In [1]:
```
# imports
import pandas as pd
import agrilution_aws
import logging
import boto3
from datetime import datetime
import sys
from boto3.dynamodb.conditions import Key, Attr
import time
from agrilution_aws import DynamoDbApi
from matplotlib.pyplot import figure
from matplotlib import pyplot as plt
import seaborn as sns
import plotly.express as px
import dask.dataframe as dd
import plotly.graph_objects as go
from plotly.subplots import make_subplots
from plotly import tools
import warnings
warnings.filterwarnings('ignore')
pd.options.mode.chained_assignment = None
import dask.dataframe as dd
```

In [2]:
```
# globals

# dynamoDB API
dynamo = DynamoDbApi(logging.getLogger(), table_name = 'archive')
# timestamp in ms marking 1st of may
timestamp2 = 1651363200000
```

In [3]:
```
# list of all lab cubes
plantcubes = {
    'A1_lower': 'd6472f5d-94f9-4a31-9a8e-ddc6744023d6',
    'A1_upper': 'bf6b3065-a5ad-49f0-96e3-f1ed22e55e18',
    'A2_lower': '07b17561-3b04-4094-a8ab-2f67315adfdd',
    'A2_upper': '2ba34bbe-1611-4c9b-8a5e-1c802ff77768',
    'A3_lower': '26b03d30-3a9d-4460-a0cb-7ef5c1d5dec8',
    'A3_upper': '955605fe-8666-449b-96b4-e973b1e197da',
    'A4_lower': '09ef2ce0-2f99-45cf-8cb5-99550fca494f',
    'A4_upper': 'b637f6a6-b6e2-486c-86db-cc431d0b2a58',
    'B1_lower': '5a9039ae-957b-42b2-9d09-3baf73cf0020',
    'B1_upper': '0b66fd54-465b-409f-838f-ca5e494e68fb',
    'B2_lower': 'd9dd3086-fe92-4cab-b235-be2b283c4999',
    'B2_upper': '2853d150-f30a-4f35-a4fc-5985b35876dc',
    'B3_lower': 'a27588d5-bc01-44ab-b96d-cad7f86402b0',
    'B3_upper': 'd22ff6af-211b-4743-a5ae-5fd89ffbe446',
    'B4_lower': '11c45cd6-8d1f-4140-a545-0db886918e3b',
    'B4_upper': '510d7df1-234c-46f8-a153-ec792edc93b1',
    'C1_lower': '0427a2fa-8a50-4d00-ad56-6246c03ef9d0',
    'C1_upper': 'eac52b39-02c0-4a7a-a9e5-010709ee15c8',
    'C2_lower': 'ab713fff-4bd2-4a72-afdd-603e31b57689',
    'C2_upper': '09aefdec-f638-4e2d-91d2-375094a3d881',
    'C3_lower': '8cb8a481-a70d-4988-b419-d905d06ca65d',
    'C3_upper': '7d53b428-7777-47f0-9605-01ac8bda96f4',
    'C4_lower': '1acd7d04-fb3b-4983-abbf-24053e3a1499',
    'C4_upper': '5b23e086-1365-48a2-af39-defa77768aa5',
    'D1_lower': '5ae3a1b3-5354-4b23-ab83-aa9f3029098d',
    'D1_upper': '820b0870-b586-45b8-9a1e-fdd41a842f5d',
    'D2_lower': 'd183f2bd-d1df-4f83-a34d-6c72601b97f2',
    'D2_upper': '69a5e2a3-624c-4522-b0ee-ee28846fc700',
    'D3_lower': 'f598f96e-b0f4-4009-85e1-e621e8306c36',
    'D3_upper': '9788f724-0b7a-47ae-8e95-2c35152e20b8',
    'E1' : '2933af4a-51d4-4894-aa60-753219ca1918',
    'E2' : 'f29ffb36-be56-46e1-9e9d-d05e44e9a1a0',
    'E3': 'b2d1811e-dbff-4fcb-a219-468adfb045ea',
    'E4' : '422d6453-a501-4ed9-bd4d-02b510a6e6d7',
    'E5' : '2b9c5df5-e286-4f0a-ab87-2271535677b6',
    'E6' : '12652341-6356-4c7f-9a60-eb5d82b16a57',
    'E7' : '52fdc759-32a3-43da-8207-3e4b89bafaae',
    'E8' : '424b5b0a-724f-4ab6-9688-f3fb2fab1cef',
    'E9' : 'dfde8871-522f-4ee5-a572-82049fe112cd',
    'E10' : 'c9299fd6-a636-4487-a252-837399139e8e',
```

```
}
```

In [4]:
```
#reading the file as dask dataframe
df2 = pd.read_csv('May.csv')
```

In [5]:
```
#converting it into pandas dataframe
#df2 = df2.compute()
```

In [6]:
```
df2.columns
```

Out[6]:
```
Index(['Unnamed: 0', 'connected', 'plantcube', 'timestamp', 'firmware_mcu',
       'verbose_reporting', 'firmware_ncu', 'tank_level_raw', 'total_offset',
       'user_button', 'cooling', 'signal_led', 'valve', 'tank_level', 'rssi',
       'wifi_level', 'door', 'recipe_id', 'temp_b', 'led_a_board_state',
       'temp_a', 'temp_tank', 'light_b', 'light_a', 'humid_b', 'humid_a',
       'fan_led_a', 'fan_b', 'fan_led_b', 'pump', 'fan_a', 'fan_led_b_tacho',
       'fan_a_tacho', 'fan_led_a_tacho', 'fan_b_tacho', 'led_a', 'led_b',
       'temp_led_a', 'temp_led_b', 'ec', 'user_offset', 'mode',
       'led_b_board_state', 'owner', 'seedbundle_variant'],
      dtype='object')
```

In [7]:
```
#extracting the required columns
cols = [1,2,3,17,18,20,41]
df2 = df2[df2.columns[cols]]
```

In [8]:
```
df2.head()
```

Out[8]:

	connected	plantcube	timestamp	recipe_id	temp_b	temp_a	mode
0	True	d6472f5d-94f9-4a31-9a8e-ddc6744023d6	1651475817282	NaN	NaN	NaN	NaN
1	NaN	d6472f5d-94f9-4a31-9a8e-ddc6744023d6	1651475817441	NaN	NaN	NaN	NaN
2	NaN	d6472f5d-94f9-4a31-9a8e-ddc6744023d6	1651475817520	NaN	NaN	NaN	NaN
3	NaN	d6472f5d-94f9-4a31-9a8e-ddc6744023d6	1651475817599	NaN	NaN	NaN	NaN
4	NaN	d6472f5d-94f9-4a31-9a8e-ddc6744023d6	1651475817608	NaN	NaN	NaN	NaN

In [9]:
```
#Number of Null values in each column
df2.isnull().sum()
```

Out[9]:
```
connected    23033510
plantcube           0
timestamp           0
recipe_id    23041436
temp_b       21819544
temp_a       21678655
mode         23042598
dtype: int64
```

In [10]:
```
#dropping the row if all the values in the given columns are NA
df2.dropna(subset=['connected','recipe_id','temp_b','temp_a','mode'], how='all', inplace=True)
```

In [11]:
```
df2.head()
```

Out[11]:

	connected	plantcube	timestamp	recipe_id	temp_b	temp_a	mode
0	True	d6472f5d-94f9-4a31-9a8e-ddc6744023d6	1651475817282	NaN	NaN	NaN	NaN
6	NaN	d6472f5d-94f9-4a31-9a8e-ddc6744023d6	1651475818230	1.000000e+00	NaN	NaN	NaN
7	NaN	d6472f5d-94f9-4a31-9a8e-ddc6744023d6	1651475818761	1.650463e+09	22.27	22.22	NaN
8	NaN	d6472f5d-94f9-4a31-9a8e-ddc6744023d6	1651475818845	1.000000e+00	NaN	NaN	NaN
9	NaN	d6472f5d-94f9-4a31-9a8e-ddc6744023d6	1651475818924	1.650463e+09	NaN	NaN	NaN

In [12]:
```
#converting timestamp to datetime format
df2['timestamp'] = df2['timestamp'].astype('int64')
df2['timestamp'] = pd.to_datetime(df2['timestamp'], unit='ms')
```

In [13]:
```
#replacing the plantcube name with their alias names
dict1 = {v : k for k, v in plantcubes.items()}
df2.plantcube = df2.plantcube.replace(dict1)
df2.head()
```

Out[13]:

	connected	plantcube	timestamp	recipe_id	temp_b	temp_a	mode
0	True	A1_lower	2022-05-02 07:16:57.282	NaN	NaN	NaN	NaN
6	NaN	A1_lower	2022-05-02 07:16:58.230	1.000000e+00	NaN	NaN	NaN
7	NaN	A1_lower	2022-05-02 07:16:58.761	1.650463e+09	22.27	22.22	NaN
8	NaN	A1_lower	2022-05-02 07:16:58.845	1.000000e+00	NaN	NaN	NaN
9	NaN	A1_lower	2022-05-02 07:16:58.924	1.650463e+09	NaN	NaN	NaN

In [14]:
```
df2 = df2.reset_index()
```

In [15]:
```
#applying ffill for the columns connected and recipe id
df2['connected'] = df2.groupby('plantcube')['connected'].apply(lambda x:x.fillna(method='ffill'))
df2['recipe_id'] = df2.groupby('plantcube')['recipe_id'].apply(lambda x:x.fillna(method='ffill'))
df2['mode'] = df2.groupby('plantcube')['mode'].apply(lambda x:x.fillna(method='ffill'))
```

In [16]:
```
df2.isnull().sum()
```

Out[16]:
```
index             0
connected     108556
plantcube          0
timestamp          0
recipe_id     178064
temp_b       1290060
temp_a       1149171
mode          350040
dtype: int64
```

In [17]:
```
#after forward filling the columns 'connected' and 'recipe id' in the above step. Remove the rows if any of these column
#values are null.
df2.dropna(subset=['connected','recipe_id'], how='any', inplace=True)
```

In [18]:
```
df2 = df2.drop('index', axis=1)
```

In [19]:
```
#changing the datatype of the temperature columns
df2['temp_a'] = df2['temp_a'].astype(float)
df2['temp_b'] = df2['temp_b'].astype(float)
```

In [20]:
```
#copying dataframe df2 to idata
idata = df2.copy()
```

In [21]:
```
#dropping the rows which contains the mode(debug)
t1 = idata[idata['mode'] == 1]
```

In [22]:
```
#total records in debug mode
t1.shape[0]
```

Out[22]: 54232

In [23]:
```
#remove the records in debug mode
idata = idata[idata['mode'] != 1]
```

In [24]:
```
idata.head()
```

Out[24]:

	connected	plantcube	timestamp	recipe_id	temp_b	temp_a	mode
1	True	A1_lower	2022-05-02 07:16:58.230	1.000000e+00	NaN	NaN	NaN
2	True	A1_lower	2022-05-02 07:16:58.761	1.650463e+09	22.27	22.22	NaN
3	True	A1_lower	2022-05-02 07:16:58.845	1.000000e+00	NaN	NaN	NaN
4	True	A1_lower	2022-05-02 07:16:58.924	1.650463e+09	NaN	NaN	NaN
5	True	A1_lower	2022-05-02 07:17:27.215	1.650463e+09	NaN	NaN	NaN

In [25]:
```
#adding recipe along with it.
rdf = pd.read_csv('Recipe_table_sApril')
```

In [26]:
```
rdf.drop('Unnamed: 0', axis=1, inplace=True)
```

In [27]:
```
rdf.head(30)
```

Out[27]:

	layers	plantcube	recipe_id
0	[[{'periods': [{'duration': Decimal('86400'), ...	A1_lower	1649666148
1	[[{'periods': [{'duration': Decimal('86400'), ...	A1_lower	1649854144
2	[[{'periods': [{'duration': Decimal('86400'), ...	A1_lower	1649934487
3	[[{'periods': [{'duration': Decimal('86400'), ...	A1_lower	1650446765
4	[[{'periods': [{'duration': Decimal('86400'), ...	A1_lower	1650462922
5	[[{'periods': [{'duration': Decimal('86400'), ...	A1_lower	1653983191
6	[[{'periods': [{'duration': Decimal('86400'), ...	A1_lower	1653983554
7	[[{'periods': [{'duration': Decimal('86400'), ...	A1_lower	1653985982
8	[[{'periods': [{'duration': Decimal('86400'), ...	A1_lower	1654078825
9	[[{'periods': [{'duration': Decimal('86400'), ...	A1_lower	1657093174
10	[[{'periods': [{'duration': Decimal('86400'), ...	A1_lower	1657281884
11	[[{'periods': [{'duration': Decimal('86400'), ...	A1_upper	1652711767
12	[[{'periods': [{'duration': Decimal('86400'), ...	A1_upper	1652711832
13	[[{'periods': [{'duration': Decimal('86400'), ...	A1_upper	1652711870
14	[[{'periods': [{'duration': Decimal('86400'), ...	A1_upper	1653900414
15	[[{'periods': [{'duration': Decimal('86400'), ...	A1_upper	1654078815
16	[[{'periods': [{'duration': Decimal('86400'), ...	A1_upper	1659961675
17	[[{'periods': [{'duration': Decimal('86400'), ...	A2_lower	1649666159
18	[[{'periods': [{'duration': Decimal('86400'), ...	A2_lower	1649754540
19	[[{'periods': [{'duration': Decimal('86400'), ...	A2_lower	1652084306
20	[[{'periods': [{'duration': Decimal('86400'), ...	A2_lower	1652184211
21	[[{'periods': [{'duration': Decimal('86400'), ...	A2_lower	1652863133
22	[[{'periods': [{'duration': Decimal('86400'), ...	A2_lower	1653910382
23	[[{'periods': [{'duration': Decimal('86400'), ...	A2_lower	1654078855
24	[[{'periods': [{'duration': Decimal('86400'), ...	A2_lower	1654078884

	layers	plantcube	recipe_id
25	[[{'periods': [{'duration': Decimal('86400'), ...	A2_lower	1657015628
26	[[{'periods': [{'duration': Decimal('86400'), ...	A2_lower	1657281891
27	[[{'periods': [{'duration': Decimal('86400'), ...	A2_lower	1662463213
28	[[{'periods': [{'duration': Decimal('86400'), ...	A2_lower	1662463214
29	[[{'periods': [{'duration': Decimal('86400'), ...	A2_lower	1662463215

In [28]:
```
#joining the recipe and archive table based on the attributes plantcube and recipe_id
jdf = pd.merge(idata, rdf, on=['plantcube','recipe_id'], how="left",indicator=True)
```

In [29]:
```
jdf.head()
```

Out[29]:

	connected	plantcube	timestamp	recipe_id	temp_b	temp_a	mode	layers	_merge
0	True	A1_lower	2022-05-02 07:16:58.230	1.000000e+00	NaN	NaN	NaN	NaN	left_only
1	True	A1_lower	2022-05-02 07:16:58.761	1.650463e+09	22.27	22.22	NaN	[[{'periods': [{'duration': Decimal('86400'), ...	both
2	True	A1_lower	2022-05-02 07:16:58.845	1.000000e+00	NaN	NaN	NaN	NaN	left_only
3	True	A1_lower	2022-05-02 07:16:58.924	1.650463e+09	NaN	NaN	NaN	[[{'periods': [{'duration': Decimal('86400'), ...	both
4	True	A1_lower	2022-05-02 07:17:27.215	1.650463e+09	NaN	NaN	NaN	[[{'periods': [{'duration': Decimal('86400'), ...	both

In [30]:
```
jdf.loc[jdf.recipe_id == 1, 'layers'] = "default recipe"
jdf.head()
```

Out[30]:

	connected	plantcube	timestamp	recipe_id	temp_b	temp_a	mode	layers	_merge
0	True	A1_lower	2022-05-02 07:16:58.230	1.000000e+00	NaN	NaN	NaN	default recipe	left_only
1	True	A1_lower	2022-05-02 07:16:58.761	1.650463e+09	22.27	22.22	NaN	[[{'periods': [{'duration': Decimal('86400'), ...	both
2	True	A1_lower	2022-05-02 07:16:58.845	1.000000e+00	NaN	NaN	NaN	default recipe	left_only
3	True	A1_lower	2022-05-02 07:16:58.924	1.650463e+09	NaN	NaN	NaN	[[{'periods': [{'duration': Decimal('86400'), ...	both
4	True	A1_lower	2022-05-02 07:17:27.215	1.650463e+09	NaN	NaN	NaN	[[{'periods': [{'duration': Decimal('86400'), ...	both

In [31]:
```
jdf.drop('_merge', axis=1, inplace=True)
```

In [32]:
```
jdf.head()
```

Out[32]:

	connected	plantcube	timestamp	recipe_id	temp_b	temp_a	mode	layers
0	True	A1_lower	2022-05-02 07:16:58.230	1.000000e+00	NaN	NaN	NaN	default recipe
1	True	A1_lower	2022-05-02 07:16:58.761	1.650463e+09	22.27	22.22	NaN	[[{'periods': [{'duration': Decimal('86400'), ...
2	True	A1_lower	2022-05-02 07:16:58.845	1.000000e+00	NaN	NaN	NaN	default recipe
3	True	A1_lower	2022-05-02 07:16:58.924	1.650463e+09	NaN	NaN	NaN	[[{'periods': [{'duration': Decimal('86400'), ...
4	True	A1_lower	2022-05-02 07:17:27.215	1.650463e+09	NaN	NaN	NaN	[[{'periods': [{'duration': Decimal('86400'), ...

