_points:
    point_data = data[data['Point Name'] == point].copy()


    missing_intervals = detect_missing_data(point_data)


        # Tracer les NGV et les anomalies pour ce point
    fig = go.Figure()

        # Trace principale pour NGV
    fig.add_trace(go.Scatter(
            x=point_data['Date of Creation'],
            y=point_data['NGV'],
            mode='markers+lines',
            name=f'NGV {point}',
            marker=dict(size=5),
            line=dict(width=2)
        ))

        # Ajouter les anomalies
    for interval in missing_intervals:
            fig.add_trace(go.Scatter(
                x=[interval[0], interval[1]],
                y=[point_data['NGV'].mean()] * 2,  # On place les points␣
↪d'anomalie à la moyenne des NGV
                mode='markers+lines',
                marker=dict(color='red', size=10),
                line=dict(color='red', width=2),
                name='Anomalies'
            ))

        # Mettre à jour la disposition
    fig.update_layout(
```

```
            title=f'NGV en fonction de la Date de Creation pour {point}',
            xaxis_title='Date de Creation',
            yaxis_title='NGV',
            template='plotly_white',
            showlegend=True
        )

        # Afficher le graphique
    fig.show()
```

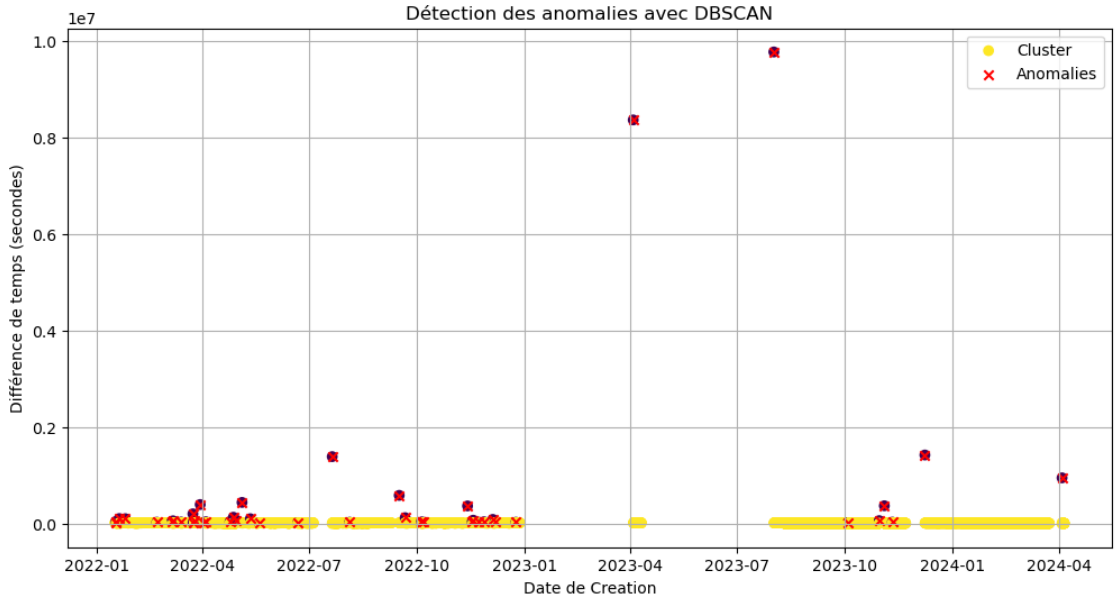
11 Problem solution

12 Clustering with DBSCAN

[96]:
```
# Charger les données
folder_path = '/Users/Ilyasse/Python projects/Equipement TA/7721.csv'
data = pd.read_csv(folder_path)

# Convertir 'Date of Creation' en datetime
data['Date of Creation'] = pd.to_datetime(data['Date of Creation'])
data.sort_values(by='Date of Creation', inplace=True)
# Calculer les différences de temps en secondes
data['time_diff'] = data['Date of Creation'].diff().dt.total_seconds()

