AnalizaGrafoviSjever_2022

May 20, 2025

1 Analiza podataka i njihov grafički prikaz

1.0.1 U ovom jupyter notebooku se analiziraju vrijednosti mjerene na kalorimetru koji se nalazi na sjeveru 11. kata C zgrade FER-a u 2022. godini. Bilo je potrebno podatke filtrirati kako bi maknuli neželjena odstupanja koja nam mogu bitno naškoditi skupu za učenje kojeg koristimo pri kreiranju modela. Sve tehnike obrade podataka su prikazane uz vezano svojstvo.

[ ]:

```

```

instaliranje nužnih library - ja

[30]:

```
!pip install pandas openpyxl
```

```
Requirement already satisfied: pandas in c:\users\martin\anaconda3\lib\site-
packages (2.2.2)
Requirement already satisfied: openpyxl in c:\users\martin\anaconda3\lib\site-
packages (3.1.5)
Requirement already satisfied: numpy>=1.26.0 in
c:\users\martin\anaconda3\lib\site-packages (from pandas) (1.26.4)
Requirement already satisfied: python-dateutil>=2.8.2 in
c:\users\martin\anaconda3\lib\site-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in
c:\users\martin\anaconda3\lib\site-packages (from pandas) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in
c:\users\martin\anaconda3\lib\site-packages (from pandas) (2023.3)
Requirement already satisfied: et-xmlfile in c:\users\martin\anaconda3\lib\site-
packages (from openpyxl) (1.1.0)
Requirement already satisfied: six>=1.5 in c:\users\martin\anaconda3\lib\site-
packages (from python-dateutil>=2.8.2->pandas) (1.16.0)
```

1.1 Ucitavanje podataka iz csv fileova

[32]:

```
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline

# Pathovi CSV fileova
file_path = "../../../data/2022/23/calorimeter_23_year_2022.csv"
```

```
file_path1="../../../data/2022/23/zones_23_year_2022.csv"


# citanje CSV fileova
data = pd.read_csv(file_path, index_col=[0])
data1=pd.read_csv(file_path1, index_col=[0])
```

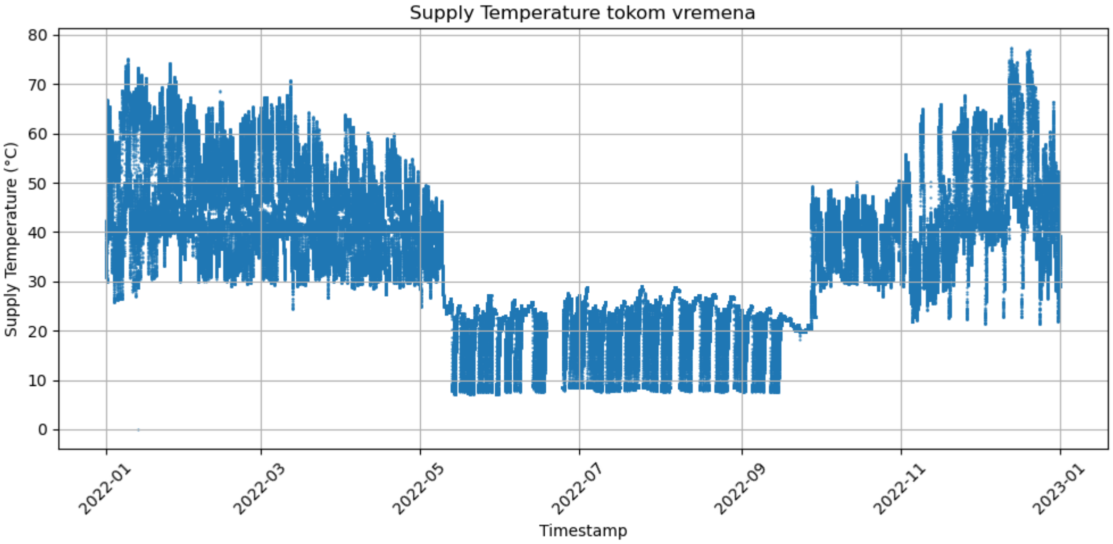
1.1.1 Analiza supply_temperature

[34]:

```
df = pd.DataFrame(data)  #ucitavamo dataframe
```

[35]:

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


df['timestamp'] = pd.to_datetime(df['timestamp'])


plt.figure(figsize=(10, 5))


df.plot.scatter(x='timestamp', y='supply_temperature', s=0.3, ax=plt.gca())


plt.xlabel('Timestamp')
plt.ylabel('Supply Temperature (°C)')
plt.title('Supply Temperature tokom vremena')
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()

# Prikazivanje grafa
plt.show()
```



[36]:
```
filtered_data = df[df['supply_temperature'] != 0]

print(f"Maknuti podatci u kojima je supply_temperature 0")
df=filtered_data.copy()
```

```
Maknuti podatci u kojima je supply_temperature 0
```

[37]:
```
import pandas as pd

# racunanje srednje vrijednosti i devijacije
mu = df['supply_temperature'].mean()
sigma = df['supply_temperature'].std()

# primjena 3-sigma pravila
filtered_df = df[(df['supply_temperature'] >= mu - 3*sigma) &
                 (df['supply_temperature'] <= mu + 3*sigma)]

df = filtered_df.copy()
print(f"Primjenjena 3-sigma za supply_temperature.")
```

```
Primjenjena 3-sigma za supply_temperature.
```

Vizualno iz grafa možemo odrediti da je granica uzmeđu hlađenja i grijanja oko 23 stupnjeva.

[39]:
```
df['interval_d'] = df['timestamp'].dt.to_period('D')

grupe_intervala = df.groupby('interval_d')

rezultat = []
```

```python
for interval, group in grupe_intervala:
    ukupno = len(group)
    ispod_23 = len(group[group['supply_temperature'] < 23])
    iznad_23 = ukupno - ispod_23

    # Odredimo većinu
    vecina = "Ispod 23°C" if ispod_23 > iznad_23 else "Iznad 23°C"

    # Dodamo rezultate u listu
    rezultat.append({
        'interval': interval,
        'ukupno_podataka': ukupno,
        'ispod_23': ispod_23,
        'iznad_23': iznad_23,
        'vecina': vecina
    })

df_intervals = pd.DataFrame(rezultat)

start_date = pd.Period("2022-05-05", freq='D')
end_date = pd.Period("2022-05-20", freq='D')

# Filtriranje podataka
konacni = df_intervals[(df_intervals['interval'] > start_date) &␣
 ↪(df_intervals['interval'] < end_date)]