In [33]:
```
jdf['layers'] = jdf.groupby('plantcube')['layers'].apply(lambda x:x.fillna(method='ffill'))
```

In [34]:
```
jdf['recipe']= jdf['layers'].map(str)
#changing the datatype of column 'recipe' to category
jdf['recipe']= jdf['recipe'].astype('category')
#converting the values in the column 'recipe' to numerical codes
jdf['recipe'] = jdf['recipe'].cat.codes
```

In [35]:
```
idata1 = jdf.copy()
```

In [36]:
```
idata1.recipe.unique()
```

Out[36]:
```
array([14,  6, 12,  1,  2,  9, 13, 10,  3, 11,  5,  0,  8,  7,  4],
      dtype=int8)
```

In [37]:
```
idata1.to_csv("preprocessed-May_Aug")
```

In [38]:
```
idata1 = pd.read_csv('preprocessed-May_Aug')
```

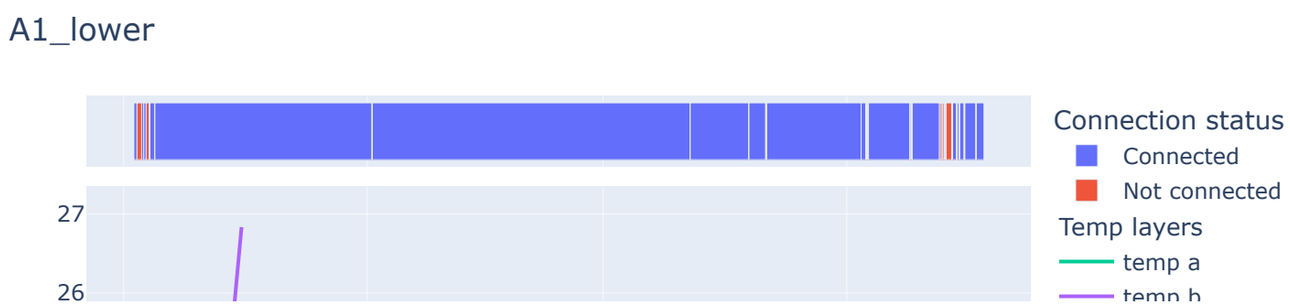
Plantcubes where sensors not working

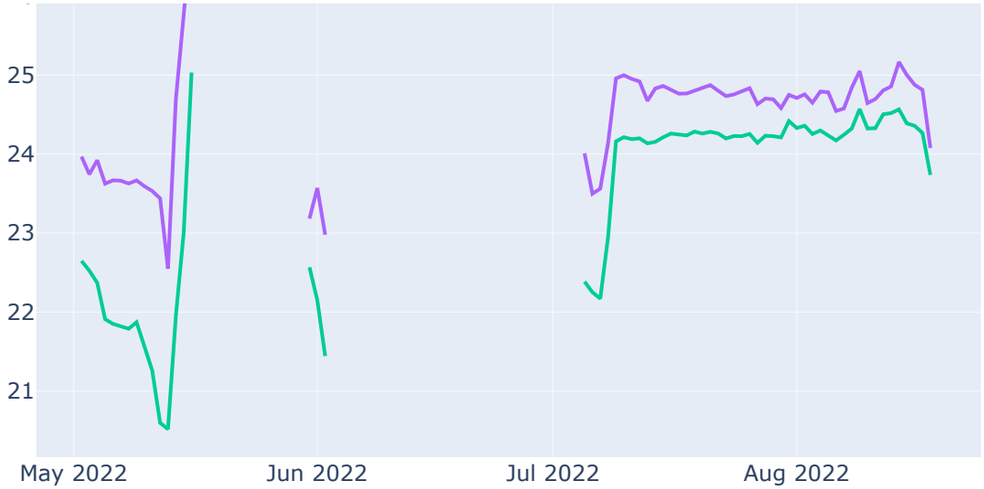
In [39]:
```
dataframes =['A1_lower','A1_upper','A2_lower','A2_upper','A3_lower','A3_upper','A4_lower','A4_upper','B1_lower','B1_upper','B2_lower','B2_upper','B3_
```

In [40]:
```
df = idata1[idata1.plantcube == 'A1_lower']

#set the timestamp as index
df['timestamp'] = pd.to_datetime(df['timestamp'])
df = df.set_index('timestamp')
df.head(10)

df.drop('Unnamed: 0', axis=1)

df['temp_a'] = df.resample('D')['temp_a'].apply(lambda x:x.interpolate(method="time",limit_direction = "forward"))
df['temp_b'] = df.resample('D')['temp_b'].apply(lambda x:x.interpolate(method="time",limit_direction = "forward"))
```

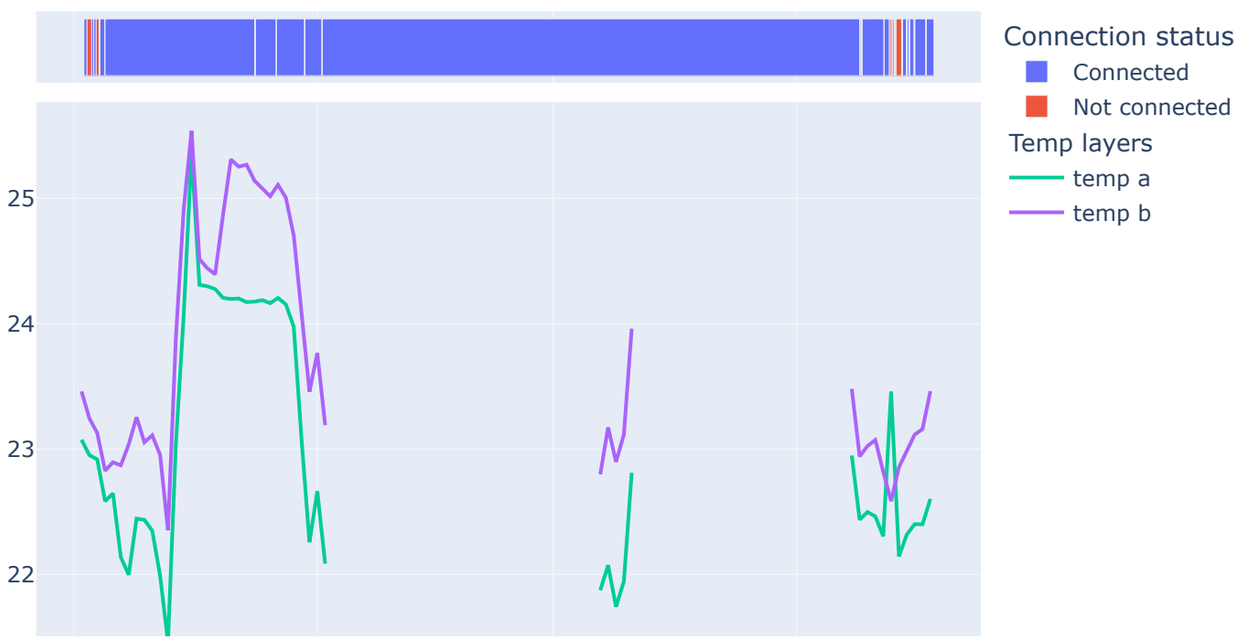
In [41]:
```
val = ""
def my_func(ndf):
    val = ndf
    # creating a dataframe to store the plantcube
    df = idata1[idata1.plantcube == val]
    df.head(10)

    #set the timestamp as index
    df['timestamp'] = pd.to_datetime(df['timestamp'])
    df = df.set_index('timestamp')

    df.drop('Unnamed: 0', axis=1)
    #interpolating the temperature values based on linear interpolation method after resampling it by date.
    df['temp_a'] = df.resample('D')['temp_a'].apply(lambda x:x.interpolate(method="time",limit_direction = "forward"))
    df['temp_b'] = df.resample('D')['temp_b'].apply(lambda x:x.interpolate(method="time",limit_direction = "forward"))

    #converting connected as category type
    df['connected']= df['connected'].astype('category')

    #instead of true and false, converting it into 0's and 1's
    df['connected'] = df['connected'].cat.codes

    #function to find out when the connection starts and ends
    current_event = None
    start_plantcube = None
    start_time = None
    time = None
    result = []
    for event, time ,plantcube in zip(df['connected'], df.index,df['plantcube']):
        if event != current_event:
            if current_event is not None and start_plantcube is not None and start_time is not None and time is not None:
                result.append([start_plantcube, current_event, start_time, time])
            current_event, start_time,start_plantcube = event, time,plantcube

    result.append([start_plantcube, current_event, start_time, time])
    ddata = pd.DataFrame(result, columns=['plantcube','connected','EventStartTime','EventEndTime'])

    #converting connected attribute to string values
    ddata['connected'] = ddata['connected'].astype(str)
    ddata['connected'].replace('0','Not connected',inplace=True)
    ddata['connected'].replace('1','Connected',inplace=True)
```

```
    #visualization of A1 lower plantcube for two layers (temp a and temp b)
    a1l_a = df[['temp_a']].resample('D').mean()
    a1l_b = df[['temp_b']].resample('D').mean()
    fig = go.Figure()
    trace1 = go.Scatter(x=a1l_a.index, y=a1l_a.temp_a, name="temp a", mode="lines",legendgroup="group2",legendgrouptitle_text="Temp layers")
    trace2 = go.Scatter(x=a1l_b.index, y=a1l_b.temp_b, name="temp b", mode="lines",legendgroup="group2",legendgrouptitle_text="Temp layers")

    #visulaization of connection status
    fig1 = px.timeline(
     ddata, x_start="EventStartTime", x_end="EventEndTime", y="plantcube",
     color='connected', height=200, width=1000
    )

    fig = tools.make_subplots(rows=2, cols=1,
                              figure=fig1,
                              shared_xaxes=True,
                              vertical_spacing=0.03,
                              row_width=[0.4, 0.05]
                             )
    fig.add_trace(trace1, row=2, col=1)
    fig.add_trace(trace2, row=2, col=1)
    fig.update_layout(xaxis2_showticklabels=True,height=500, width=750,showlegend=True,yaxis1={'visible': False, 'showticklabels': False},legend={"ti
    fig.show()
    return df

res = []
for dataframe in dataframes:
    data = my_func(dataframe)
    res.append(data)
res1= pd.concat(res)
```

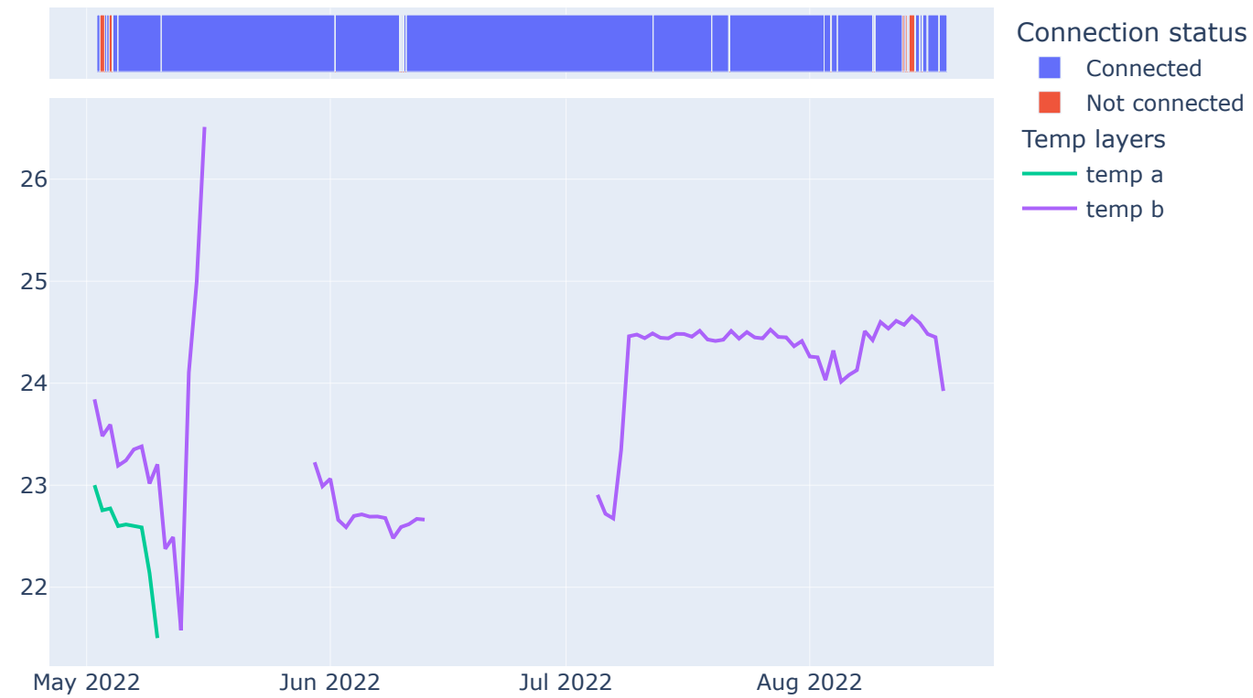



25
24
23
22
21
May 2022
Jun 2022
Jul 2022
Aug 2022

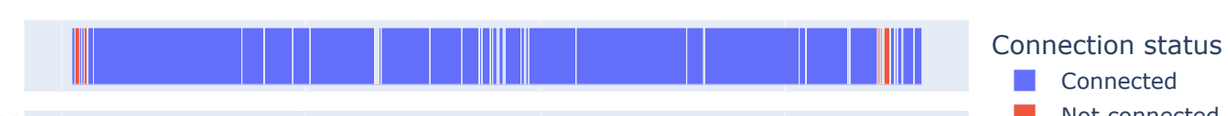
A1_upper

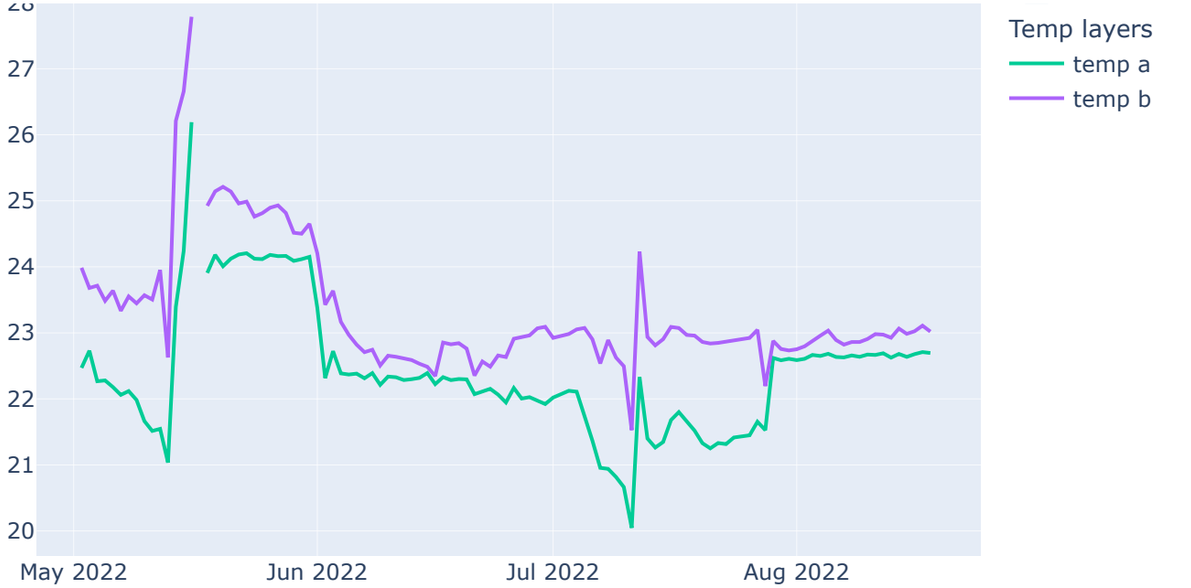

Connection status
Connected
Not connected
Temp layers
temp a
temp b
25
24
23
22

A2_lower

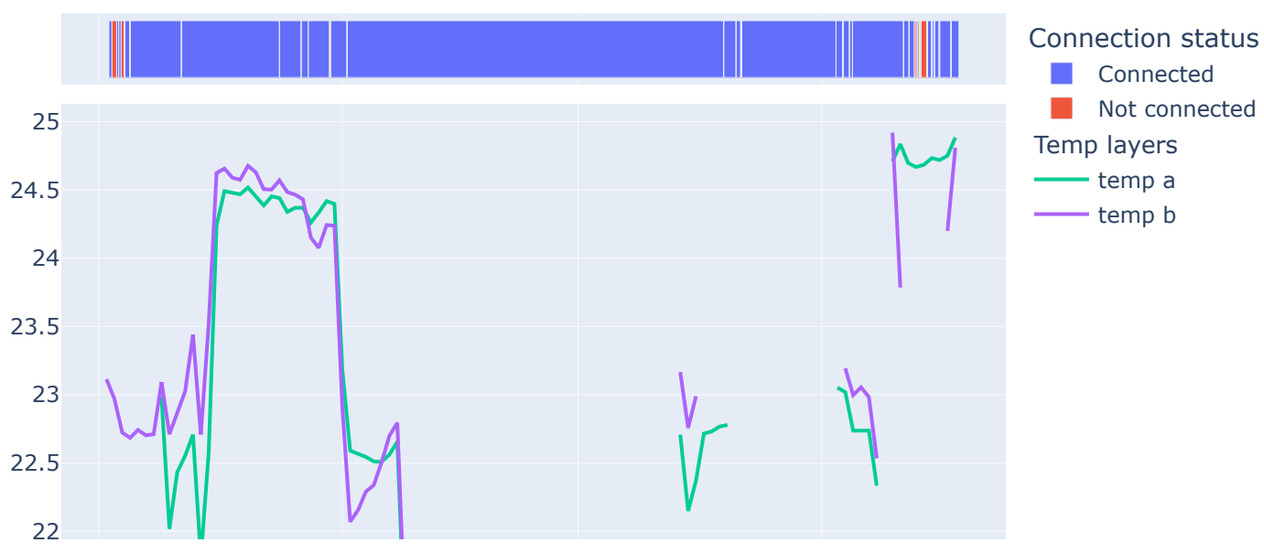


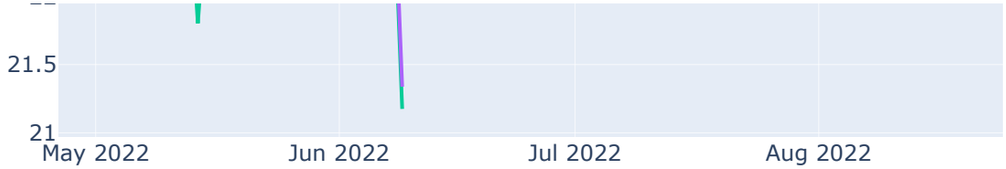
A2_upper



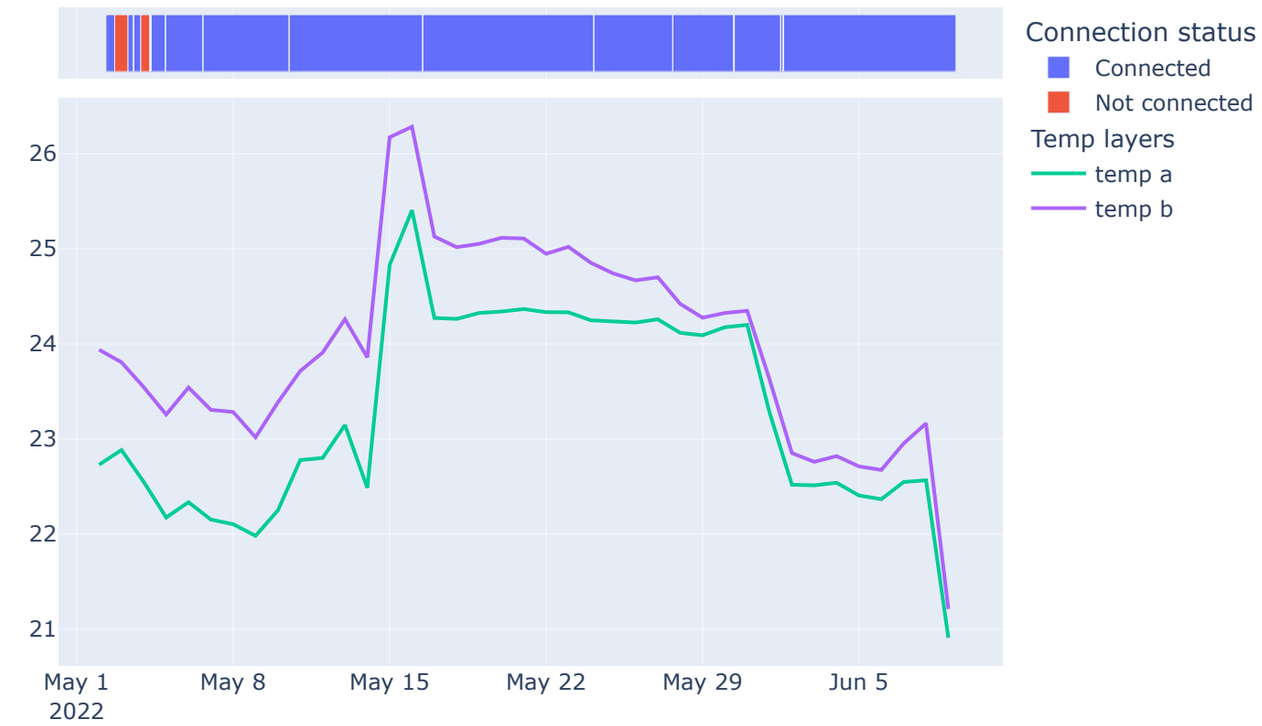

28
27
26
25
24
23
22
21
20
May 2022
Jun 2022
Jul 2022
Aug 2022
Temp layers
temp a
temp b