# Supprimer les lignes avec des valeurs manquantes
data = data.dropna(subset=['time_diff'])

# Préparer les données pour le clustering
X = data[['time_diff']].values

# Appliquer DBSCAN
dbscan = DBSCAN(eps=1000, min_samples=10)  # Ajuster les hyperparamètres eps et␣
↪min_samples
data['cluster'] = dbscan.fit_predict(X)

# Identifier les anomalies (points marqués comme bruit avec DBSCAN)
data['is_anomaly'] = data['cluster'] == -1

# Tracer les résultats
plt.figure(figsize=(12, 6))

# Tracer les clusters
plt.scatter(data['Date of Creation'], data['time_diff'], c=data['cluster'],␣
↪cmap='viridis', marker='o', label='Cluster')

# Tracer les anomalies
```

```
anomalies = data[data['is_anomaly']]
plt.scatter(anomalies['Date of Creation'], anomalies['time_diff'], color='red',␣
 ↪marker='x', label='Anomalies')

plt.title('Détection des anomalies avec DBSCAN')
plt.xlabel('Date de Creation')
plt.ylabel('Différence de temps (secondes)')
plt.legend()
plt.grid(True)
plt.show()
# Visualisez l'évolution de NGV au fil du temps avec les anomalies et les␣
 ↪interpolations
    # Create a scatter plot for all data points
fig = px.scatter(data, x=data['Date of Creation'], y=data['NGV'], title='NGV␣
 ↪Over Time')

# Add red color for anomalies
fig.add_scatter(x=anomalies['Date of Creation'], y=anomalies['NGV'],␣
 ↪mode='markers', marker=dict(color='red'), name='Anomalies')


fig = go.Figure()

    # Add trace for NGV data
fig.add_trace(go.Scatter(x=data['Date of Creation'],
                         y=data['NGV'],
                         mode='markers+lines',
                         name='NGV Équipement TA',
                         marker=dict(size=5),
                         line=dict(width=2)))
    # Ajouter des points rouges pour les anomalies
fig.add_trace(go.Scatter(
        x=anomalies['Date of Creation'],
        y=anomalies['NGV'],
        mode='markers',
        marker=dict(color='red'),
        name='Anomalies'
    ))

# Update layout of the plot
fig.update_layout(title='NGV en fonction de la Date de Creation',
                  xaxis_title='Date de Creation',
                  yaxis_title='NGV',
                  template='plotly_white',
                  legend_title='Legend')

    # Show the plot
```

```
fig.show()
```



[97]:
```
# Charger les données
folder_path = '/Users/Ilyasse/Python projects/Equipement TA/7721.csv'
data = pd.read_csv(folder_path)

# Convertir 'Date of Creation' en datetime et s'assurer qu'il est en UTC
data['Date of Creation'] = pd.to_datetime(data['Date of Creation']).dt.
 ↪tz_convert('UTC')
data.sort_values(by='Date of Creation', inplace=True)

# Calculer les différences de temps en secondes
data['time_diff'] = data['Date of Creation'].diff().dt.total_seconds()

# Supprimer les lignes avec des valeurs manquantes
data = data.dropna(subset=['time_diff'])

# Préparer les données pour le clustering
X = data[['time_diff']].values

# Appliquer DBSCAN
dbscan = DBSCAN(eps=1000, min_samples=5)  # Ajuster les hyperparamètres eps et␣
 ↪min_samples
data['cluster'] = dbscan.fit_predict(X)

# Identifier les anomalies (points marqués comme bruit avec DBSCAN)
```

```
data['is_anomaly'] = data['cluster'] == -1

# Définir la période d'analyse
start_date = pd.to_datetime('2022-11-04').tz_localize('UTC')
end_date = pd.to_datetime('2022-12-11').tz_localize('UTC')