konacni
```

```
[39]:        interval  ukupno_podataka  ispod_23  iznad_23      vecina
      125  2022-05-06             1440         0      1440  Iznad 23°C
      126  2022-05-07             1440         0      1440  Iznad 23°C
      127  2022-05-08             1440         0      1440  Iznad 23°C
      128  2022-05-09             1440         0      1440  Iznad 23°C
      129  2022-05-10             1440         0      1440  Iznad 23°C
      130  2022-05-11             1440         0      1440  Iznad 23°C
      131  2022-05-12             1440         0      1440  Iznad 23°C
      132  2022-05-13             1440       967       473  Ispod 23°C
      133  2022-05-14             1440      1440         0  Ispod 23°C
      134  2022-05-15             1440       290      1150  Iznad 23°C
      135  2022-05-16             1440      1184       256  Ispod 23°C
      136  2022-05-17             1440      1398        42  Ispod 23°C
      137  2022-05-18             1440      1432         8  Ispod 23°C
      138  2022-05-19             1440      1440         0  Ispod 23°C
```

Iz sljedećih podataka možemo zaključiti da je 2019-05-13 završio period grijanja, a krenuo period hlađenja.

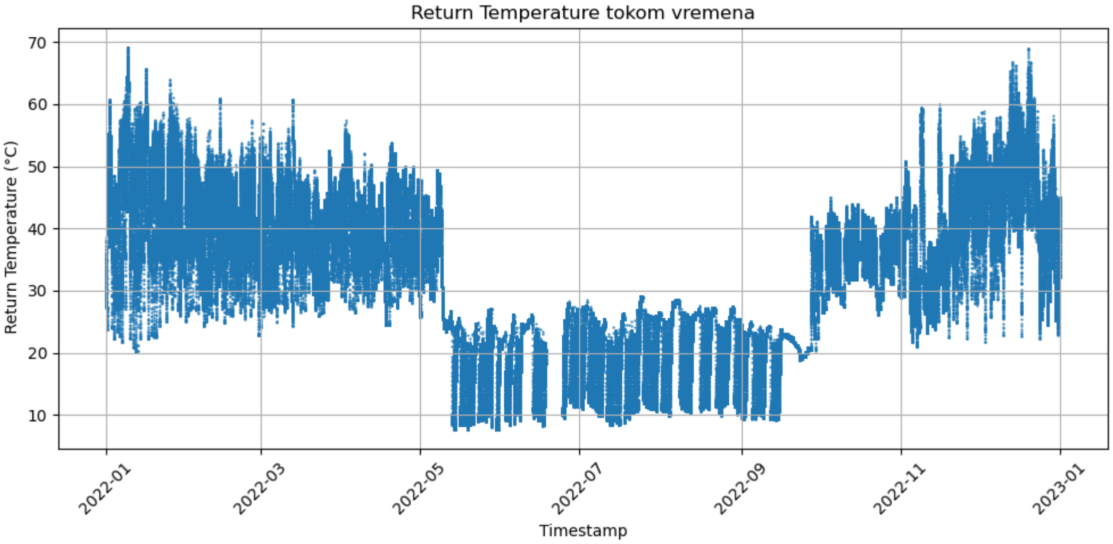
[41]:
```
start_date = pd.Period("2022-09-20", freq='D')
end_date = pd.Period("2022-10-10", freq='D')

# Filtriranje podataka
konacni = df_intervals[(df_intervals['interval'] > start_date) &␣
 ↪(df_intervals['interval'] < end_date)]
konacni
```

[41]:
```
       interval  ukupno_podataka  ispod_23  iznad_23      vecina
258  2022-09-21             1440      1440         0  Ispod 23°C
259  2022-09-22             1440      1440         0  Ispod 23°C
260  2022-09-23             1440      1440         0  Ispod 23°C
261  2022-09-24             1440      1440         0  Ispod 23°C
262  2022-09-25             1440      1440         0  Ispod 23°C
263  2022-09-26             1440      1440         0  Ispod 23°C
264  2022-09-27             1440       831       609  Ispod 23°C
265  2022-09-28             1440         0      1440  Iznad 23°C
266  2022-09-29             1334       179      1155  Iznad 23°C
267  2022-09-30             1440         0      1440  Iznad 23°C
268  2022-10-01             1440         0      1440  Iznad 23°C
269  2022-10-02             1440         0      1440  Iznad 23°C
270  2022-10-03             1440         0      1440  Iznad 23°C
271  2022-10-04             1440         0      1440  Iznad 23°C
272  2022-10-05             1440         0      1440  Iznad 23°C
273  2022-10-06             1440         0      1440  Iznad 23°C
274  2022-10-07             1440         0      1440  Iznad 23°C
275  2022-10-08             1440         0      1440  Iznad 23°C
276  2022-10-09             1440         0      1440  Iznad 23°C
```

Iz sljedećih podataka možemo zaključiti da je 2019-09-28 završio period hlađenja, a krenuo period grijanja.

1.1.2 Analiza return_temperature

[44]:
```
#  'timestamp' se pretvara u  datetime
df['timestamp'] = pd.to_datetime(df['timestamp'])

plt.figure(figsize=(10, 5))
# crtanje grafa
df.plot.scatter(x='timestamp', y='return_temperature', s=0.3, ax=plt.gca())
plt.xlabel('Timestamp')
plt.ylabel(' Return Temperature (°C)')
plt.title('Return Temperature tokom vremena')
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