A3_lower

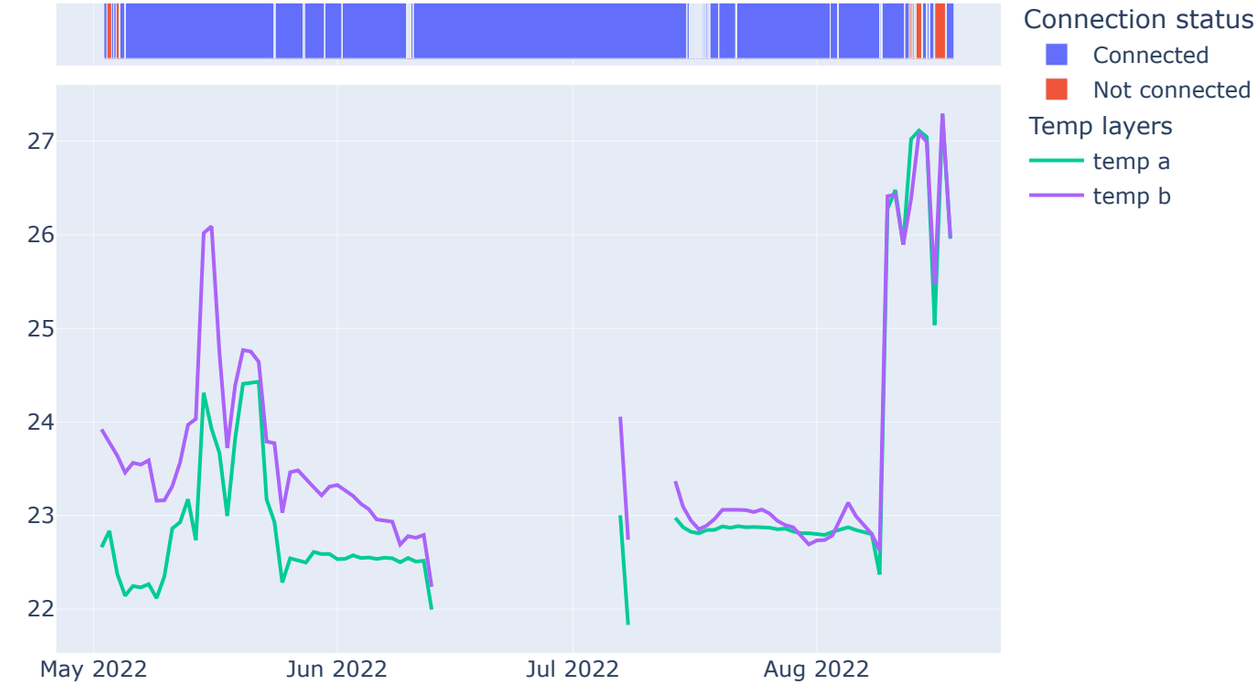

25
24.5
24
23.5
23
22.5
22
Connection status
Connected
Not connected
Temp layers
temp a
temp b



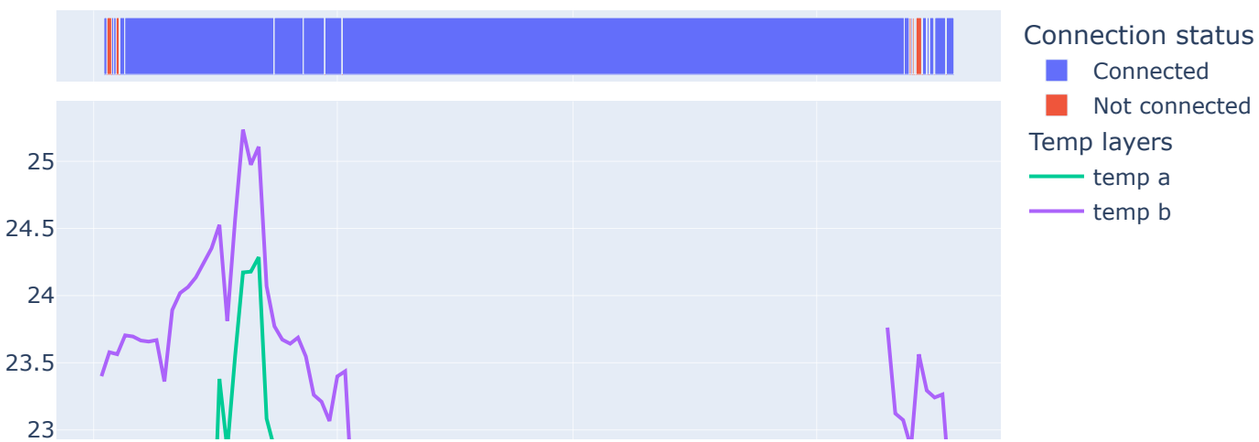
A3_upper

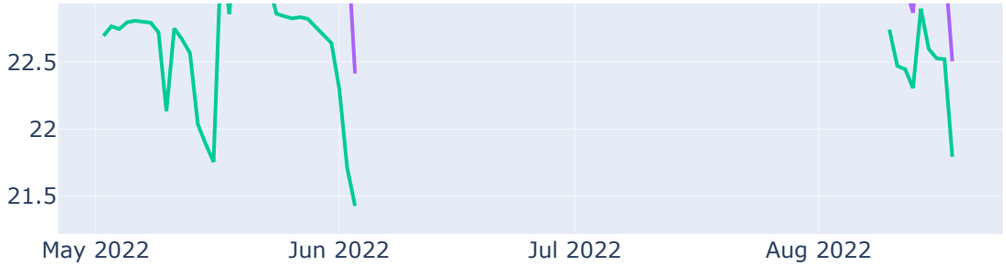


A4_lower

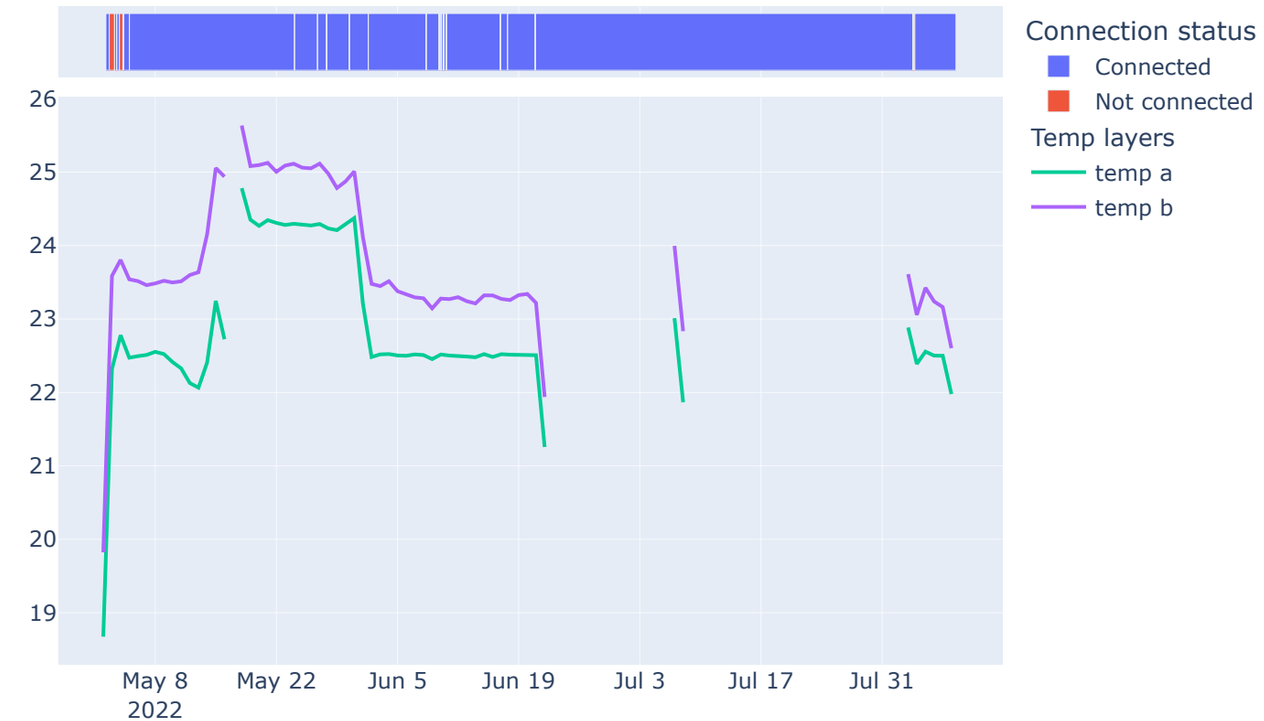

Connection status
Connected
Not connected
Temp layers
temp a
temp b
27
26
25
24
23
22
May 2022
Jun 2022
Jul 2022
Aug 2022

A4_upper

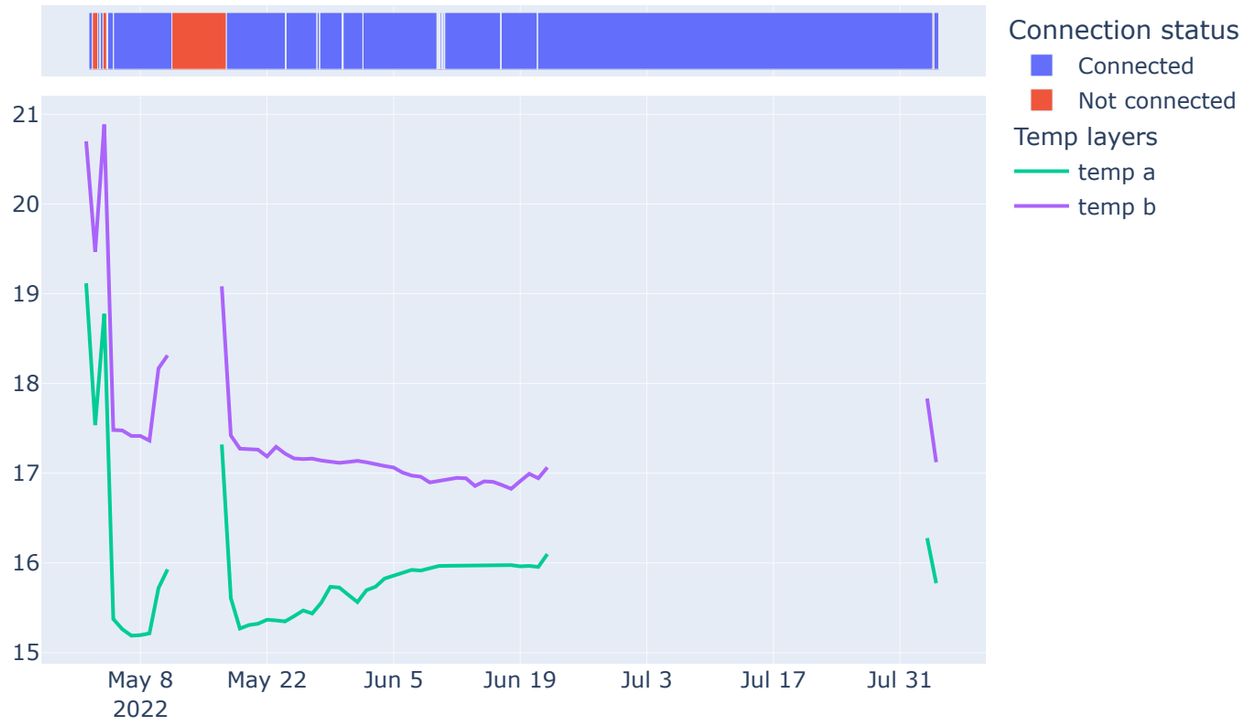

Connection status
Connected
Not connected
Temp layers
temp a
temp b
25
24.5
24
23.5
23


22.5
22
21.5
May 2022
Jun 2022
Jul 2022
Aug 2022

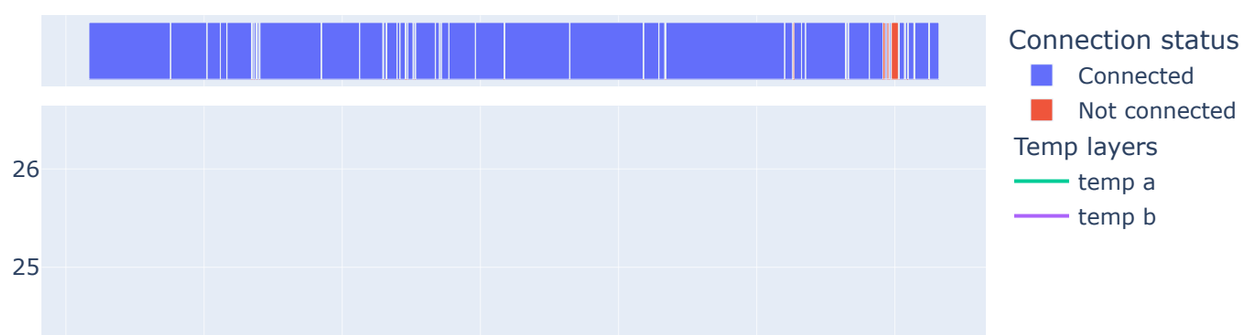
B1_lower


Connection status
Connected
Not connected
Temp layers
temp a
temp b
26
25
24
23
22
21
20
19
May 8
2022
May 22
Jun 5
Jun 19
Jul 3
Jul 17
Jul 31

B1_upper

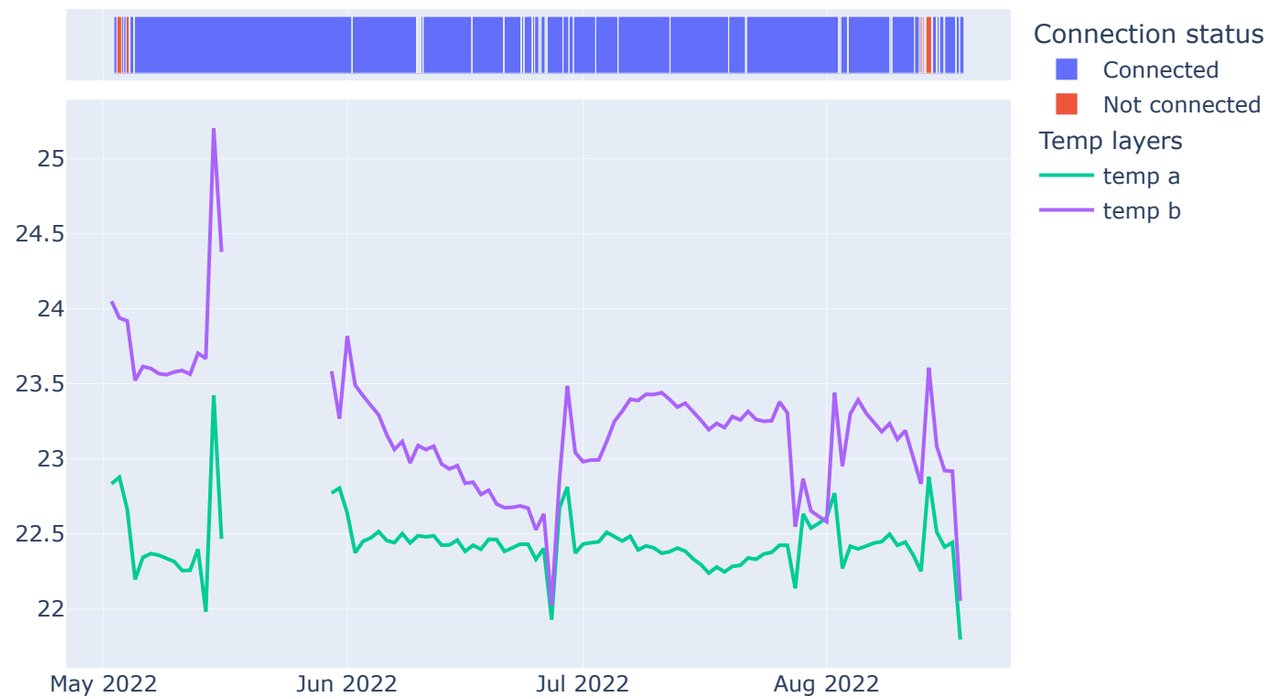


B2_lower

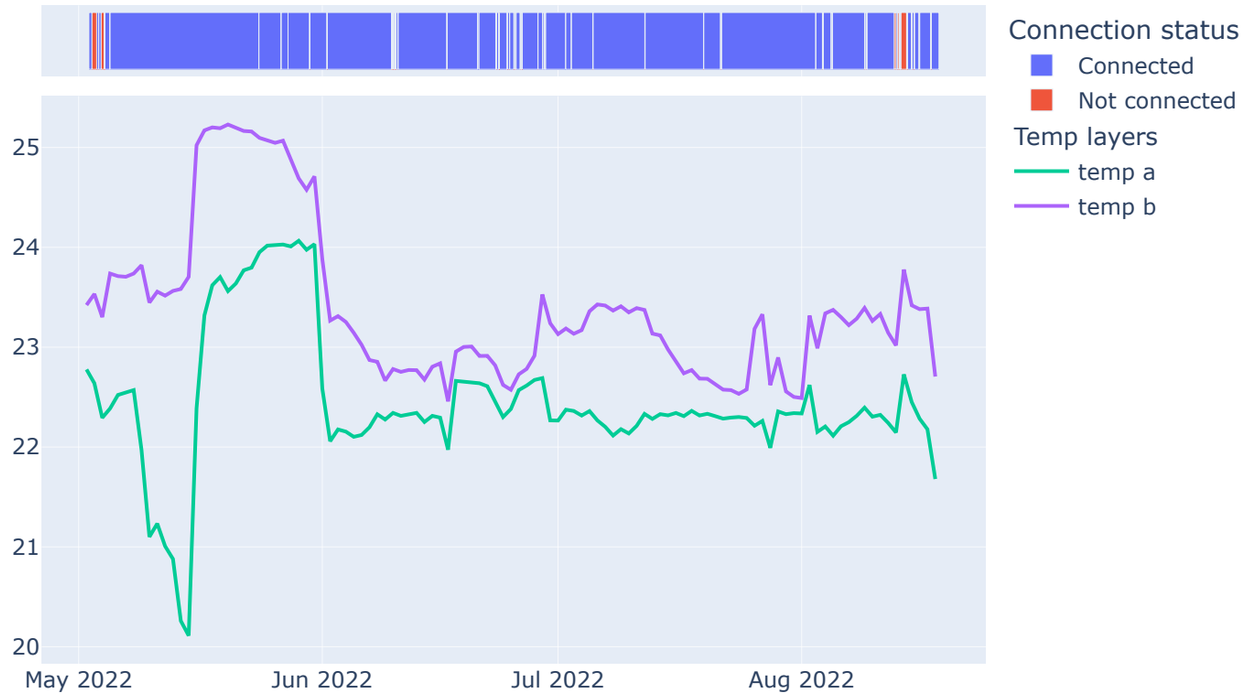



24
23
22
21
May 22
2022
Jun 5
Jun 19
Jul 3
Jul 17
Jul 31
Aug 14

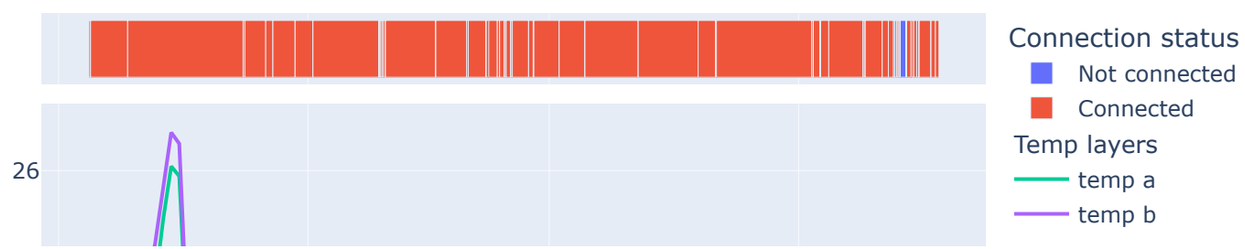
B2_upper


Connection status
Connected
Not connected
Temp layers
temp a
temp b
25
24.5
24
23.5
23
22.5
22
May 2022
Jun 2022
Jul 2022
Aug 2022