# Filtrer les données pour la période définie
data_period = data[(data['Date of Creation'] >= start_date) & (data['Date of␣
↪Creation'] <= end_date)]

# Anomalies détectées dans la période
anomalies_period = data_period[data_period['is_anomaly']]

# Afficher le nombre total de points et le nombre d'anomalies
print(f"Nombre total de points dans la période : {len(data_period)}")
print(f"Nombre d'anomalies détectées dans la période : {len(anomalies_period)}")

# Parcourir les anomalies détectées
for anomaly_index in anomalies_period.index:
    anomaly = data_period.loc[anomaly_index]

    print("\n--- Anomalie détectée ---")
    print(f"Date de l'anomalie: {anomaly['Date of Creation']}")
    print(f"Time diff de l'anomalie: {anomaly['time_diff']} secondes")

    # Obtenir la position de l'anomalie dans data_period
    position = data_period.index.get_loc(anomaly_index)

    # Temps juste avant l'anomalie
    print("\nTemps avant l'anomalie :")
    if position - 2 >= 0:
        prev_anomaly_2 = data_period.iloc[position - 2]
        print(f"Date avant l'anomalie 2: {prev_anomaly_2['Date of Creation']}␣
↪avec time diff {prev_anomaly_2['time_diff']} secondes")
    if position - 1 >= 0:
        prev_anomaly_1 = data_period.iloc[position - 1]
        print(f"Date avant l'anomalie 1: {prev_anomaly_1['Date of Creation']}␣
↪avec time diff {prev_anomaly_1['time_diff']} secondes")

    # Temps juste après l'anomalie
    print("\nTemps après l'anomalie :")
    if position + 1 < len(data_period):
        next_anomaly_1 = data_period.iloc[position + 1]
        print(f"Date après l'anomalie 1: {next_anomaly_1['Date of Creation']}␣
↪avec time diff {next_anomaly_1['time_diff']} secondes")
    if position + 2 < len(data_period):
        next_anomaly_2 = data_period.iloc[position + 2]
```

```
        print(f"Date après l'anomalie 2: {next_anomaly_2['Date of Creation']}␣
↪avec time diff {next_anomaly_2['time_diff']} secondes")
```

```
Nombre total de points dans la période : 186
Nombre d'anomalies détectées dans la période : 5

--- Anomalie détectée ---
Date de l'anomalie: 2022-11-13 00:15:44+00:00
Time diff de l'anomalie: 365954.0 secondes

Temps avant l'anomalie :
Date avant l'anomalie 2: 2022-11-08 14:22:31+00:00 avec time diff 207.0 secondes
Date avant l'anomalie 1: 2022-11-08 18:36:30+00:00 avec time diff 15239.0
secondes

Temps après l'anomalie :
Date après l'anomalie 1: 2022-11-13 04:15:44+00:00 avec time diff 14400.0
secondes
Date après l'anomalie 2: 2022-11-13 08:15:44+00:00 avec time diff 14400.0
secondes

--- Anomalie détectée ---
Date de l'anomalie: 2022-11-17 19:41:09+00:00
Time diff de l'anomalie: 69952.0 secondes

Temps avant l'anomalie :
Date avant l'anomalie 2: 2022-11-16 20:15:17+00:00 avec time diff 28800.0
secondes
Date avant l'anomalie 1: 2022-11-17 00:15:17+00:00 avec time diff 14400.0
secondes

Temps après l'anomalie :
Date après l'anomalie 1: 2022-11-17 23:41:09+00:00 avec time diff 14400.0
secondes
Date après l'anomalie 2: 2022-11-18 03:41:09+00:00 avec time diff 14400.0
secondes

--- Anomalie détectée ---
Date de l'anomalie: 2022-11-27 02:00:41+00:00
Time diff de l'anomalie: 37594.0 secondes