[45]:
```
filtered_data = df[df['return_temperature'] != 0]

print(f"Maknuti podatci u kojima je return temp 0")
df = filtered_data.copy()
```

```
Maknuti podatci u kojima je return temp 0
```

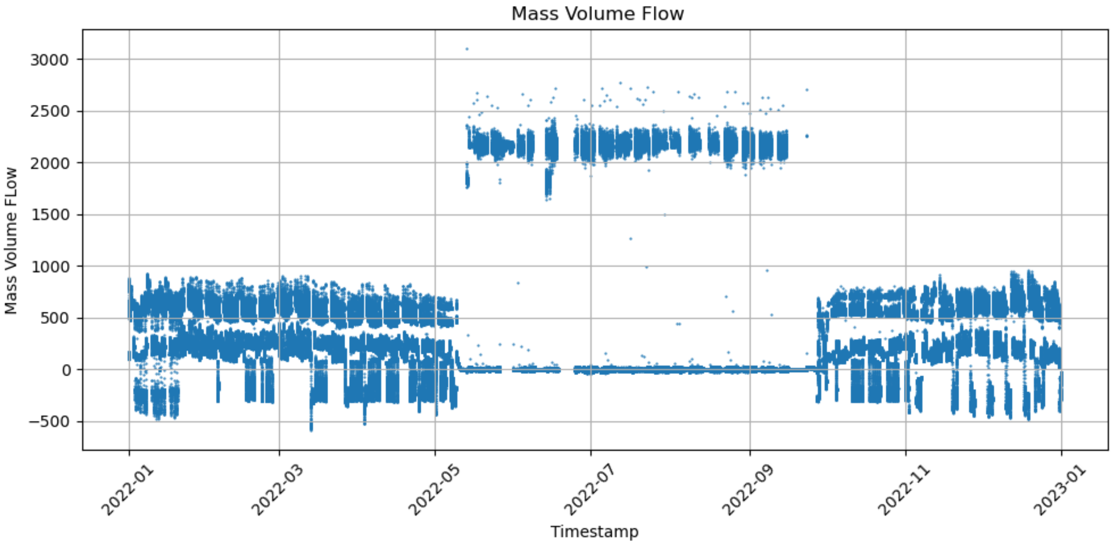
[46]:
```
import pandas as pd

mu = df['return_temperature'].mean()
sigma = df['return_temperature'].std()


filtered_df = df[(df['return_temperature'] >= mu - 3*sigma) &
                 (df['return_temperature'] <= mu + 3*sigma)]

df = filtered_df.copy()
print(f"Primjenjena 3-sigma za return_temperature.")
```

```
Primjenjena 3-sigma za return_temperature.
```

1.1.3 Analiza mass_volume_flow

[48]:
```
#  'timestamp' se pretvara u  datetime
df['timestamp'] = pd.to_datetime(df['timestamp'])

plt.figure(figsize=(10, 5))
# crtanje grafa
df.plot.scatter(x='timestamp', y='mass_volume_flow', s=0.3, ax=plt.gca())
```

```
plt.xlabel('Timestamp')
plt.ylabel(' Mass Volume FLow ')
plt.title(' Mass Volume Flow')
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

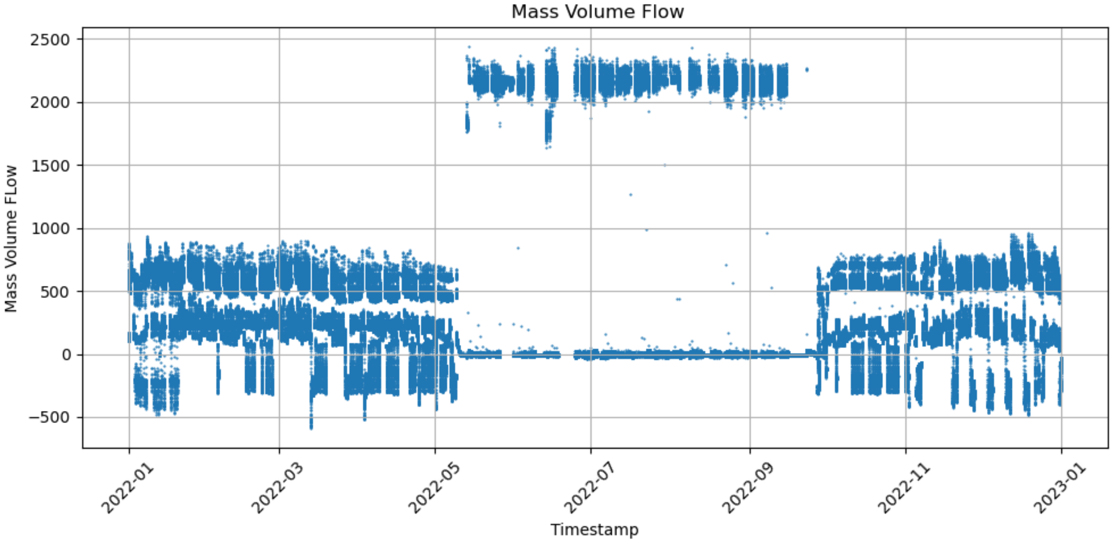


[49]:
```
import pandas as pd

# racunanje srednje vrijednosti i devijacije
mu = df['mass_volume_flow'].mean()
sigma = df['mass_volume_flow'].std()

# primjena 3-sigma pravila
filtered_df = df[(df['mass_volume_flow'] >= mu - 3*sigma) &
                  (df['mass_volume_flow'] <= mu + 3*sigma)]

df = filtered_df.copy()
print(f"Primjenjena 3-sigma za mass_volume_flow.")
```

```
Primjenjena 3-sigma za mass_volume_flow.
```

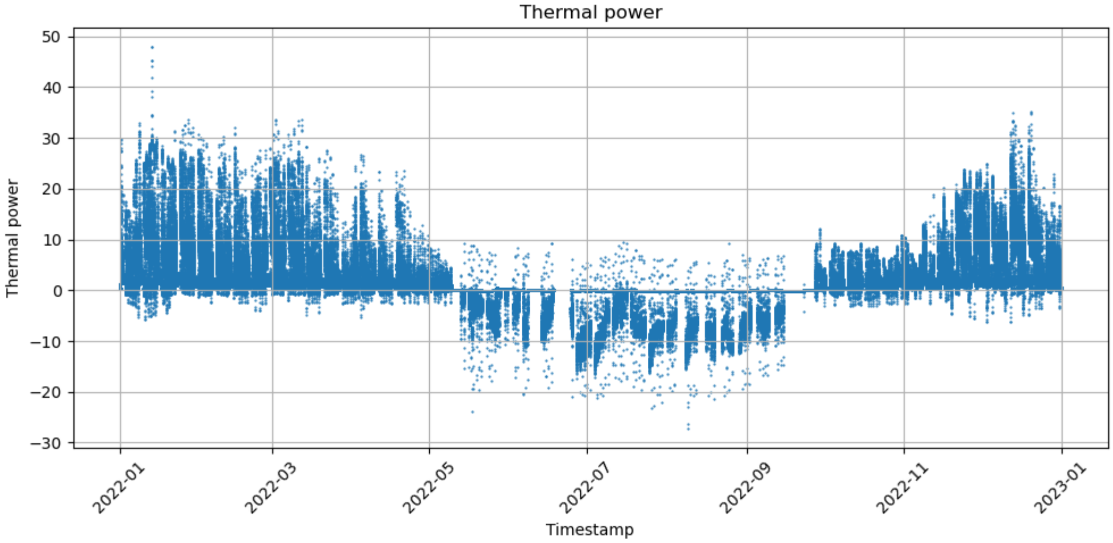
Mass_volume flow nakon 3-sigma

[51]:
```
plt.figure(figsize=(10, 5))
# crtanje grafa
df.plot.scatter(x='timestamp', y='mass_volume_flow', s=0.3, ax=plt.gca())
plt.xlabel('Timestamp')
plt.ylabel(' Mass Volume FLow ')
plt.title(' Mass Volume Flow')
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



1.1.4 Analiza thermal_power

[53]:
```
#  'timestamp' se pretvara u  datetime
df['timestamp'] = pd.to_datetime(df['timestamp'])
plt.figure(figsize=(10, 5))
# crtanje grafa
df.plot.scatter(x='timestamp', y='thermal_power', s=0.3, ax=plt.gca())
plt.xlabel('Timestamp')
plt.ylabel('Thermal power')
plt.title('Thermal power')
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



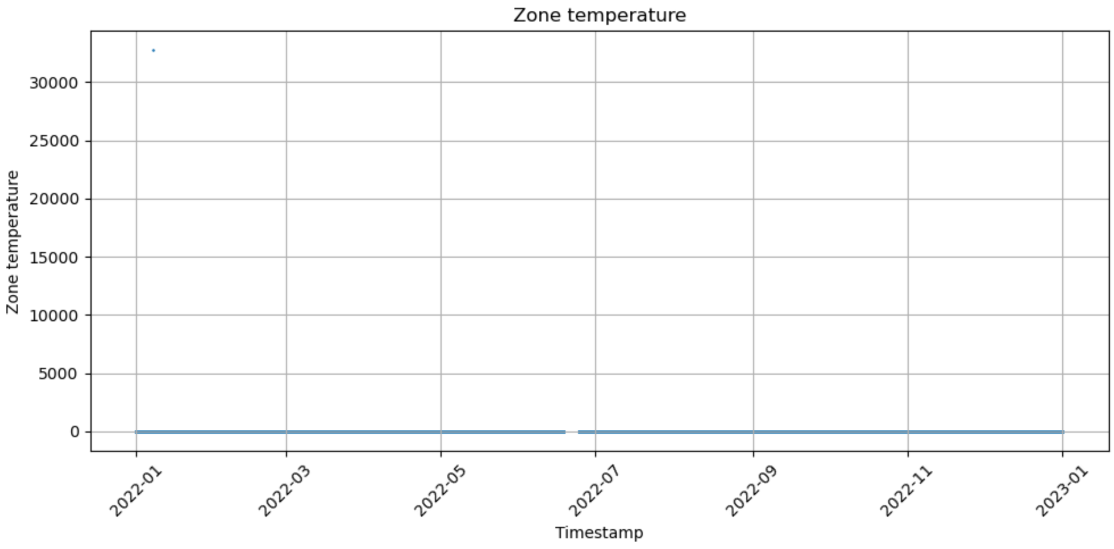
1.1.5 Analiza zone_temperature

[55]:

```
df1 = pd.DataFrame(data1)
```

[56]:

```
df1['timestamp'] = pd.to_datetime(df1['timestamp'])
plt.figure(figsize=(10, 5))

df1.plot.scatter(x='timestamp', y='zone_temperature', s=0.3, ax=plt.gca())
plt.xlabel('Timestamp')
plt.ylabel('Zone temperature')
plt.title('Zone temperature')
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



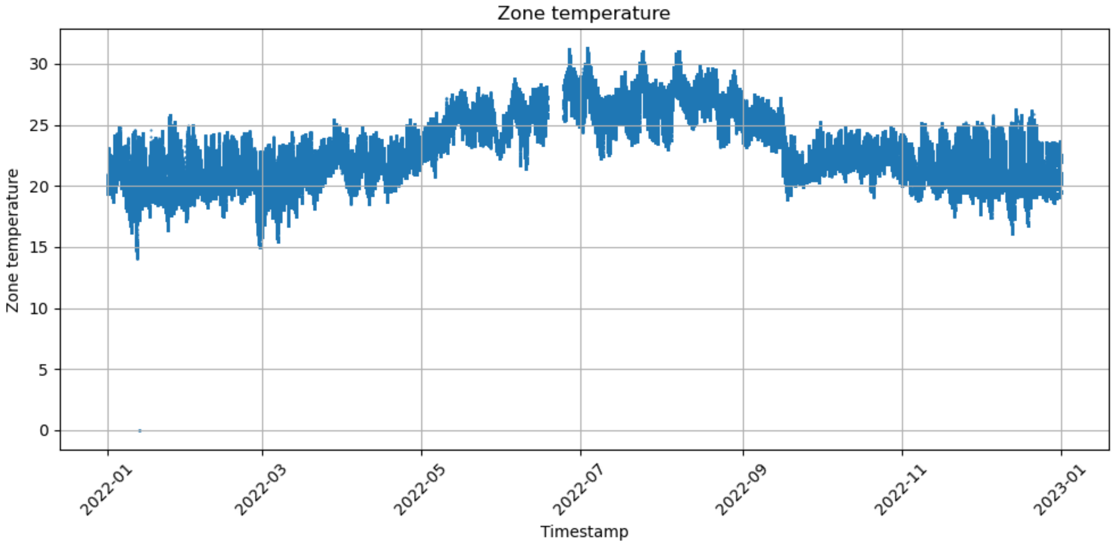
[57]:
```
import pandas as pd



mu = df1['zone_temperature'].mean()
sigma = df1['zone_temperature'].std()



filtered_df1 = df1[(df1['zone_temperature'] >= mu - 3*sigma) &
                   (df1['zone_temperature'] <= mu + 3*sigma)]

df1 = filtered_df1.copy()
print(f"Primjenjena 3-sigma za zone_temperature.")
```

```
Primjenjena 3-sigma za zone_temperature.
```

Prikaz zone_temperature nakon 3-sigma

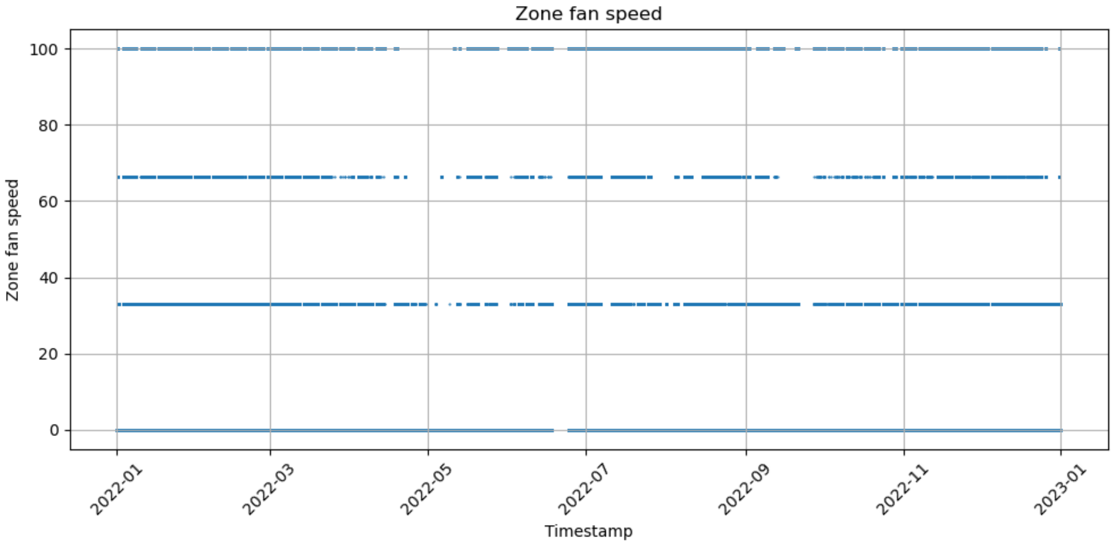
zbog velikih odstupanja prikazujemo novi graf

[60]:
```
df1['timestamp'] = pd.to_datetime(df1['timestamp'])
plt.figure(figsize=(10, 5))
# crtanje grafa
df1.plot.scatter(x='timestamp', y='zone_temperature', s=0.3, ax=plt.gca())
plt.xlabel('Timestamp')
plt.ylabel('Zone temperature')
plt.title('Zone temperature')
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



1.1.6 Analiza zone_fan_speed

[62]:
```
df1['timestamp'] = pd.to_datetime(df1['timestamp'])

plt.figure(figsize=(10, 5))

df1.plot.scatter(x='timestamp', y='zone_fan_speed', s=0.3, ax=plt.gca())
plt.xlabel('Timestamp')
plt.ylabel('Zone fan speed')
plt.title('Zone fan speed')
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

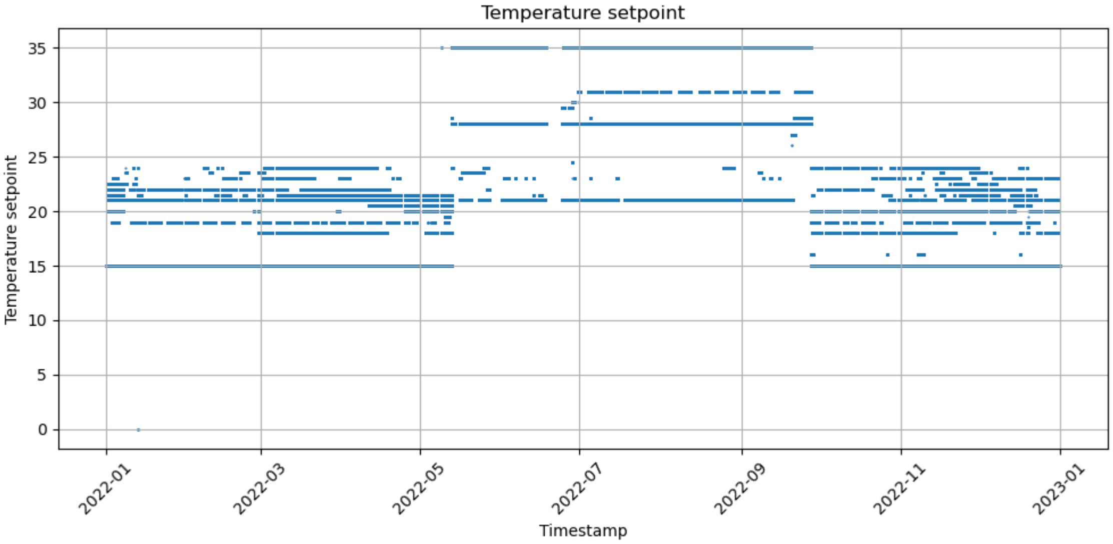


[63]:
```
import pandas as pd



mu = df1['zone_fan_speed'].mean()
sigma = df1['zone_fan_speed'].std()



filtered_df1 = df1[(df1['zone_fan_speed'] >= mu - 3*sigma) &
                   (df1['zone_fan_speed'] <= mu + 3*sigma)]

df1 = filtered_df1.copy()
print(f"Primjenjena 3-sigma za zone_fan_speed.")
```

```
Primjenjena 3-sigma za zone_fan_speed.
```

1.1.7 Analiza temperature_setpoint

[65]:
```
df1['timestamp'] = pd.to_datetime(df1['timestamp'])
plt.figure(figsize=(10, 5))

df1.plot.scatter(x='timestamp', y='temperature_setpoint', s=0.3, ax=plt.gca())
plt.xlabel('Timestamp')
plt.ylabel('Temperature setpoint')
plt.title('Temperature setpoint')
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



[66]:
```
filtered_data = df1[df1['temperature_setpoint'] != 0]

print(f"Maknuti podatci u kojima je temperature_setpoint 0")
df1 = filtered_data.copy()
```

```
Maknuti podatci u kojima je temperature_setpoint 0
```

1.1.8 Exportanje rezultata u CVS fileove

[68]:
```
df.to_csv('../../../results/2022/23/calorimeter_23_year_2022filtriranje.csv',␣
    ↪index=False)
df1.to_csv('../../../results/2022/23/zones_23_year_2022filtriranje.csv',␣
    ↪index=False)
print("Upisani podatci")
```

```
Upisani podatci
```

1.1.9 Na kraju filtrirane podatke pohranjujemo u zasebne CSV fileove

[ ]:
```

```