B3_upper

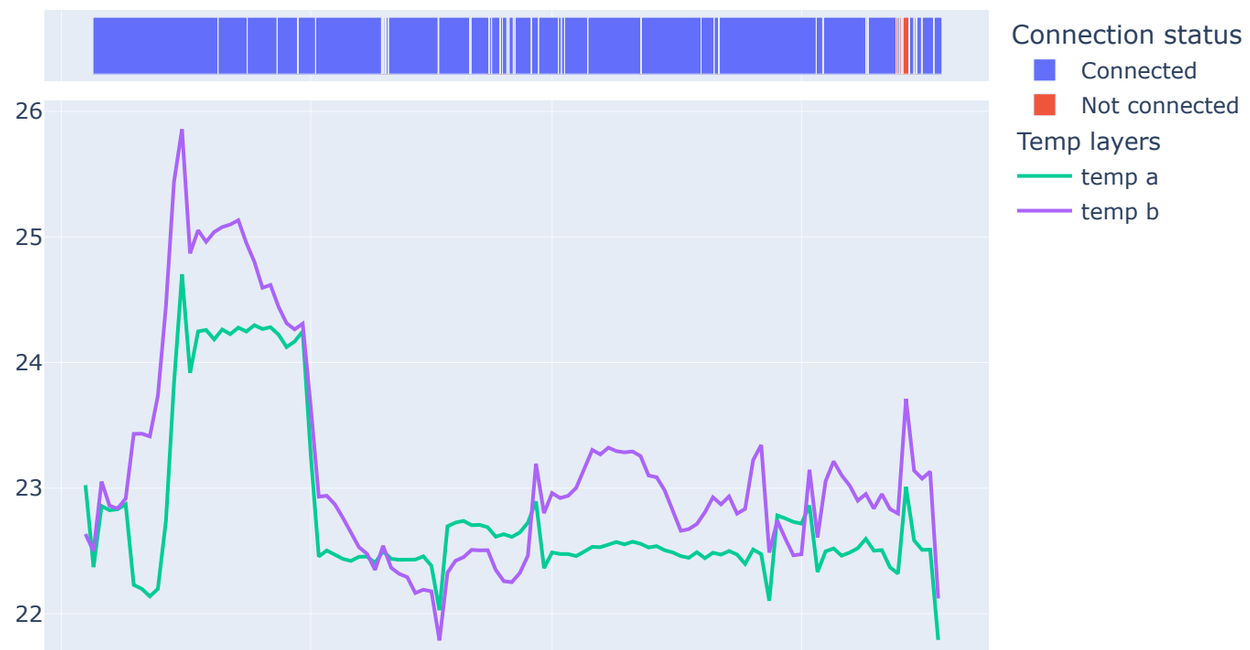


B3_lower

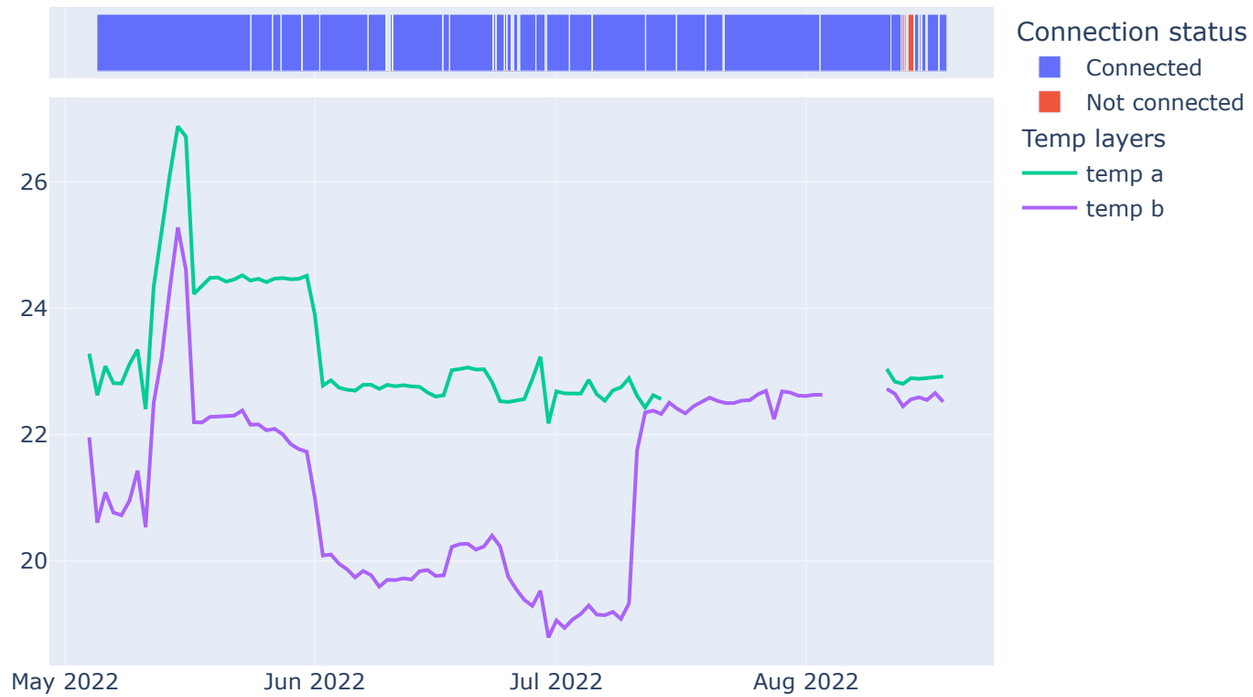



25
24
23
22
21
May 2022
Jun 2022
Jul 2022
Aug 2022

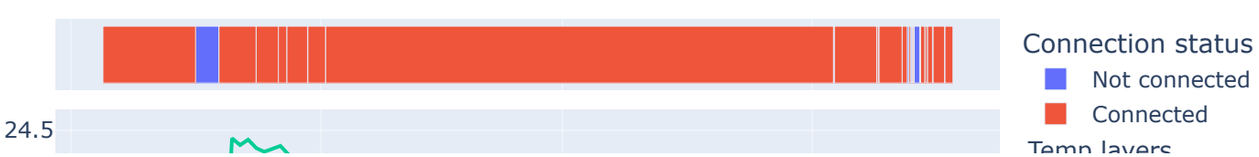
B4_lower

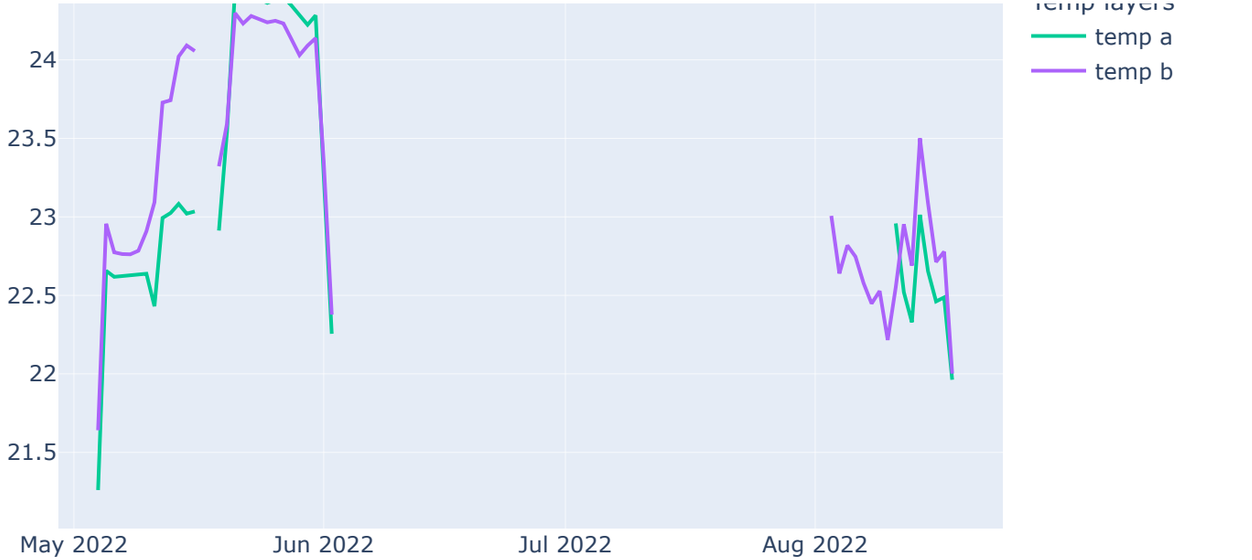

26
25
24
23
22
Connection status
Connected
Not connected
Temp layers
temp a
temp b

B4_upper



C1_lower

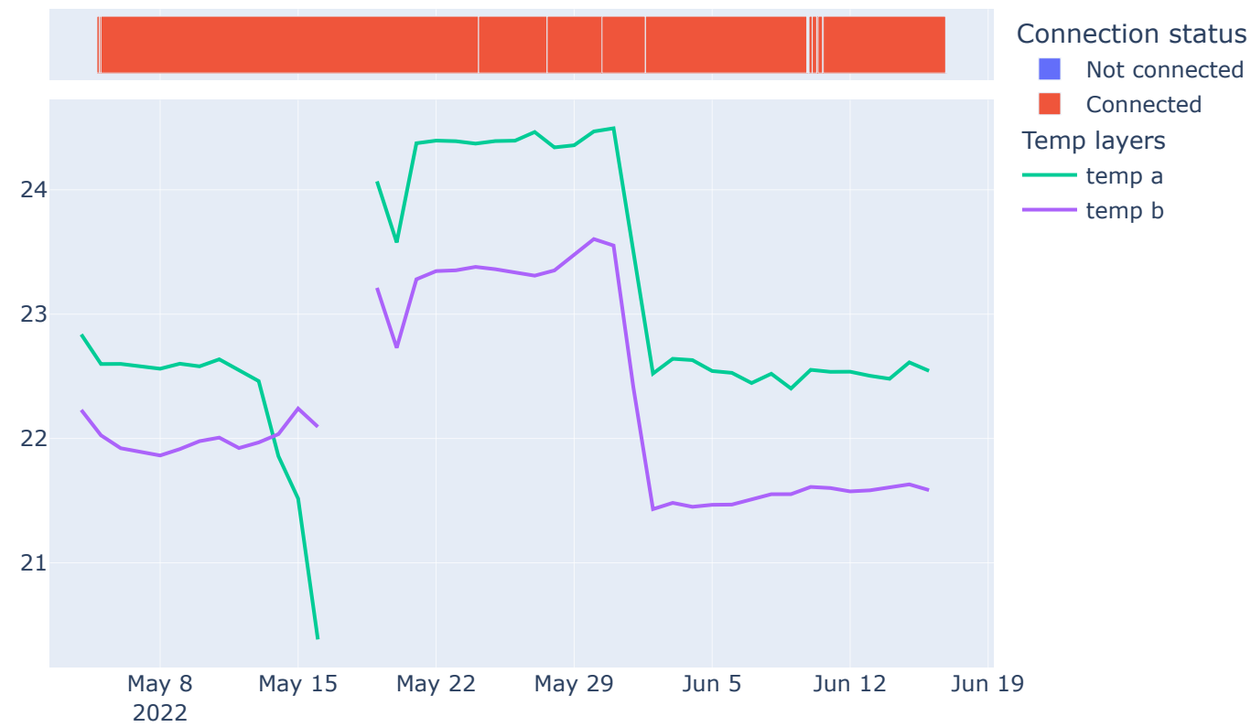



24
23.5
23
22.5
22
21.5
May 2022
Jun 2022
Jul 2022
Aug 2022
temp a
temp b

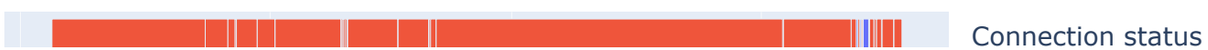
C1_upper

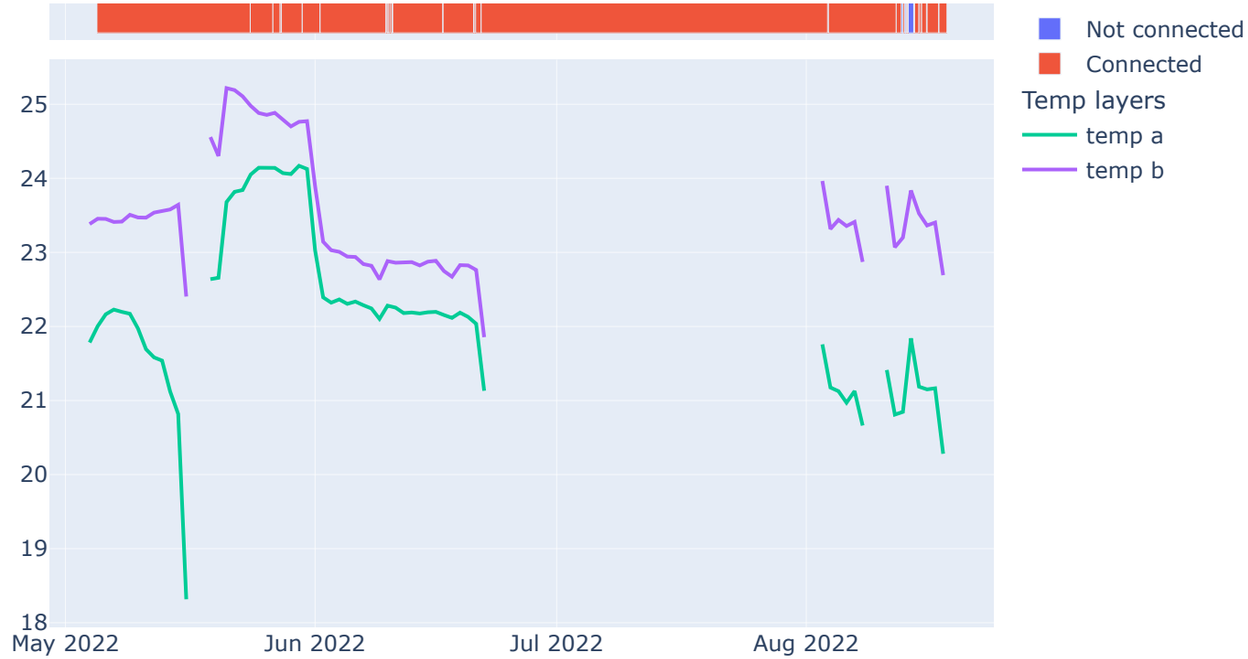

Connection status
Not connected
Connected
Temp layers
temp a
temp b
25.5
25
24.5
24
23.5
23
22.5

C2_lower

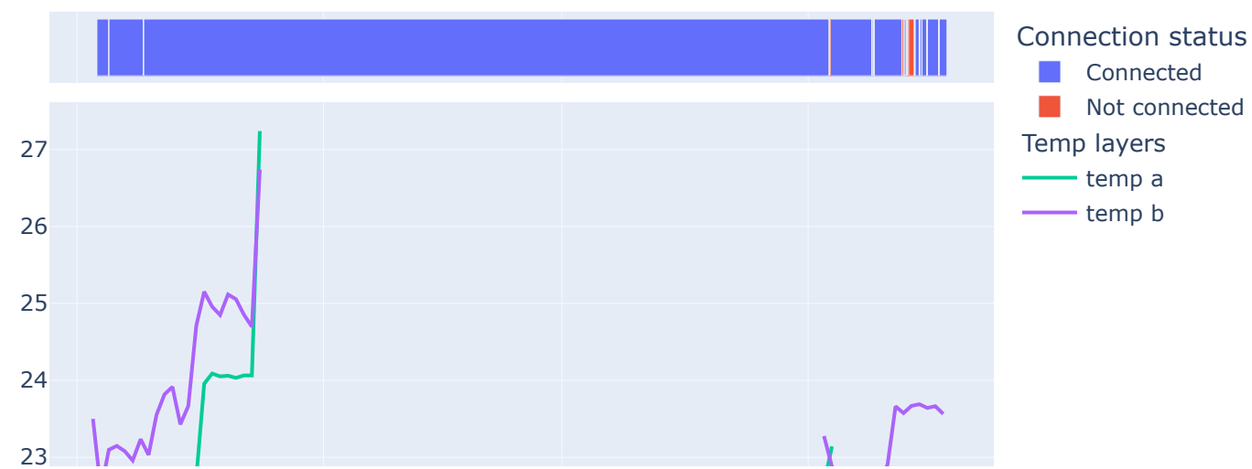


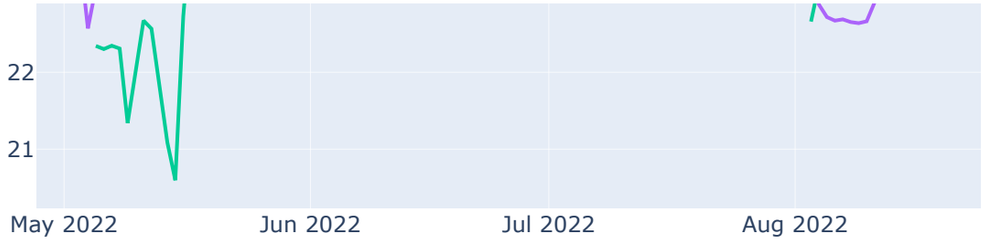
C2_upper



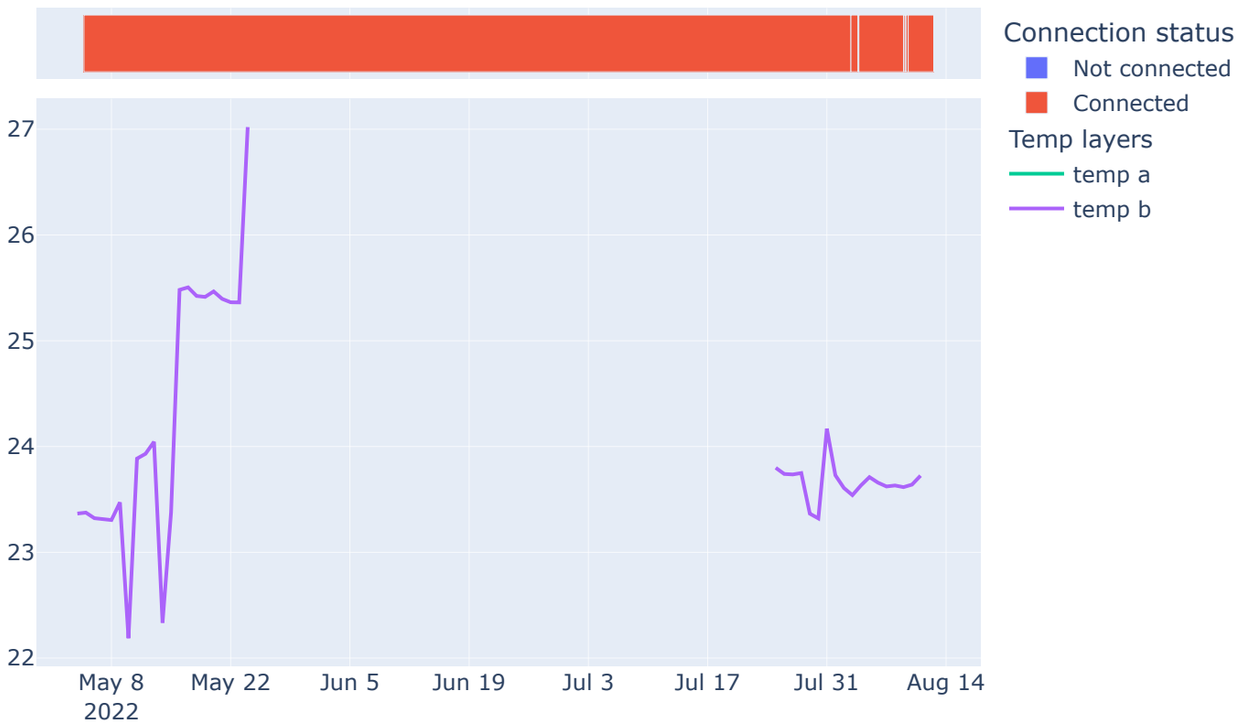

Not connected
Connected
Temp layers
temp a
temp b
25
24
23
22
21
20
19
18
May 2022
Jun 2022
Jul 2022
Aug 2022

C3_lower

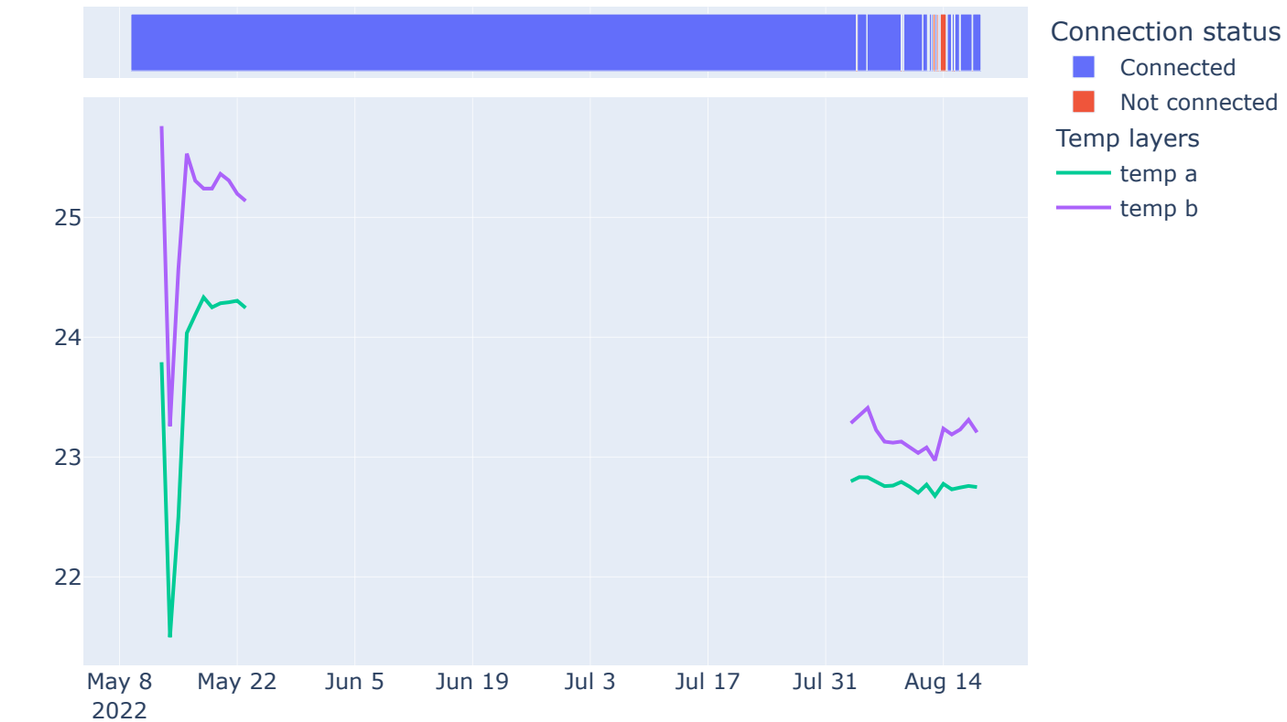

Connection status
Connected
Not connected
Temp layers
temp a
temp b
27
26
25
24
23



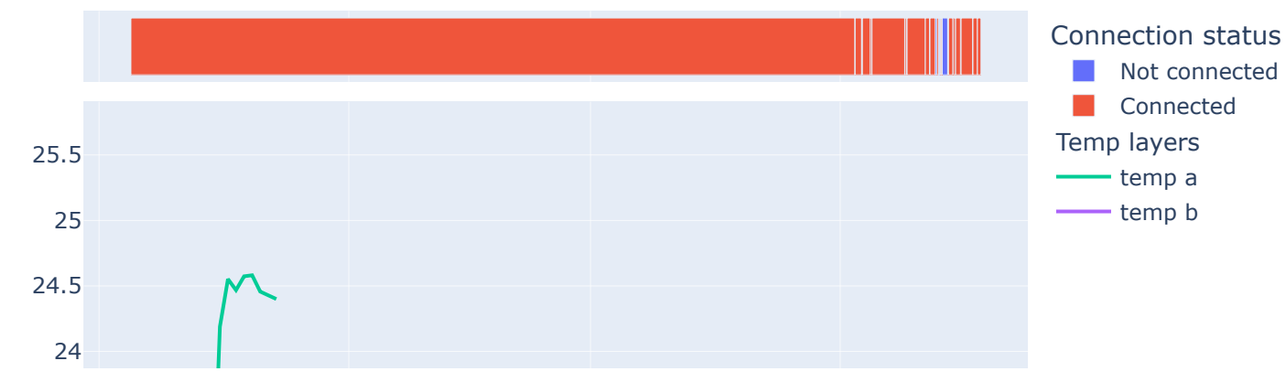
C3_upper

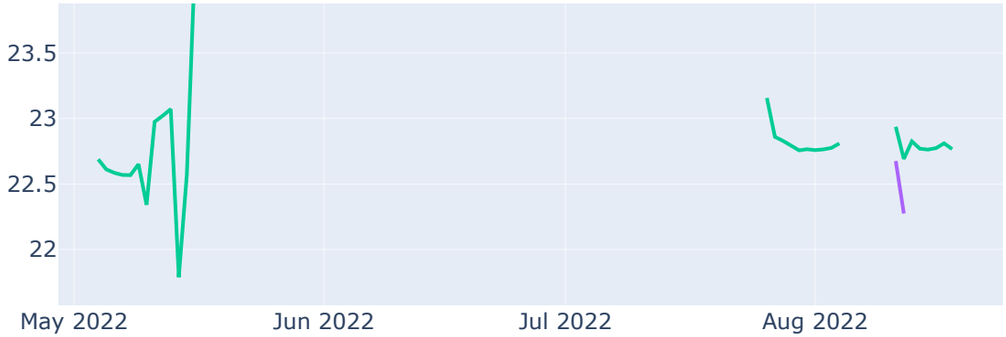


C4_lower

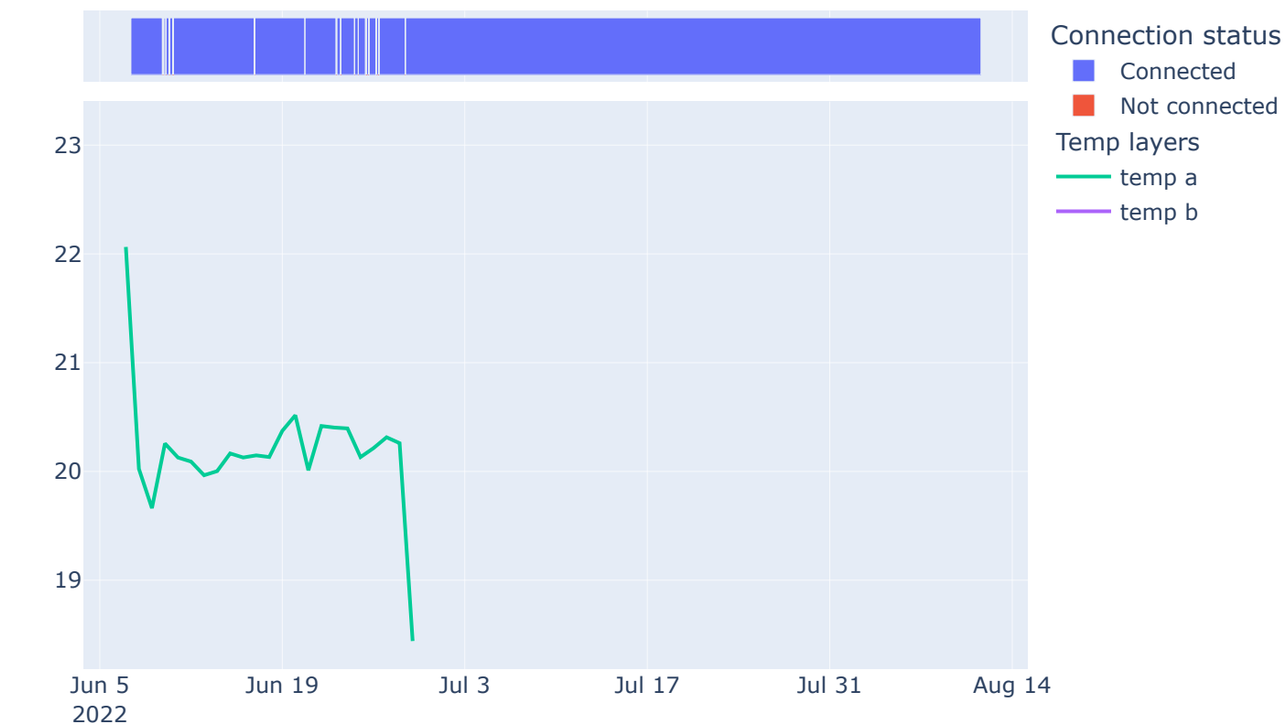

Connection status
Connected
Not connected
Temp layers
temp a
temp b
25
24
23
22
May 8
2022
May 22
Jun 5
Jun 19
Jul 3
Jul 17
Jul 31
Aug 14

C4_upper

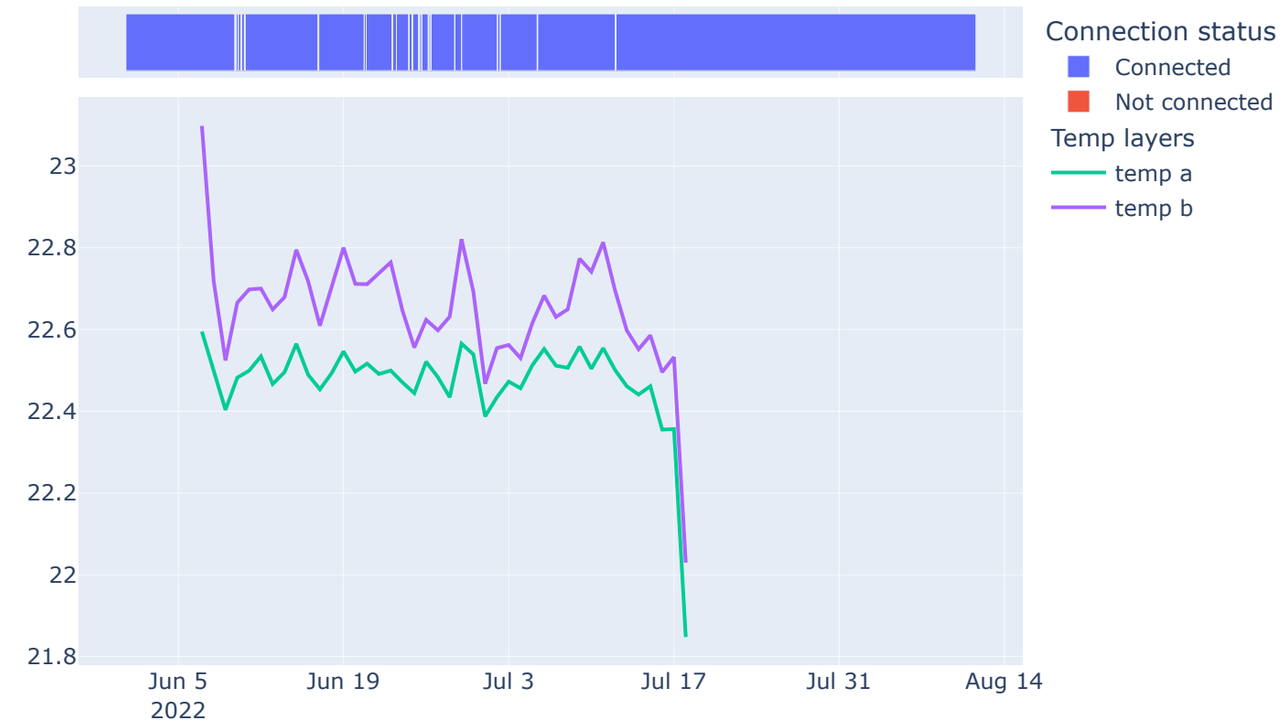

Connection status
Not connected
Connected
Temp layers
temp a
temp b
25.5
25
24.5
24


23.5
23
22.5
22
May 2022
Jun 2022
Jul 2022
Aug 2022

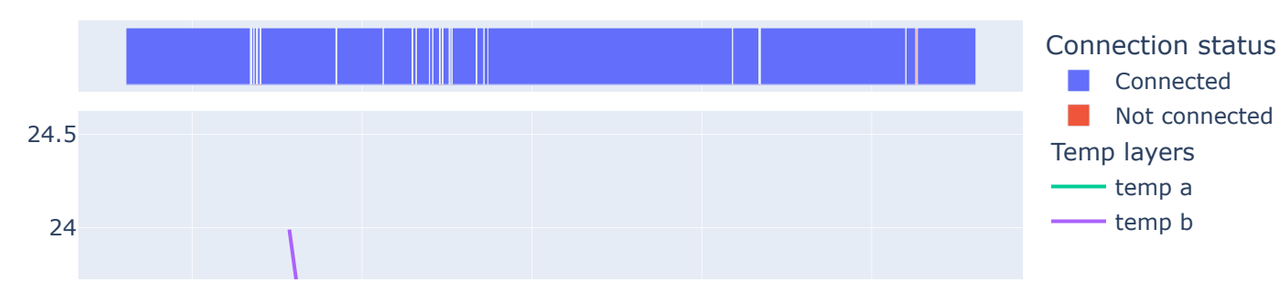
D1_lower


Connection status
Connected
Not connected
Temp layers
temp a
temp b
23
22
21
20
19
Jun 5
2022
Jun 19
Jul 3
Jul 17
Jul 31
Aug 14

D1_upper



D2_lower



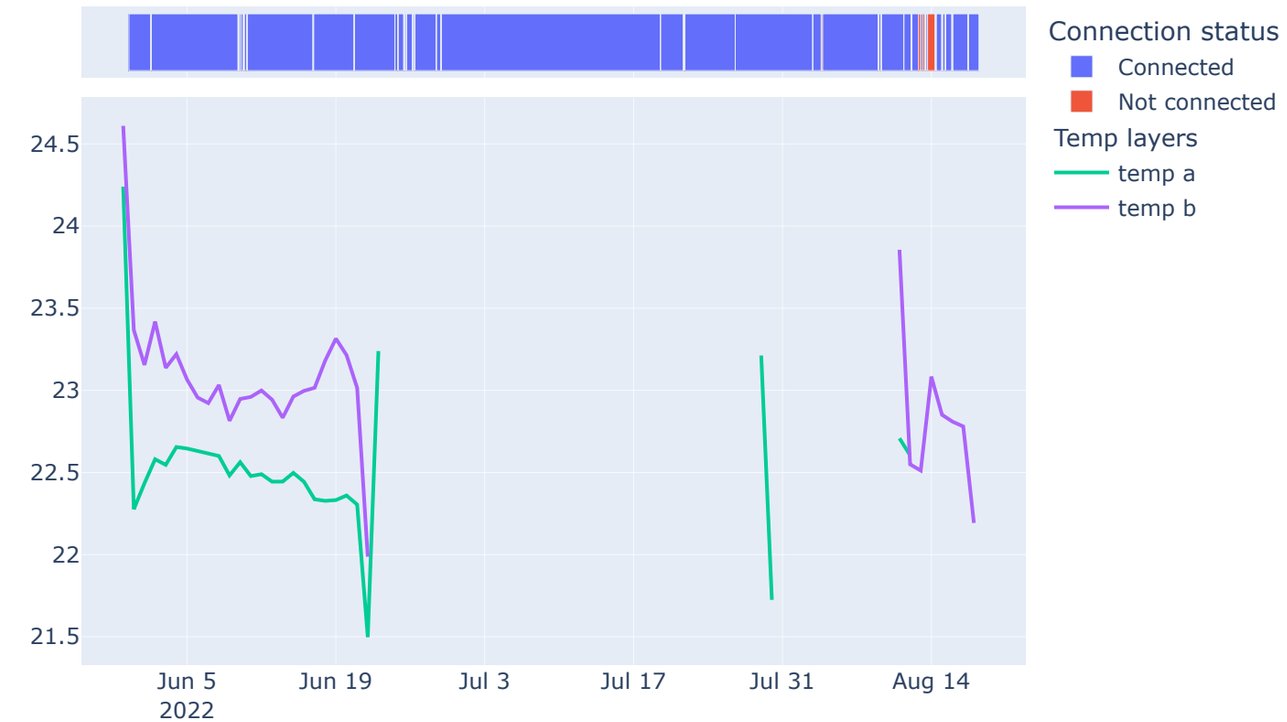

23.5
23
22.5
22
Jun 5
2022
Jun 19
Jul 3
Jul 17
Jul 31

D2_upper

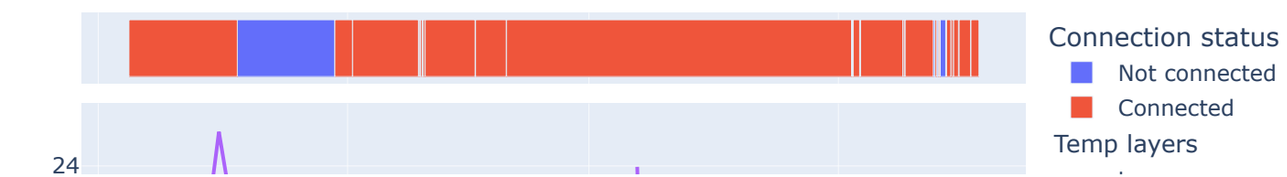

Connection status
Connected
Not connected
Temp layers
temp a
temp b
23.5
23
22.5
22
Jun 5
Jun 19
Jul 3
Jul 17
Jul 31

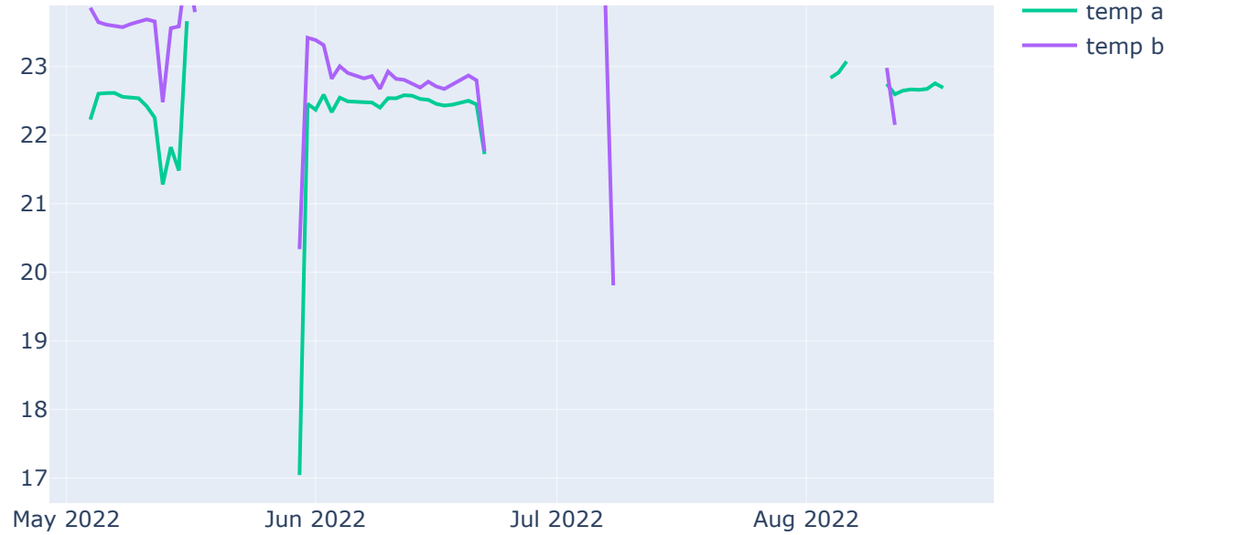
2022

D3_lower

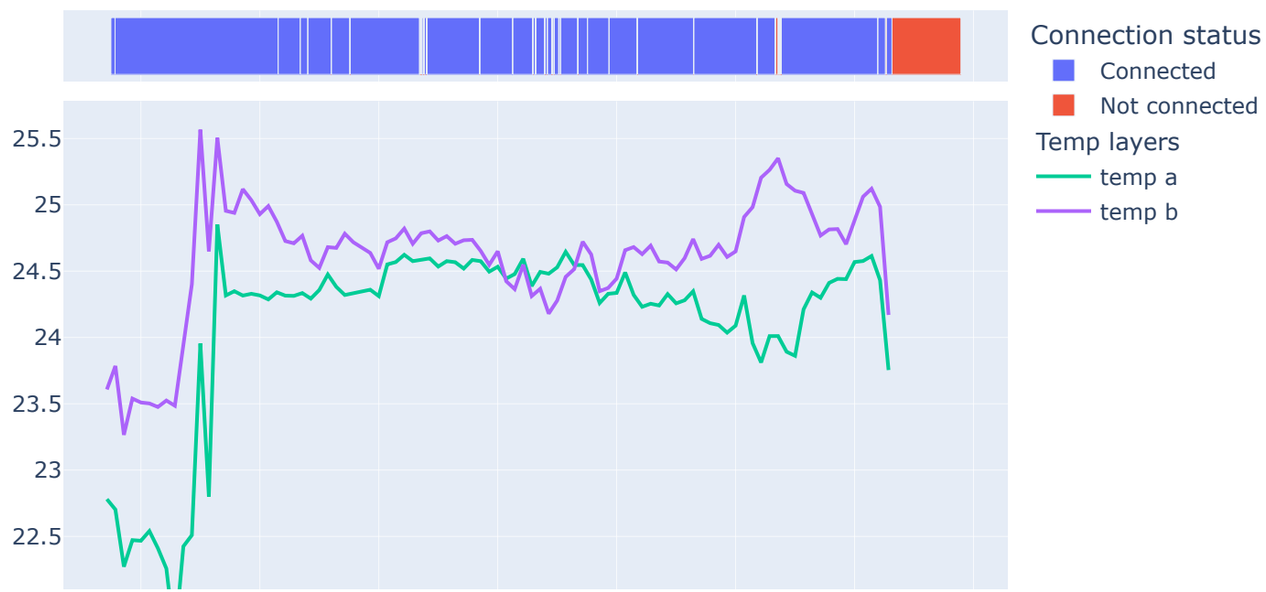


D3_upper



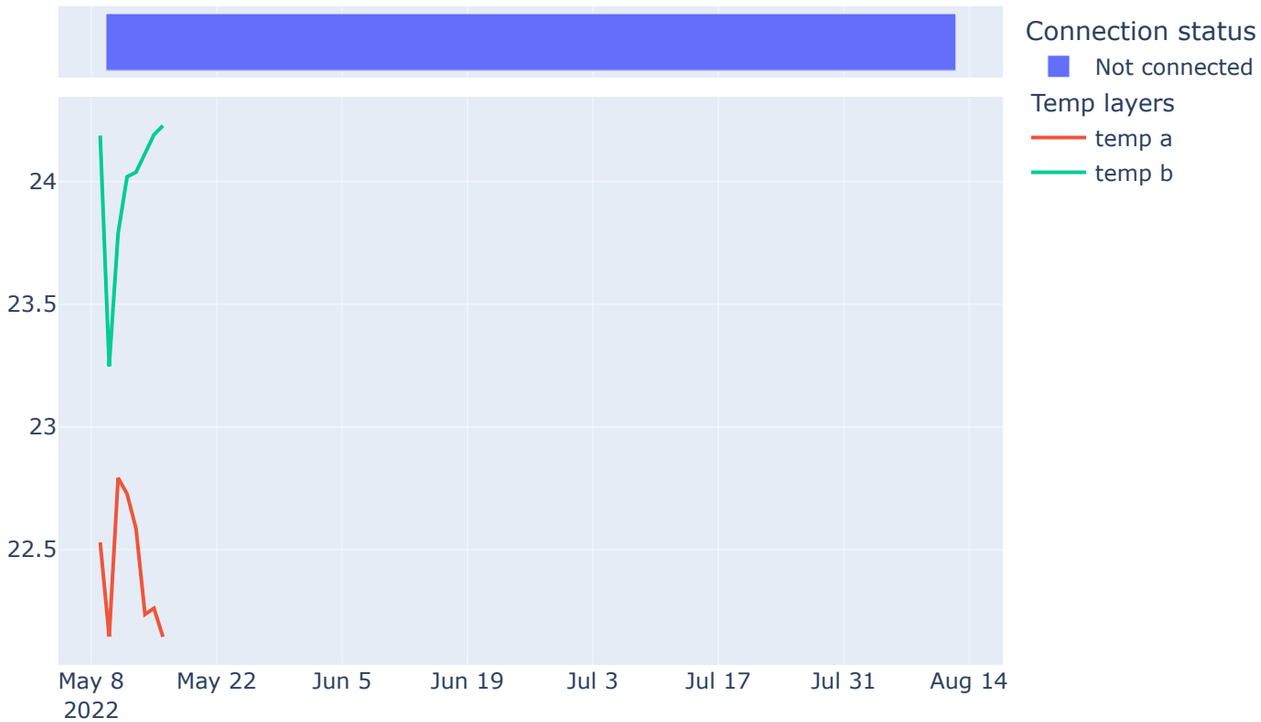

temp a
temp b
23
22
21
20
19
18
17
May 2022
Jun 2022
Jul 2022
Aug 2022

E1

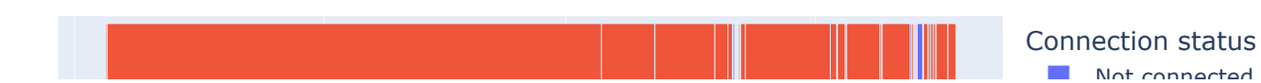

Connection status
Connected
Not connected
Temp layers
temp a
temp b
25.5
25
24.5
24
23.5
23
22.5


22
21.5
May 8
2022
May 22
Jun 5
Jun 19
Jul 3
Jul 17
Jul 31
Aug 14

E2

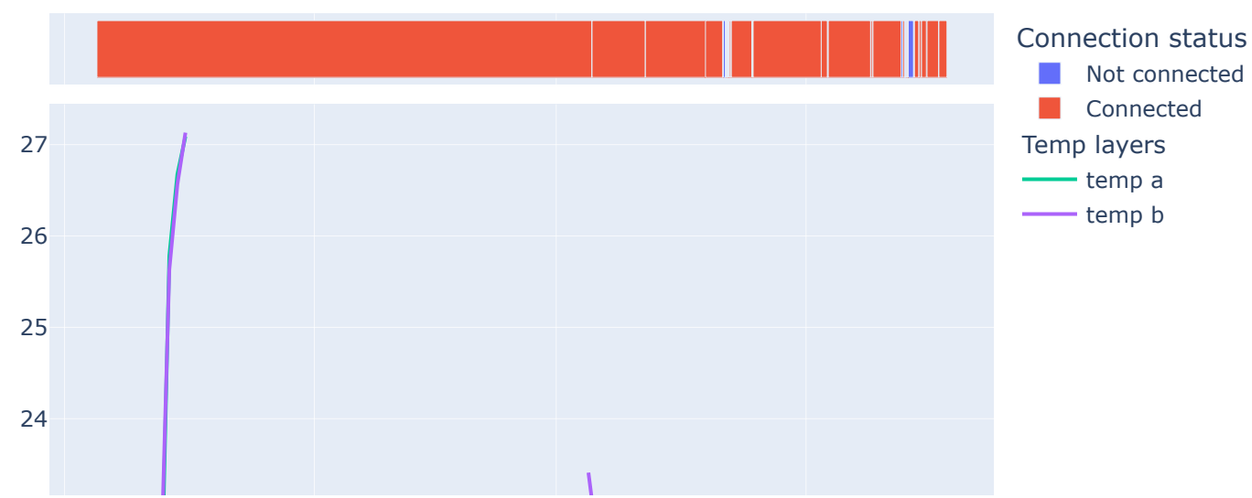

Connection status
Not connected
Temp layers
temp a
temp b
24
23.5
23
22.5
May 8
2022
May 22
Jun 5
Jun 19
Jul 3
Jul 17
Jul 31
Aug 14

E3


Connection status

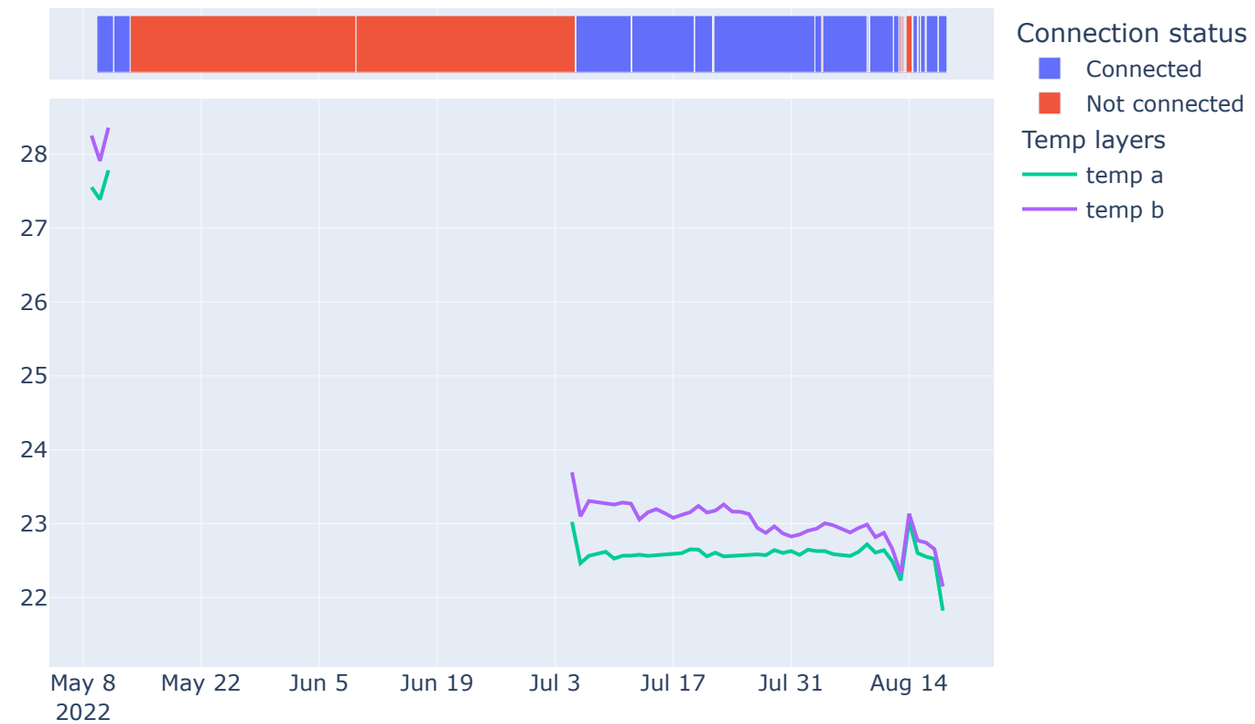

Not connected
Connected
Temp layers
temp a
temp b
28
27
26
25
24
23
22
May 2022
Jun 2022
Jul 2022
Aug 2022

E4

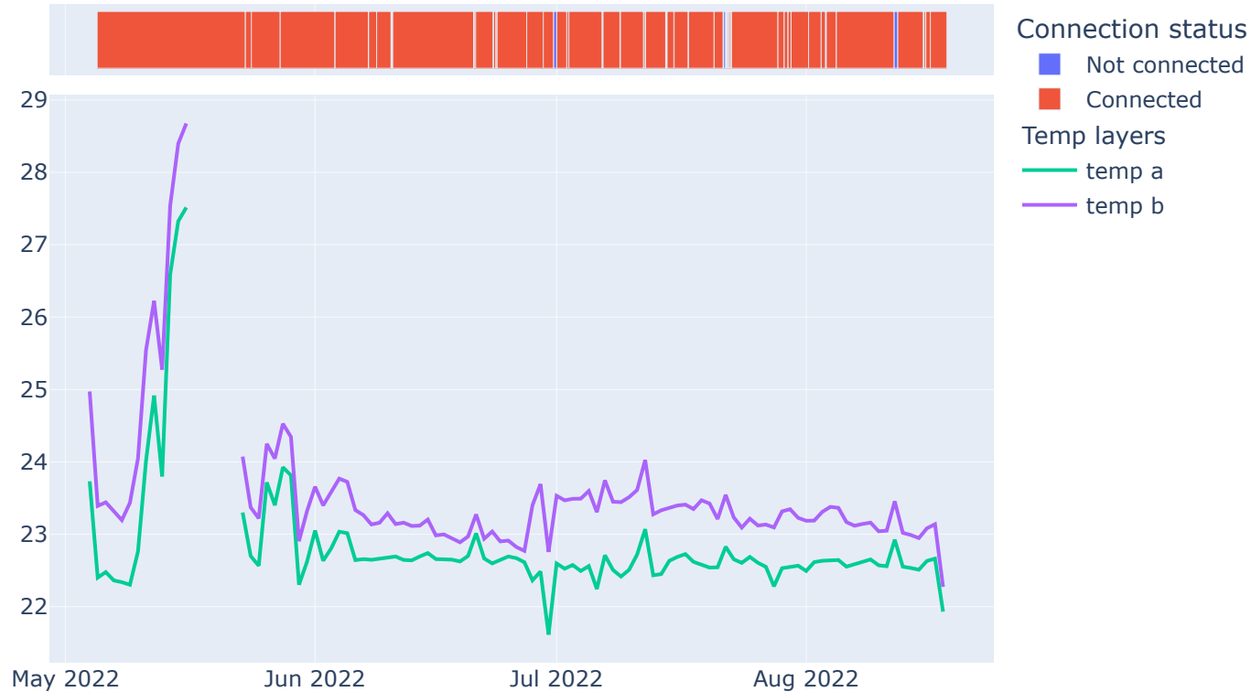

Connection status
Not connected
Connected
Temp layers
temp a
temp b
27
26
25
24


23
22
May 2022
Jun 2022
Jul 2022
Aug 2022

E5

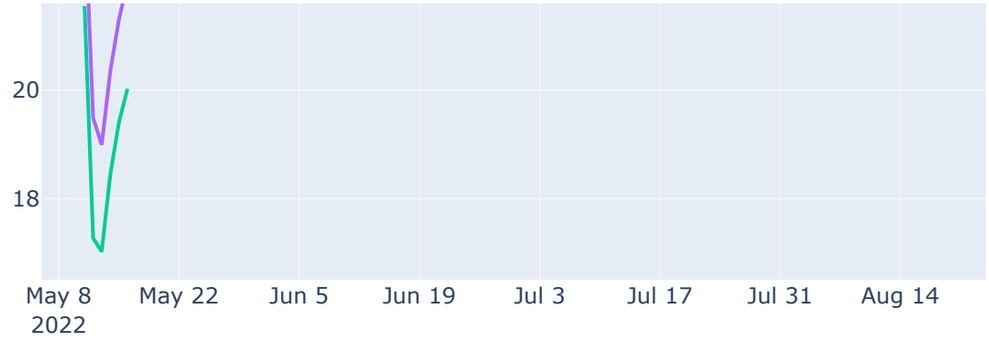

Connection status
Connected
Not connected
Temp layers
temp a
temp b
28
27
26
25
24
23
22
May 8
2022
May 22
Jun 5
Jun 19
Jul 3
Jul 17
Jul 31
Aug 14

E6

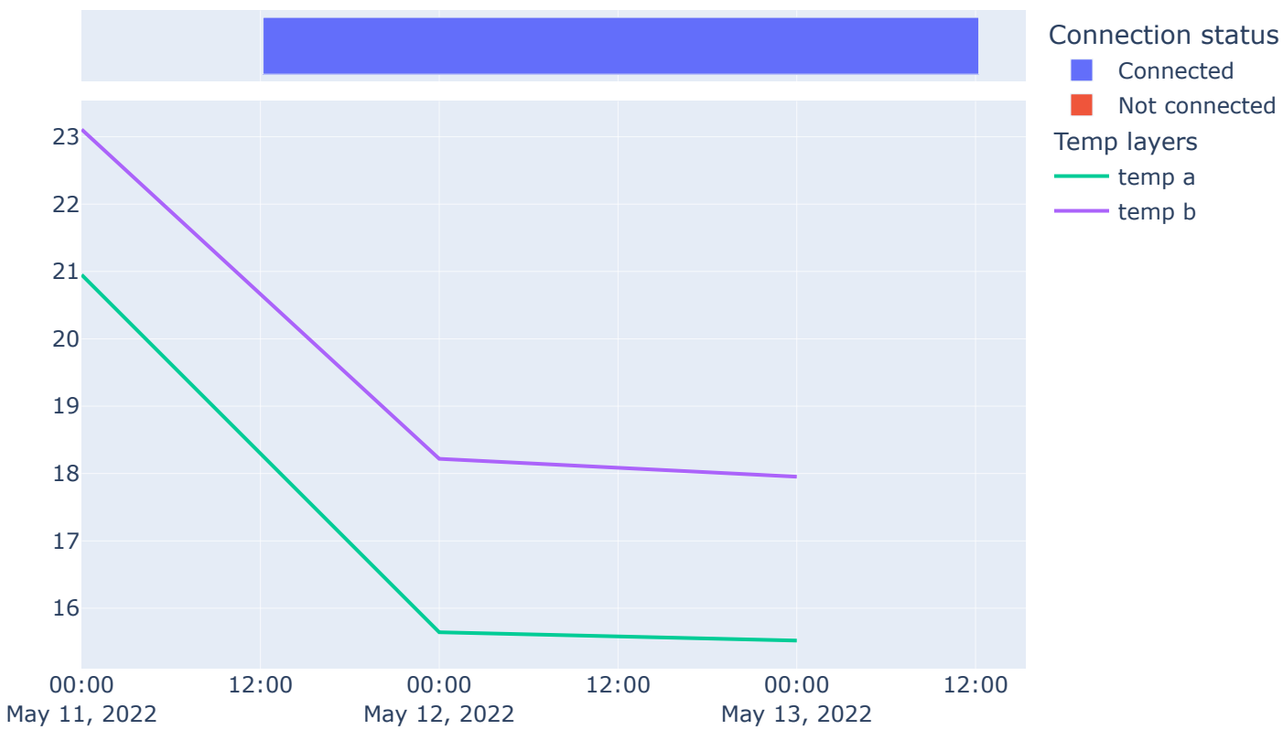

Connection status
Not connected
Connected
Temp layers
temp a
temp b
29
28
27
26
25
24
23
22
May 2022
Jun 2022
Jul 2022
Aug 2022

E7

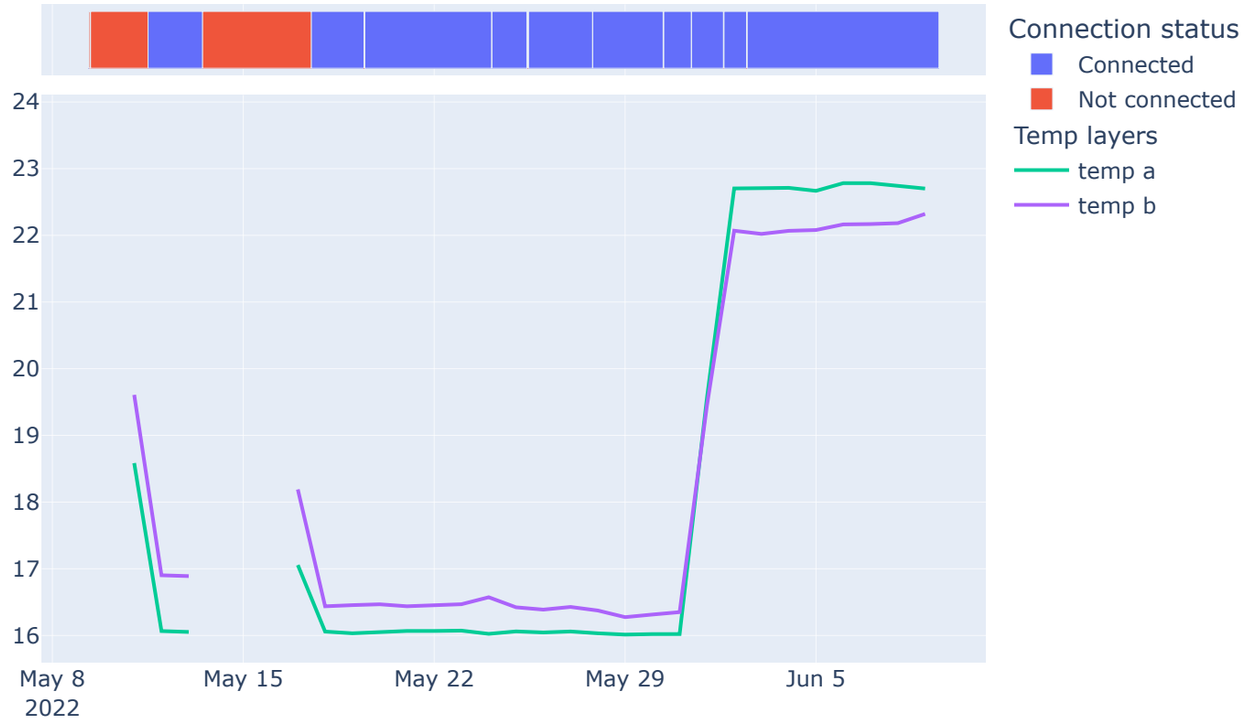

Connection status
Connected
Not connected
Temp layers
temp a
temp b
26
24
22


20
18
May 8
2022
May 22
Jun 5
Jun 19
Jul 3
Jul 17
Jul 31
Aug 14

E8

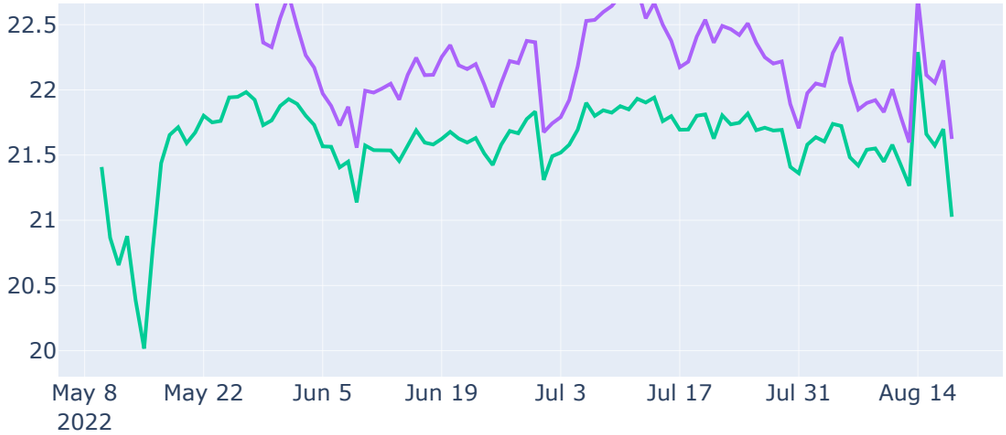

Connection status
Connected
Not connected
Temp layers
temp a
temp b
23
22
21
20
19
18
17
16
00:00
May 11, 2022
12:00
00:00
May 12, 2022
12:00
00:00
May 13, 2022
12:00

E9


Connection status
Connected
Not connected
Temp layers
temp a
temp b
24
23
22
21
20
19
18
17
16
May 8
2022
May 15
May 22
May 29
Jun 5

E10


Connection status
Connected
Not connected
Temp layers
temp a
temp b
24
23.5
23



In [42]:
```
#list of plantcubes where sensors not working
```

In [43]:
```
#A2 lower,A3 upper,A3 lower,B2 lower,B4 upper,C1 lower,C2 lower,C3 lower,C3 upper,C4 upper,D1 lower,D2 lower,D3 lower,D3 upper
```

In [44]:
```
#maximum temp possible - 28, minimum temp possible - 16
```

In [45]:
```
#data after interpolation
res1.head()
```

Out[45]:

timestamp	Unnamed: 0	connected	plantcube	recipe_id	temp_b	temp_a	mode	layers	recipe
2022-05-02 07:16:58.230	0	1	A1_lower	1.000000e+00	NaN	NaN	NaN	default recipe	14
2022-05-02 07:16:58.761	1	1	A1_lower	1.650463e+09	22.270000	22.220000	NaN	[[{'periods': [{'duration': Decimal('86400'), ...	6
2022-05-02 07:16:58.845	2	1	A1_lower	1.000000e+00	22.270029	22.220036	NaN	default recipe	14
2022-05-02 07:16:58.924	3	1	A1_lower	1.650463e+09	22.270057	22.220070	NaN	[[{'periods': [{'duration': Decimal('86400'), ...	6
2022-05-02 07:17:27.215	4	1	A1_lower	1.650463e+09	22.279900	22.232296	NaN	[[{'periods': [{'duration': Decimal('86400'), ...	6

In [46]:
```
res1 = res1.reset_index()
```

In [47]:
```
res1 = res1.drop('Unnamed: 0', axis=1)
```

In [48]:
```
res1.head()
```

Out[48]:

	timestamp	connected	plantcube	recipe_id	temp_b	temp_a	mode	layers	recipe
0	2022-05-02 07:16:58.230	1	A1_lower	1.000000e+00	NaN	NaN	NaN	default recipe	14
1	2022-05-02 07:16:58.761	1	A1_lower	1.650463e+09	22.270000	22.220000	NaN	[[{'periods': [{'duration': Decimal('86400'), ...	6
2	2022-05-02 07:16:58.845	1	A1_lower	1.000000e+00	22.270029	22.220036	NaN	default recipe	14
3	2022-05-02 07:16:58.924	1	A1_lower	1.650463e+09	22.270057	22.220070	NaN	[[{'periods': [{'duration': Decimal('86400'), ...	6
4	2022-05-02 07:17:27.215	1	A1_lower	1.650463e+09	22.279900	22.232296	NaN	[[{'periods': [{'duration': Decimal('86400'), ...	6

In [49]:
```
a5 = res1[(res1.temp_b > 29)& (res1.temp_a > 29)]
```

In [50]:
```
a5
```

Out[50]:

timestamp	connected	plantcube	recipe_id	temp_b	temp_a	mode	layers	recipe

In [51]:
```
a5['plantcube'].unique()
```

Out[51]:
```
array([], dtype=object)
```

In [52]:
```
a6 = res1[(res1.temp_b < 15)& (res1.temp_a < 15)]
```

In [53]:
```
a6['plantcube'].unique()
#d3 upper
```

Out[53]:
```
array([], dtype=object)
```

In [54]:
```
##
```

In [55]:
```
#automation

#minimum and maximum temperature for recipe1 to recipe13
#Night target
mintemp = [23,23,16,21,19,21,19,21,23,23,16,23,23]
#day target
maxtemp = [25,25,23,23,23,23,23,23,25,28,23,23,23]
rha = []
rla = []

for i in range(0,13):
    j = i+1
    print("Recipe ",j)
    print('\n')
    recipe = res1[res1.recipe == j]

    print("Max temperature:",maxtemp[i])
    rh= recipe[(recipe.temp_a > (maxtemp[i]+1)) & (recipe.temp_b > (maxtemp[i]+1))]
    if rh.empty:
        print("")
    else:
        print("Greater than max:")
        print(rh.plantcube.unique())
    rh['dev_temp_a'] = abs(rh['temp_a'] - maxtemp[i])
    rh['dev_temp_b'] = abs(rh['temp_b'] - maxtemp[i])
    #appending the greater than max values for all recipes
    rha.append(rh)
    g1 = rh.groupby('plantcube')['plantcube','dev_temp_a','dev_temp_b'].mean(numeric_only = True)
    if g1.empty:
        print("")
    else:
        print("Deviation:")
        #print(tabulate(g1,headers='keys', tablefmt='psql'))
        display(g1)
    print('\n')

    print("Min temperature:",mintemp[i])
    rl= recipe[(recipe.temp_a < (mintemp[i]-1)) & (recipe.temp_b < (mintemp[i]-1))]
    if rl.empty:
        print("")
    else:
        print("Lesser than min:")
```

```
        print(rl.plantcube.unique())
    rl['dev_temp_a'] = abs(rl['temp_a'] - mintemp[i])
    rl['dev_temp_b'] = abs(rl['temp_b'] - mintemp[i])
    #appending the greater than max values for all recipes
    rla.append(rl)
    g2 = rl.groupby('plantcube')['plantcube','dev_temp_a','dev_temp_b'].mean(numeric_only = True)
    if g2.empty:
        print("")
    else:
        print("Deviation:")
        #print(tabulate(g2,headers='keys', tablefmt='psql'))
        display(g2)
    print('\n')
    print('**************************************************')
final_rh = pd.concat(rha, ignore_index=True)
final_rl = pd.concat(rla, ignore_index=True)
```

```
Recipe  1


Max temperature: 25
Greater than max:
['A3_upper' 'B1_lower' 'B3_lower' 'B4_upper' 'C3_lower']
Deviation:
```

plantcube	dev_temp_a	dev_temp_b
A3_upper	1.010000	1.774872
B1_lower	1.374160	1.416574
B3_lower	1.170989	1.279066
B4_upper	2.535888	1.217575
C3_lower	2.240000	1.740000

```
Min temperature: 23
Lesser than min:
['A3_lower' 'A3_upper' 'B4_lower' 'C1_lower' 'C2_lower' 'E1']
Deviation:
```

plantcube	dev_temp_a	dev_temp_b
A3_lower	1.071891	1.080000
A3_upper	1.268928	1.102542
B4_lower	1.054763	1.157963
C1_lower	1.043781	1.120000
C2_lower	1.308458	1.213797
E1	1.209945	1.038332

```
************************************************
Recipe  2


Max temperature: 25
Greater than max:
['A1_upper' 'A2_upper' 'A3_lower' 'A3_upper' 'B3_upper' 'B3_lower'
 'B4_lower' 'B4_upper' 'C3_lower' 'C4_lower' 'E1']
Deviation:
```

plantcube	dev_temp_a	dev_temp_b
A1_upper	2.561455	2.968932
A2_upper	1.305375	1.881106
A3_lower	1.103653	1.755439
A3_upper	2.537824	3.023994
B3_lower	1.634612	1.683045
B3_upper	1.406611	2.528543
B4_lower	1.685958	2.287897
B4_upper	4.278522	2.140027
C3_lower	1.453968	2.461896

plantcube	dev_temp_a	dev_temp_b
C4_lower	1.678923	2.500299
E1	1.643419	1.820521

```
Min temperature: 23
```

```
**************************************************
Recipe  3
```

```
Max temperature: 23
```

```
Min temperature: 16
```

```
**************************************************
Recipe  4
```

```
Max temperature: 23
```

```
Min temperature: 21
```

```
**************************************************
```

```
Recipe  5


Max temperature: 23




Min temperature: 19



**************************************************
Recipe  6


Max temperature: 23
Greater than max:
['A1_lower' 'A1_upper' 'A3_lower' 'A3_upper' 'A4_lower' 'B1_lower'
 'B2_upper' 'B3_upper' 'B3_lower' 'B4_lower' 'B4_upper' 'C1_lower'
 'C1_upper' 'C2_lower' 'C2_upper' 'C3_lower' 'C4_lower' 'D1_upper'
 'D2_lower' 'D2_upper' 'D3_lower' 'D3_upper' 'E1' 'E3' 'E4' 'E5' 'E6' 'E7'
 'E9']
Deviation:
```

plantcube	dev_temp_a	dev_temp_b
A1_lower	1.896712	4.029589
A1_upper	2.038934	2.067575
A3_lower	1.652223	2.436690
A3_upper	2.921914	4.062982
A4_lower	2.287679	3.839813
B1_lower	1.844784	3.202414
B2_upper	1.619824	3.021401
B3_lower	2.605329	3.002293
B3_upper	1.867534	2.652392

plantcube	dev_temp_a	dev_temp_b
B4_lower	2.077635	3.675402
B4_upper	4.022508	2.244126
C1_lower	1.442408	1.722228
C1_upper	2.525411	3.594521
C2_lower	2.577867	1.867053
C2_upper	1.616631	2.406239
C3_lower	3.771107	4.670325
C4_lower	2.066252	3.893387
D1_upper	1.188317	1.254832
D2_lower	1.740768	2.249390
D2_upper	1.435727	1.426184
D3_lower	2.600973	2.982633
D3_upper	1.376462	3.495647
E1	2.000791	3.346017
E3	2.275010	3.901704
E4	2.559452	2.559789
E5	3.735318	4.444904
E6	2.281961	3.255328
E7	1.419667	1.920239
E9	1.445611	1.981149

```
Min temperature: 21
Lesser than min:
['A2_upper' 'A3_upper' 'B1_lower' 'B2_upper' 'B3_upper' 'B4_upper'
 'C1_lower' 'C2_lower' 'C2_upper' 'D1_upper' 'D2_upper' 'D3_upper' 'E4'
 'E9']
Deviation:
```

plantcube	dev_temp_a	dev_temp_b
A2_upper	1.279577	1.072466
A3_upper	1.249782	1.048095
B1_lower	3.605712	1.371740
B2_upper	1.066318	1.053442
B3_upper	1.925506	1.155323
B4_upper	1.111857	1.138339
C1_lower	1.842127	1.156780
C2_lower	1.190610	1.192717
C2_upper	1.594160	1.095108
D1_upper	1.161529	1.178892
D2_upper	1.117982	1.096472
D3_upper	1.708607	1.239085
E4	1.081994	1.335529
E9	2.631065	2.869274

```
**************************************************
Recipe  7

Max temperature: 23



Min temperature: 19
```

```
**************************************************
Recipe  8


Max temperature: 23
Greater than max:
['E7']
Deviation:
```

plantcube	dev_temp_a	dev_temp_b
E7	3.657824	4.258093

```
Min temperature: 21



**************************************************
Recipe  9

Max temperature: 25



Min temperature: 23



**************************************************
Recipe  10
```

```
Max temperature: 28
```

```
Min temperature: 23
```

```
************************************************
Recipe  11
```

```
Max temperature: 23
Greater than max:
['B1_upper' 'E7' 'E8' 'E9']
Deviation:
```

plantcube	dev_temp_a	dev_temp_b
B1_upper	1.102652	1.153916
E7	1.909653	2.601208
E8	1.686717	2.643841
E9	2.228933	2.033378

```
Min temperature: 16
```

```
************************************************
Recipe  12
```

```
Max temperature: 23
Greater than max:
['A2_lower' 'A3_lower' 'A3_upper' 'A4_lower' 'A4_upper' 'B2_upper'
```

```
 'B3_upper' 'B4_upper' 'C1_lower' 'C1_upper' 'C2_lower' 'C2_upper'
 'C4_upper' 'D3_upper']
Deviation:
```

plantcube	dev_temp_a	dev_temp_b
A2_lower	1.089946	1.479516
A3_lower	1.189649	1.494685
A3_upper	1.529044	2.285800
A4_lower	1.281292	1.953084
A4_upper	2.789045	3.420649
B2_upper	1.512561	2.259173
B3_upper	1.178688	1.281660
B4_upper	1.721434	1.201865
C1_lower	1.973616	2.530475
C1_upper	2.535180	3.128114
C2_lower	2.383037	2.936735
C2_upper	2.063887	3.138050
C4_upper	1.089146	1.983429
D3_upper	1.377064	1.248904

```
Min temperature: 23
Lesser than min:
['A3_lower' 'A4_lower' 'B2_upper' 'B3_upper' 'B3_lower' 'B4_lower'
 'B4_upper' 'C1_upper' 'C4_upper' 'D1_lower' 'D2_upper' 'D3_upper']
Deviation:
```

	dev_temp_a	dev_temp_b

plantcube	dev_temp_a	dev_temp_b

plantcube		
A3_lower	1.728105	1.250510
A4_lower	1.919171	1.343106
B2_upper	1.266284	1.058648
B3_lower	1.098661	2.148593
B3_upper	1.547847	1.199997
B4_lower	1.068134	1.281754
B4_upper	1.216667	3.995207
C1_upper	1.215097	1.135381
C4_upper	1.268151	1.695874
D1_lower	1.799816	1.477813
D2_upper	1.119141	1.052926
D3_upper	1.150888	1.062246

```
**************************************************
Recipe  13


Max temperature: 23
Greater than max:
['A4_lower' 'A4_upper' 'E3']
Deviation:
```

plantcube	dev_temp_a	dev_temp_b
A4_lower	2.540683	3.384385
A4_upper	3.241186	3.694953
E3	2.472052	3.310782

```
Min temperature: 23
```

```
**************************************************
```

In [56]:
```
#above the max target
final_rh
```

Out[56]:

	timestamp	connected	plantcube	recipe_id	temp_b	temp_a	mode	layers	recipe	dev_temp_a	dev_temp_b
0	2022-05-20 13:16:59.203	1	A3_upper	1.652862e+09	26.774872	26.010000	0.0	[[{'periods': [{'duration': Decimal('86400'), ...	1	1.010000	1.774872
1	2022-05-18 08:40:20.637	1	B1_lower	1.652863e+09	26.910000	26.580000	0.0	[[{'periods': [{'duration': Decimal('86400'), ...	1	1.580000	1.910000
2	2022-05-18 08:43:09.763	1	B1_lower	1.652863e+09	26.810000	26.543556	0.0	[[{'periods': [{'duration': Decimal('86400'), ...	1	1.543556	1.810000
3	2022-05-18 08:44:18.715	1	B1_lower	1.652863e+09	26.710000	26.528697	0.0	[[{'periods': [{'duration': Decimal('86400'), ...	1	1.528697	1.710000
4	2022-05-18 08:45:28.785	1	B1_lower	1.652863e+09	26.610000	26.513598	0.0	[[{'periods': [{'duration': Decimal('86400'), ...	1	1.513598	1.610000
...	...	...	...	...	...	...	...	...	...	...	...
11879	2022-05-24 15:03:12.389	1	E3	1.653317e+09	25.840000	24.767420	0.0	[[{'periods': [{'duration': Decimal('86400'), ...	13	1.767420	2.840000
11880	2022-05-24 15:03:21.381	1	E3	1.653317e+09	25.800329	24.700000	0.0	[[{'periods': [{'duration': Decimal('86400'), ...	13	1.700000	2.800329
11881	2022-05-24 15:03:58.364	1	E3	1.653317e+09	25.637166	24.400000	0.0	[[{'periods': [{'duration': Decimal('86400'), ...	13	1.400000	2.637166
11882	2022-05-24 15:04:20.388	1	E3	1.653317e+09	25.540000	24.242671	0.0	[[{'periods': [{'duration': Decimal('86400'), ...	13	1.242671	2.540000
11883	2022-05-24 15:04:40.360	1	E3	1.653317e+09	25.466014	24.100000	0.0	[[{'periods': [{'duration': Decimal('86400'), ...	13	1.100000	2.466014

11884 rows × 11 columns

In [57]:
```
final_rh['date_time1'] = final_rh['timestamp'].dt.date
```

In [58]:
```
#below the min target
final_rl
```

Out[58]:

	timestamp	connected	plantcube	recipe_id	temp_b	temp_a	mode	layers	recipe	dev_temp_a	dev_temp_b
0	2022-06-01 06:08:44.697	1	A3_lower	1.652862e+09	21.970000	21.917117	0.0	[[{'periods': [{'duration': Decimal('86400'), ...	1	1.082883	1.030000
1	2022-06-01 06:10:37.685	1	A3_lower	1.652862e+09	21.870000	21.939101	0.0	[[{'periods': [{'duration': Decimal('86400'), ...	1	1.060899	1.130000
2	2022-05-29 03:21:23.821	1	A3_upper	1.652862e+09	21.990000	21.685498	0.0	[[{'periods': [{'duration': Decimal('86400'), ...	1	1.314502	1.010000
3	2022-05-29 05:43:14.170	1	A3_upper	1.652862e+09	21.999415	21.710000	0.0	[[{'periods': [{'duration': Decimal('86400'), ...	1	1.290000	1.000585
4	2022-05-29 05:43:25.166	1	A3_upper	1.652862e+09	21.990000	21.700057	0.0	[[{'periods': [{'duration': Decimal('86400'), ...	1	1.299943	1.010000
...	...	...	...	...	...	...	...	...	...	...	...
5934	2022-08-12 16:51:51.025	1	D3_upper	1.659614e+09	21.950000	21.850000	0.0	[[{'periods': [{'duration': Decimal('86400'), ...	12	1.150000	1.050000
5935	2022-08-12 17:09:17.970	1	D3_upper	1.659614e+09	21.950000	21.950000	0.0	[[{'periods': [{'duration': Decimal('86400'), ...	12	1.050000	1.050000
5936	2022-08-12 18:35:43.634	1	D3_upper	1.659614e+09	21.950000	21.950000	0.0	[[{'periods': [{'duration': Decimal('86400'), ...	12	1.050000	1.050000
5937	2022-08-12 18:36:22.548	1	D3_upper	1.659614e+09	21.950000	21.850000	0.0	[[{'periods': [{'duration': Decimal('86400'), ...	12	1.150000	1.050000
5938	2022-08-12 18:51:13.576	1	D3_upper	1.659614e+09	21.950000	21.950000	0.0	[[{'periods': [{'duration': Decimal('86400'), ...	12	1.050000	1.050000

5939 rows × 11 columns

In [59]:
```
final_rl['date_time1'] = final_rl['timestamp'].dt.date
```

In [60]:
```
#above the max target - deviation
frh_g1 = final_rh.groupby('plantcube')['plantcube','dev_temp_a','dev_temp_b'].mean(numeric_only = True)
frh_g2 = final_rh.groupby('plantcube')['plantcube'].count()
frh_g3 = final_rh.groupby([final_rh.plantcube]).date_time1.nunique()
frh = pd.concat([frh_g1, frh_g2,frh_g3],axis=1)
frh.rename(columns = {'plantcube':'No of records considered', 'date_time1':'No of days included'}, inplace = True)
frh
```

Out[60]:

plantcube	dev_temp_a	dev_temp_b	No of records considered	No of days included
A1_lower	1.896712	4.029589	181	5
A1_upper	2.430824	2.743593	72	2
A2_lower	1.089946	1.479516	4	1
A2_upper	1.305375	1.881106	8	1
A3_lower	1.622508	2.385450	282	5
A3_upper	2.807249	3.871564	678	8
A4_lower	2.279905	3.715210	507	8
A4_upper	3.084991	3.600194	110	2
B1_lower	1.821253	3.113122	280	7
B1_upper	1.102652	1.153916	5	1
B2_upper	1.611905	2.965129	149	5
B3_lower	2.484381	2.845643	1727	34
B3_upper	1.512690	2.218292	35	3
B4_lower	2.052701	3.587073	377	6
B4_upper	3.968727	2.208459	1582	12
C1_lower	1.739465	2.174208	152	3
C1_upper	2.527081	3.514780	620	9
C2_lower	2.479345	2.407972	176	4
C2_upper	1.859649	2.803871	173	2
C3_lower	1.883826	2.789862	18	2

plantcube	dev_temp_a	dev_temp_b	No of records considered	No of days included
C4_lower	2.034994	3.780962	285	4
C4_upper	1.089146	1.983429	16	1
D1_upper	1.188317	1.254832	19	2
D2_lower	1.740768	2.249390	33	3
D2_upper	1.435727	1.426184	15	1
D3_lower	2.600973	2.982633	48	2
D3_upper	1.376525	3.261612	96	5
E1	1.980452	3.259200	369	5
E3	2.292734	3.848553	189	12
E4	2.559452	2.559789	994	21
E5	3.735318	4.444904	253	8
E6	2.281961	3.255328	2266	29
E7	2.471438	3.103438	86	4
E8	1.686717	2.643841	23	1
E9	2.158994	2.028715	56	2

In [61]:
```
#below the min target - deviation
frl_g1 = final_rl.groupby('plantcube')['plantcube','dev_temp_a','dev_temp_b'].mean(numeric_only = True)
frl_g2 = final_rl.groupby('plantcube')['plantcube'].count()
frl_g3 = final_rl.groupby([final_rl.plantcube]).date_time1.nunique()
frl = pd.concat([frl_g1, frl_g2,frl_g3],axis=1)
frl.rename(columns = {'plantcube':'No of records considered', 'date_time1':'No of days included'}, inplace = True)
frl
```

Out[61]:

plantcube	dev_temp_a	dev_temp_b	No of records considered	No of days included
A2_upper	1.279577	1.072466	12	2
A3_lower	1.725749	1.249897	557	7
A3_upper	1.263965	1.088426	108	9
A4_lower	1.919171	1.343106	27	1
B1_lower	3.605712	1.371740	15	1
B2_upper	1.246933	1.058144	62	7
B3_lower	1.098661	2.148593	895	21
B3_upper	1.553665	1.199308	714	10
B4_lower	1.067423	1.275172	489	19
B4_upper	1.212999	3.895249	1772	8
C1_lower	1.704481	1.150439	29	4
C1_upper	1.215097	1.135381	29	2
C2_lower	1.217683	1.197560	222	21
C2_upper	1.594160	1.095108	18	3
C4_upper	1.268151	1.695874	17	1
D1_lower	1.799816	1.477813	51	1
D1_upper	1.161529	1.178892	349	32
D2_upper	1.118065	1.093361	140	11
D3_upper	1.612837	1.208719	99	3
E1	1.209945	1.038332	7	4

plantcube	dev_temp_a	dev_temp_b	No of records considered	No of days included
E4	1.081994	1.335529	267	24
E9	2.631065	2.869274	60	3

In [62]:
```
#merge two dataframes
merge = pd.concat([final_rh, final_rl], axis=0)
```

In [63]:
```
merge.groupby('plantcube')['plantcube','dev_temp_a','dev_temp_b'].mean(numeric_only = True)
```

Out[63]:

plantcube	dev_temp_a	dev_temp_b
A1_lower	1.896712	4.029589
A1_upper	2.430824	2.743593
A2_lower	1.089946	1.479516
A2_upper	1.289896	1.395922
A3_lower	1.691048	1.631573
A3_upper	2.595195	3.489148
A4_lower	2.261666	3.595272
A4_upper	3.084991	3.600194
B1_lower	1.911988	3.024577
B1_upper	1.102652	1.153916
B2_upper	1.504662	2.404783
B3_lower	2.011376	2.607710
B3_upper	1.551751	1.246924
B4_lower	1.496349	2.281623
B4_upper	2.512809	3.099631
C1_lower	1.733860	2.010179
C1_upper	2.468456	3.408459
C2_lower	1.775604	1.732817
C2_upper	1.834629	2.642835
C3_lower	1.883826	2.789862
C4_lower	2.034994	3.780962
C4_upper	1.181361	1.835294
D1_lower	1.799816	1.477813
D1_upper	1.162912	1.182813

plantcube	dev_temp_a	dev_temp_b
D2_lower	1.740768	2.249390
D2_upper	1.148806	1.125570
D3_lower	2.600973	2.982633
D3_upper	1.496499	2.219374
E1	1.966108	3.217854
E3	2.292734	3.848553
E4	2.246620	2.300568
E5	3.735318	4.444904
E6	2.281961	3.255328
E7	2.471438	3.103438
E8	1.686717	2.643841
E9	2.403169	2.463487

In [64]:
```
m_g1 = merge.groupby('plantcube')['plantcube','dev_temp_a','dev_temp_b'].mean(numeric_only = True)
m_g2 = merge.groupby('plantcube')['plantcube'].count()
m_g3 = merge.groupby([merge.plantcube]).date_time1.nunique()
m = pd.concat([m_g1, m_g2,m_g3],axis=1)
m.rename(columns = {'plantcube':'No of records considered', 'date_time1':'No of days included'}, inplace = True)
m
```

Out[64]:

plantcube	dev_temp_a	dev_temp_b	No of records considered	No of days included
A1_lower	1.896712	4.029589	181	5
A1_upper	2.430824	2.743593	72	2
A2_lower	1.089946	1.479516	4	1
A2_upper	1.289896	1.395922	20	3
A3_lower	1.691048	1.631573	839	11
A3_upper	2.595195	3.489148	786	16
A4_lower	2.261666	3.595272	534	9
A4_upper	3.084991	3.600194	110	2
B1_lower	1.911988	3.024577	295	7
B1_upper	1.102652	1.153916	5	1
B2_upper	1.504662	2.404783	211	12
B3_lower	2.011376	2.607710	2622	54
B3_upper	1.551751	1.246924	749	12
B4_lower	1.496349	2.281623	866	24
B4_upper	2.512809	3.099631	3354	19
C1_lower	1.733860	2.010179	181	6
C1_upper	2.468456	3.408459	649	11
C2_lower	1.775604	1.732817	398	25
C2_upper	1.834629	2.642835	191	5
C3_lower	1.883826	2.789862	18	2

plantcube	dev_temp_a	dev_temp_b	No of records considered	No of days included
C4_lower	2.034994	3.780962	285	4
C4_upper	1.181361	1.835294	33	2
D1_lower	1.799816	1.477813	51	1
D1_upper	1.162912	1.182813	368	34
D2_lower	1.740768	2.249390	33	3
D2_upper	1.148806	1.125570	155	12
D3_lower	2.600973	2.982633	48	2
D3_upper	1.496499	2.219374	195	8
E1	1.966108	3.217854	376	9
E3	2.292734	3.848553	189	12
E4	2.246620	2.300568	1261	45
E5	3.735318	4.444904	253	8
E6	2.281961	3.255328	2266	29
E7	2.471438	3.103438	86	4
E8	1.686717	2.643841	23	1
E9	2.403169	2.463487	116	5

In [65]:
```
m['diff a&b'] = abs(m.dev_temp_a - m.dev_temp_b)
m = m.round(3)
m
```

Out[65]:

plantcube	dev_temp_a	dev_temp_b	No of records considered	No of days included	diff a&b
A1_lower	1.897	4.030	181	5	2.133
A1_upper	2.431	2.744	72	2	0.313
A2_lower	1.090	1.480	4	1	0.390
A2_upper	1.290	1.396	20	3	0.106
A3_lower	1.691	1.632	839	11	0.059
A3_upper	2.595	3.489	786	16	0.894
A4_lower	2.262	3.595	534	9	1.334
A4_upper	3.085	3.600	110	2	0.515
B1_lower	1.912	3.025	295	7	1.113
B1_upper	1.103	1.154	5	1	0.051
B2_upper	1.505	2.405	211	12	0.900
B3_lower	2.011	2.608	2622	54	0.596
B3_upper	1.552	1.247	749	12	0.305
B4_lower	1.496	2.282	866	24	0.785
B4_upper	2.513	3.100	3354	19	0.587
C1_lower	1.734	2.010	181	6	0.276
C1_upper	2.468	3.408	649	11	0.940
C2_lower	1.776	1.733	398	25	0.043
C2_upper	1.835	2.643	191	5	0.808
C3_lower	1.884	2.790	18	2	0.906
C4_lower	2.035	3.781	285	4	1.746
C4_upper	1.181	1.835	33	2	0.654

plantcube	dev_temp_a	dev_temp_b	No of records considered	No of days included	diff a&b
D1_lower	1.800	1.478	51	1	0.322
D1_upper	1.163	1.183	368	34	0.020
D2_lower	1.741	2.249	33	3	0.509
D2_upper	1.149	1.126	155	12	0.023
D3_lower	2.601	2.983	48	2	0.382
D3_upper	1.496	2.219	195	8	0.723
E1	1.966	3.218	376	9	1.252
E3	2.293	3.849	189	12	1.556
E4	2.247	2.301	1261	45	0.054
E5	3.735	4.445	253	8	0.710
E6	2.282	3.255	2266	29	0.973
E7	2.471	3.103	86	4	0.632
E8	1.687	2.644	23	1	0.957
E9	2.403	2.463	116	5	0.060

In [66]:
```
m.to_excel("m4.xlsx")
```

In [ ]:
```

```