Temps avant l'anomalie :
Date avant l'anomalie 2: 2022-11-26 11:34:07+00:00 avec time diff 14400.0
secondes
Date avant l'anomalie 1: 2022-11-26 15:34:07+00:00 avec time diff 14400.0
secondes

Temps après l'anomalie :
```

```
Date après l'anomalie 1: 2022-11-27 06:00:41+00:00 avec time diff 14400.0
secondes
Date après l'anomalie 2: 2022-11-27 10:00:41+00:00 avec time diff 14400.0
secondes

--- Anomalie détectée ---
Date de l'anomalie: 2022-12-04 14:00:47+00:00
Time diff de l'anomalie: 86400.0 secondes

Temps avant l'anomalie :
Date avant l'anomalie 2: 2022-12-03 10:00:47+00:00 avec time diff 14400.0
secondes
Date avant l'anomalie 1: 2022-12-03 14:00:47+00:00 avec time diff 14400.0
secondes

Temps après l'anomalie :
Date après l'anomalie 1: 2022-12-04 18:00:47+00:00 avec time diff 14400.0
secondes
Date après l'anomalie 2: 2022-12-04 22:00:47+00:00 avec time diff 14400.0
secondes

--- Anomalie détectée ---
Date de l'anomalie: 2022-12-06 10:58:01+00:00
Time diff de l'anomalie: 43200.0 secondes

Temps avant l'anomalie :
Date avant l'anomalie 2: 2022-12-05 18:58:01+00:00 avec time diff 14402.0
secondes
Date avant l'anomalie 1: 2022-12-05 22:58:01+00:00 avec time diff 14400.0
secondes

Temps après l'anomalie :
Date après l'anomalie 1: 2022-12-06 14:58:01+00:00 avec time diff 14400.0
secondes
Date après l'anomalie 2: 2022-12-06 18:58:01+00:00 avec time diff 14400.0
secondes
```

[98]:
```
conda install -c conda-forge pandoc
```

```
Collecting package metadata (current_repodata.json): ...working... done
Solving environment: ...working... done

## Package Plan ##

  environment location: C:\Users\Ilyasse\anaconda3

  added / updated specs:
    - pandoc
```

```
The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    ca-certificates-2024.8.30  |       h56e8100_0         155 KB  conda-forge
    certifi-2024.8.30          |     pyhd8ed1ab_0         160 KB  conda-forge
    pandoc-3.3                 |       h57928b3_0        24.0 MB  conda-forge
    ------------------------------------------------------------
                                           Total:        24.3 MB

The following NEW packages will be INSTALLED:

  pandoc             conda-forge/win-64::pandoc-3.3-h57928b3_0

The following packages will be UPDATED:

  ca-certificates    pkgs/main::ca-certificates-2023.08.22~ --> conda-forge::ca-
certificates-2024.8.30-h56e8100_0
  certifi            pkgs/main/win-64::certifi-2023.7.22-p~ --> conda-
forge/noarch::certifi-2024.8.30-pyhd8ed1ab_0



Downloading and Extracting Packages

ca-certificates-2024 | 155 KB    |            |   0%

pandoc-3.3           | 24.0 MB   |            |   0%


certifi-2024.8.30    | 160 KB    |            |   0%

pandoc-3.3           | 24.0 MB   |            |   0%


certifi-2024.8.30    | 160 KB    | #          |  10%
ca-certificates-2024 | 155 KB    | #          |  10%

pandoc-3.3           | 24.0 MB   |            |   0%


certifi-2024.8.30    | 160 KB    | ######     |  60%
ca-certificates-2024 | 155 KB    | ####1      |  41%

pandoc-3.3           | 24.0 MB   | 1          |   1%
```

```
certifi-2024.8.30    | 160 KB    | ########## | 100%


certifi-2024.8.30    | 160 KB    | ########## | 100%
ca-certificates-2024 | 155 KB    | ########## | 100%
ca-certificates-2024 | 155 KB    | ########## | 100%

pandoc-3.3           | 24.0 MB   | 3          |   4%

pandoc-3.3           | 24.0 MB   | #1         |  12%

pandoc-3.3           | 24.0 MB   | #8         |  18%

pandoc-3.3           | 24.0 MB   | ##1        |  21%

pandoc-3.3           | 24.0 MB   | ##9        |  29%

pandoc-3.3           | 24.0 MB   | ###3       |  33%

pandoc-3.3           | 24.0 MB   | ###6       |  37%

pandoc-3.3           | 24.0 MB   | ####4      |  44%

pandoc-3.3           | 24.0 MB   | ####9      |  49%

pandoc-3.3           | 24.0 MB   | #####3     |  54%

pandoc-3.3           | 24.0 MB   | #####7     |  58%

pandoc-3.3           | 24.0 MB   | ######8    |  69%

pandoc-3.3           | 24.0 MB   | #######4   |  74%

pandoc-3.3           | 24.0 MB   | #######9   |  79%

pandoc-3.3           | 24.0 MB   | ########3  |  84%

pandoc-3.3           | 24.0 MB   | ########8  |  88%

pandoc-3.3           | 24.0 MB   | #########2 |  92%

pandoc-3.3           | 24.0 MB   | #########6 |  96%

pandoc-3.3           | 24.0 MB   | ########## | 100%
```

```
Preparing transaction: ...working... done
Verifying transaction: ...working... done
Executing transaction: ...working... done

Note: you may need to restart the kernel to use updated packages.


==> WARNING: A newer version of conda exists. <==
  current version: 23.7.4
  latest version: 24.7.1

Please update conda by running

    $ conda update -n base -c defaults conda

Or to minimize the number of packages updated during conda update use

     conda install conda=24.7.1
```

[99]:
```
pip install --upgrade pandoc
```

```
Collecting pandoc
  Downloading pandoc-2.4.tar.gz (34 kB)
  Preparing metadata (setup.py): started
  Preparing metadata (setup.py): finished with status 'done'
Collecting plumbum (from pandoc)
  Obtaining dependency information for plumbum from https://files.pythonhosted.o
rg/packages/fa/08/53cf4fb6bebdfd2598e9d620a587229c3bfcc8df1a202289da07e5b282cd/p
lumbum-1.8.3-py3-none-any.whl.metadata
  Downloading plumbum-1.8.3-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: ply in c:\users\ilyasse\anaconda3\lib\site-
packages (from pandoc) (3.11)
Requirement already satisfied: pywin32 in c:\users\ilyasse\anaconda3\lib\site-
packages (from plumbum->pandoc) (305.1)
Downloading plumbum-1.8.3-py3-none-any.whl (127 kB)
   ---------------------------------------- 0.0/127.6 kB ? eta -:--:--
   --------- ------------------------------ 30.7/127.6 kB 660.6 kB/s eta 0:00:01
   ------------------ --------------------- 61.4/127.6 kB 825.8 kB/s eta 0:00:01
   ---------------------------------------- 127.6/127.6 kB 938.9 kB/s eta 0:00:00
Building wheels for collected packages: pandoc
  Building wheel for pandoc (setup.py): started
```

```
  Building wheel for pandoc (setup.py): finished with status 'done'
  Created wheel for pandoc: filename=pandoc-2.4-py3-none-any.whl size=34819
sha256=18d3d2e383083bd8865308de862b1bcfa35f5334bf5d3bbdc0a86cdd3ba8f8de
  Stored in directory: c:\users\ilyasse\appdata\local\pip\cache\wheels\4f\d7\32\
c6c9b7b05e852e920fd72174487be3a0f18e633a7adcc303be
Successfully built pandoc
Installing collected packages: plumbum, pandoc
Successfully installed pandoc-2.4 plumbum-1.8.3
Note: you may need to restart the kernel to use updated packages.
```

[ ]: