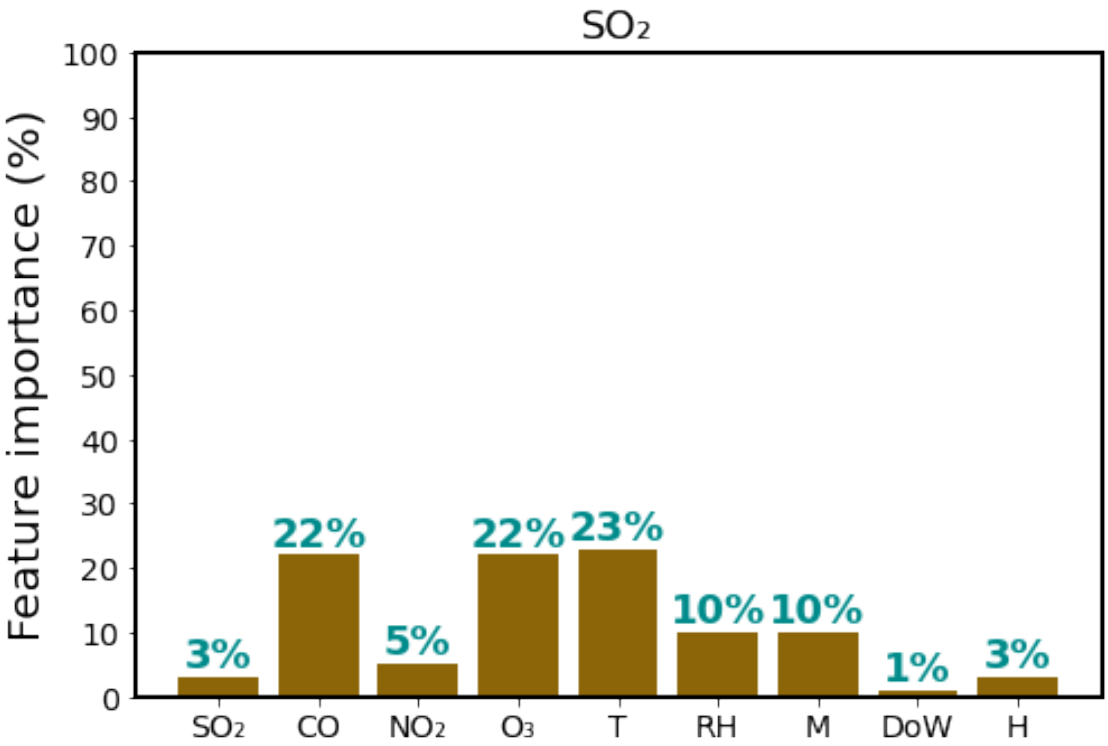
Calibration Schemes Plots 2

April 23, 2023

1 Resolution

[1]:
```
import numpy as np
SUB = str.maketrans("0123456789", "₀₁₂₃₄₅₆₇₈₉")
SUP = str.maketrans("0123456789", "⁰¹²³⁴⁵⁶⁷⁸⁹")
import matplotlib.pyplot as plt
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
langs = ['SO2'.translate(SUB),'CO'.translate(SUB),'NO2'.translate(SUB),
         'O3'.translate(SUB), 'T', 'RH', 'M', 'DoW','H']
students = [3,22,5,22,23,10,10,1,3]
A=[ 3., 22.,  5., 22., 23., 10., 10.,  1.,  3.]
graph=ax.bar(langs,students, color='#8B6508')
i = 0
for p in graph:
    width = p.get_width()
    height = p.get_height()
    x, y = p.get_xy()
    plt.text(x+width/2,
             y+height+1.5,
             str(students[i])+'%',
             ha='center',
             weight='bold',fontsize=18, color='#008B8B')
    i+=1
plt.xticks(fontsize=14)
plt.yticks(fontsize=14)
plt.yticks(np.arange(0,101, step=10))
plt.ylabel('Feature importance (%)', fontsize=20)
plt.setp(ax.spines.values(), linewidth=2)
plt.title('SO2'.translate(SUB),fontsize=18)
plt.savefig("F_SO2.pdf", format="pdf", bbox_inches="tight")
plt.show()
```



[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

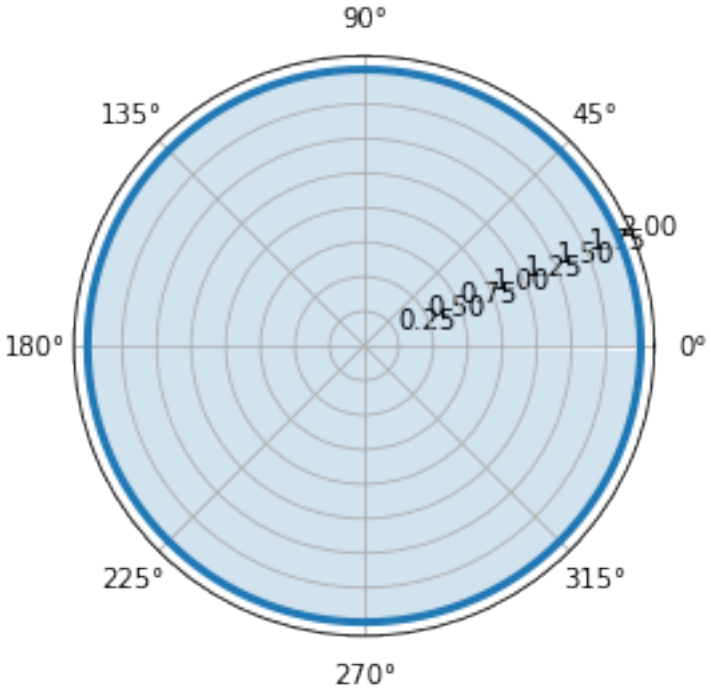
2 SAMPLED

[2]:
```
NO2_1_RMSE=[0.341, 0.2793, 0.2682, 0.2463, 0.2402, 0.2323, 0.2212, 0.1902]
NO2_3_RMSE=[0.253, 0.211, 0.199, 0.1932, 0.1885, 0.1778, 0.1772, 0.162]
NO2_6_RMSE=[0.2526, 0.2166, 0.2014, 0.2, 0.1958, 0.185, 0.1788, 0.16]
```

[3]:
```
CO_1_REU=[69.35, 43.73, 37.19, 31.27, 28.78, 27.5, 21.78, 18.6]
CO_3_REU=[48.18, 31.86, 25.64, 23.85, 21.72, 20.46, 17.9, 17.28]
CO_6_REU=[47.42, 32.62, 24.54, 26.4, 24.46, 22.99, 19.02, 16.86]
```

[4]:
```
import numpy as np
import matplotlib.pyplot as plt
```

```
theta = np.arange(0, 2, 1./180)*np.pi
r = abs(4*np.sin(2*theta))
r2 = 2 + 0*theta

#plt.polar(theta, r, lw=3)
plt.polar(theta, r2, lw=3)
plt.fill_between(theta, 0, r2, alpha=0.2)
plt.show()
```



[5]:
```
Oct_2=[0.38, 0.36, 0.32, 0.32, 0.31, 0.33, 0.24, 0.2]
Oct_60=[0.46, 0.43, 0.35, 0.34, 0.35, 0.35, 0.25, 0.23]

NOv_2=[0.38, 0.23, 0.21, 0.21, 0.19, 0.19, 0.18, 0.17]
Nov_60=[0.44, 0.29, 0.24, 0.23, 0.2, 0.2, 0.2, 0.18]

Dec_2=[0.22, 0.23, 0.22, 0.21, 0.21, 0.21, 0.22, 0.2]
Dec_60=[0.25, 0.26, 0.25, 0.23, 0.23, 0.23, 0.23, 0.2]

Jan_2=[0.22, 0.23, 0.22, 0.21, 0.21, 0.21, 0.22, 0.2]
Jan_60=[0.25, 0.26, 0.25, 0.23, 0.23, 0.23, 0.23, 0.2]
```

[6]:
```
NO2_2=[0.29, 0.26, 0.24, 0.24, 0.24, 0.23, 0.22, 0.21]
NO2_60=[0.32, 0.29, 0.27, 0.26, 0.26, 0.23, 0.23, 0.22]
```

[7]:
```
import pandas as pd
import numpy as np

df = pd.DataFrame({'id': range(5),
                   'value': range(100,600,100)})

# some other similar statistics
df['cum_sum'] = df['value'].cumsum()
df['count'] = range(1,len(df['value'])+1)
df['mov_avg'] = df['cum_sum'] / df['count']

# other statistics
df['rolling_mean2'] = df['value'].rolling(window=2).mean()

print(df)
```

```
   id  value  cum_sum  count  mov_avg  rolling_mean2
0   0    100      100      1    100.0            NaN
1   1    200      300      2    150.0          150.0
2   2    300      600      3    200.0          250.0
3   3    400     1000      4    250.0          350.0
4   4    500     1500      5    300.0          450.0
```

[ ]:

[ ]:

3 Sampled

4 CO

[8]:
```
import numpy as np
import random
CO_mean=442
CO_std=329


REU_6_CO_C=[57.8, 48.45, 47.43, 44.65, 41.26, 40.42, 39.92, 37.21]
REU_3_CO_C=[52.97, 46.13, 41.61, 41.54, 38.04, 36.91, 34.85, 32.93]
REU_6_CO=[36.52, 28.28, 18.63, 19.32, 18.54, 16.95, 15.02, 13.92]
REU_3_CO=[32.11, 29.09, 26.53, 23.61, 23.78, 21.97, 20.71, 17.9]
REU_1_CO=[53.11, 43.53, 39.8, 37.14, 35.34, 34.5, 29.72, 23.21]

RMSE_6_CO_C=[0.38, 0.35, 0.33, 0.29, 0.3, 0.27, 0.26, 0.22]
RMSE_3_CO_C=[0.33, 0.31, 0.3, 0.3, 0.28, 0.26, 0.25, 0.23]
RMSE_6_CO=[0.36, 0.27, 0.2, 0.18, 0.16, 0.15, 0.15, 0.14]
RMSE_3_CO=[0.39, 0.28, 0.2, 0.19, 0.2, 0.19, 0.18, 0.17]
```

```
RMSE_1_CO=[0.29, 0.27, 0.26, 0.26, 0.25, 0.25, 0.24, 0.22]

REU_1_CO_2=[59.48, 51.37, 46.96, 44.2 , 42.41, 40.36, 33.58, 26.23]
REU_3_CO_2=[35.96, 33.16, 29.71, 26.92, 28.06, 26.8 , 24.02, 20.58]
REU_6_CO_2=[42.36, 31.39, 21.8 , 22.99, 21.14, 20.51, 18.32, 15.87]
REU_3_CO_C_2=[59.33, 53.97, 49.52, 48.6 , 46.03, 41.71, 39.03, 39.19]
REU_6_CO_C_2=[68.2 , 53.78, 57.39, 49.56, 48.27, 47.7 , 43.91, 41.68]

RMSE_1_CO_2=[0.35, 0.32, 0.31, 0.32, 0.29, 0.3 , 0.27, 0.25]
RMSE_3_CO_2=[0.46, 0.31, 0.24, 0.22, 0.24, 0.22, 0.22, 0.2 ]
RMSE_6_CO_2=[0.4 , 0.31, 0.23, 0.2 , 0.19, 0.16, 0.17, 0.17]
RMSE_3_CO_C_2=[0.38, 0.34, 0.37, 0.35, 0.31, 0.29, 0.3 , 0.26]
RMSE_6_CO_C_2=[0.43, 0.39, 0.38, 0.35, 0.35, 0.32, 0.31, 0.26]

RMSE_6_CO_C_2
```

[8]:
```
[0.43, 0.39, 0.38, 0.35, 0.35, 0.32, 0.31, 0.26]
```

[9]:
```
RMSE_1_CO_LB=(np.array(RMSE_1_CO)*CO_mean)/(CO_mean+CO_std)
RMSE_1_CO_UB=(np.array(RMSE_1_CO)*CO_mean)/(CO_mean-CO_std)
```

[ ]:
```

```

[10]:
```
np.mean(REU_6_CO_2)-np.mean(REU_6_CO)
```

[10]:
```
3.3999999999999986
```

[ ]:
```

```

[11]:
```
#!pip install -U kaleido
import plotly.graph_objects as go
from functools import reduce
from itertools import product

z=[list(np.ceil(np.array(RMSE_6_CO_C_2)*100).astype(int)),list(np.ceil(np.
 ↪array(RMSE_6_CO_C)*100).astype(int)),
   list(np.ceil(np.array(RMSE_3_CO_C_2)*100).astype(int)),list(np.ceil(np.
 ↪array(RMSE_3_CO_C)*100).astype(int)),
  list(np.ceil(np.array(RMSE_1_CO_2)*100).astype(int)),list(np.ceil(np.
 ↪array(RMSE_1_CO)*100).astype(int))]
x=['<b>10</b>', '<b>20</b>', '<b>30</b>', '<b>40</b>', '<b>50</b>','<b>60</b>',
   '<b>70</b>','<b>80</b>']
y=['<b>6 month (1h)</b>','<b>6 month (2min)</b>','<b>3 month (1h)</b>','<b>3␣
 ↪month (2min)</b>',
   '<b>1 month (1h)','<b>1 month (2min)</b>']

def get_anno_text(z_value):
```

```
    annotations=[]
    a, b = len(z_value), len(z_value[0])
    flat_z = reduce(lambda x,y: x+y, z_value) # z_value.flat if you deal with␣
 ↪numpy
    coords = product(range(a), range(b))
    for pos, elem in zip(coords, flat_z):
        annotations.append({'font': {'color': 'black'},
                    'showarrow': False,
                    'text': str(elem),
                    'x': pos[1],
                    'y': pos[0],
                        'font.size':24   })
    return annotations

fig = go.Figure(data=go.Heatmap(
                    z=z,
                    x=x,
                    y=y,
                    hoverongaps = True, colorscale ='turbid',
    opacity=0.6,colorbar=dict(tickfont=dict(size=24)) ))#matter#

fig.update_layout(title={'text': "",
        'y':0.8,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},
          plot_bgcolor='rgba(0,0,0,0)',
    annotations = get_anno_text(z),
                  width=1000,
height=400,xaxis={'side': 'top'},margin=dict(l=20, r=20, t=20, b=20))

fig.update_xaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.update_yaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.write_image("RMSE_CO_C_POLAR.pdf",engine="kaleido")
#plt.savefig("table2a.pdf", format="pdf", bbox_inches="tight")
fig.show()
```

[12]:
```
#!pip install -U kaleido
import plotly.graph_objects as go
from functools import reduce
from itertools import product
```

```
z=[list(np.ceil(np.array(RMSE_6_CO_2)*100).astype(int)),list(np.ceil(np.
 ↪array(RMSE_6_CO)*100).astype(int)),
   list(np.ceil(np.array(RMSE_3_CO_2)*100).astype(int)),list(np.ceil(np.
 ↪array(RMSE_3_CO)*100).astype(int)),
  list(np.ceil(np.array(RMSE_1_CO_2)*100).astype(int)),list(np.ceil(np.
 ↪array(RMSE_1_CO)*100).astype(int))]
x=['<b>10</b>', '<b>20</b>', '<b>30</b>', '<b>40</b>', '<b>50</b>','<b>60</b>',
   '<b>70</b>','<b>80</b>']
y=['<b>6 month (1h)</b>','<b>6 month (2min)</b>','<b>3 month (1h)</b>','<b>3␣
 ↪month (2min)</b>',
   '<b>1 month (1h)','<b>1 month (2min)</b>']

def get_anno_text(z_value):
    annotations=[]
    a, b = len(z_value), len(z_value[0])
    flat_z = reduce(lambda x,y: x+y, z_value) # z_value.flat if you deal with␣
 ↪numpy
    coords = product(range(a), range(b))
    for pos, elem in zip(coords, flat_z):
        annotations.append({'font': {'color': 'black'},
                    'showarrow': False,
                    'text': str(elem),
                    'x': pos[1],
                    'y': pos[0],
                        'font.size':24   })
    return annotations

fig = go.Figure(data=go.Heatmap(
                    z=z,
                    x=x,
                    y=y,
                    hoverongaps = True, colorscale ='turbid',
    opacity=0.6,colorbar=dict(tickfont=dict(size=24)) ))#matter#

fig.update_layout(title={'text': "",
        'y':0.8,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},
           plot_bgcolor='rgba(0,0,0,0)',
    annotations = get_anno_text(z),
                width=1000,
height=400,xaxis={'side': 'top'},margin=dict(l=20, r=20, t=20, b=20))

fig.update_xaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
```

```
fig.update_yaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.write_image("RMSE_CO_S_POLAR.pdf",engine="kaleido")
#plt.savefig("table2a.pdf", format="pdf", bbox_inches="tight")
fig.show()
```

[13]:
```
#!pip install -U kaleido
import plotly.graph_objects as go
from functools import reduce
from itertools import product

z=[list(np.ceil(np.array(REU_6_CO_C_2)).astype(int)),list(np.ceil(np.
 ↪array(REU_6_CO_C)).astype(int)),
   list(np.ceil(np.array(REU_3_CO_C_2)).astype(int)),list(np.ceil(np.
 ↪array(REU_3_CO_C)).astype(int)),
  list(np.ceil(np.array(REU_1_CO_2)).astype(int)),list(np.ceil(np.
 ↪array(REU_1_CO)).astype(int))]
x=['<b>10</b>', '<b>20</b>', '<b>30</b>', '<b>40</b>', '<b>50</b>','<b>60</b>',
   '<b>70</b>','<b>80</b>']
y=['<b>6 month (1h)</b>','<b>6 month (2min)</b>','<b>3 month (1h)</b>','<b>3␣
 ↪month (2min)</b>',
   '<b>1 month (1h)','<b>1 month (2min)</b>']

def get_anno_text(z_value):
    annotations=[]
    a, b = len(z_value), len(z_value[0])
    flat_z = reduce(lambda x,y: x+y, z_value) # z_value.flat if you deal with␣
 ↪numpy
    coords = product(range(a), range(b))
    for pos, elem in zip(coords, flat_z):
        annotations.append({'font': {'color': 'black'},
                    'showarrow': False,
                    'text': str(elem),
                    'x': pos[1],
                    'y': pos[0],
                        'font.size':24   })
    return annotations

fig = go.Figure(data=go.Heatmap(
                    z=z,
                    x=x,
                    y=y,
                    hoverongaps = True, colorscale ='turbid',
    opacity=0.6,colorbar=dict(tickfont=dict(size=24)) ))#matter#

fig.update_layout(title={'text': "",
```

```
        'y':0.8,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},
           plot_bgcolor='rgba(0,0,0,0)',
    annotations = get_anno_text(z),
                  width=1000,
height=400,xaxis={'side': 'top'},margin=dict(l=20, r=20, t=20, b=20))

fig.update_xaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.update_yaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.write_image("REU_CO_C_POLAR.pdf",engine="kaleido")
#plt.savefig("table2a.pdf", format="pdf", bbox_inches="tight")
fig.show()
```

[14]:
```
#!pip install -U kaleido
import plotly.graph_objects as go
from functools import reduce
from itertools import product

z=[list(np.ceil(np.array(REU_6_CO_2)).astype(int)),list(np.ceil(np.
 ↪array(REU_6_CO)).astype(int)),
   list(np.ceil(np.array(REU_3_CO_2)).astype(int)),list(np.ceil(np.
 ↪array(REU_3_CO)).astype(int)),
  list(np.ceil(np.array(REU_1_CO_2)).astype(int)),list(np.ceil(np.
 ↪array(REU_1_CO)).astype(int))]
x=['<b>10</b>', '<b>20</b>', '<b>30</b>', '<b>40</b>', '<b>50</b>','<b>60</b>',
   '<b>70</b>','<b>80</b>']
y=['<b>6 month (1h)</b>','<b>6 month (2min)</b>','<b>3 month (1h)</b>','<b>3␣
 ↪month (2min)</b>',
   '<b>1 month (1h)','<b>1 month (2min)</b>']

def get_anno_text(z_value):
    annotations=[]
    a, b = len(z_value), len(z_value[0])
    flat_z = reduce(lambda x,y: x+y, z_value) # z_value.flat if you deal with␣
 ↪numpy
    coords = product(range(a), range(b))
    for pos, elem in zip(coords, flat_z):
        annotations.append({'font': {'color': 'black'},
                    'showarrow': False,
                    'text': str(elem),
                    'x': pos[1],
```

```
                    'y': pos[0],
                        'font.size':24   })
    return annotations

fig = go.Figure(data=go.Heatmap(
                    z=z,
                    x=x,
                    y=y,
                    hoverongaps = True, colorscale ='turbid',
    opacity=0.6,colorbar=dict(tickfont=dict(size=24)) ))#matter#

fig.update_layout(title={'text': "",
        'y':0.8,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},
           plot_bgcolor='rgba(0,0,0,0)',
    annotations = get_anno_text(z),
                  width=1000,
height=400,xaxis={'side': 'top'},margin=dict(l=20, r=20, t=20, b=20))

fig.update_xaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.update_yaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.write_image("REU_CO_S_POLAR.pdf",engine="kaleido")
#plt.savefig("table2a.pdf", format="pdf", bbox_inches="tight")
fig.show()
```

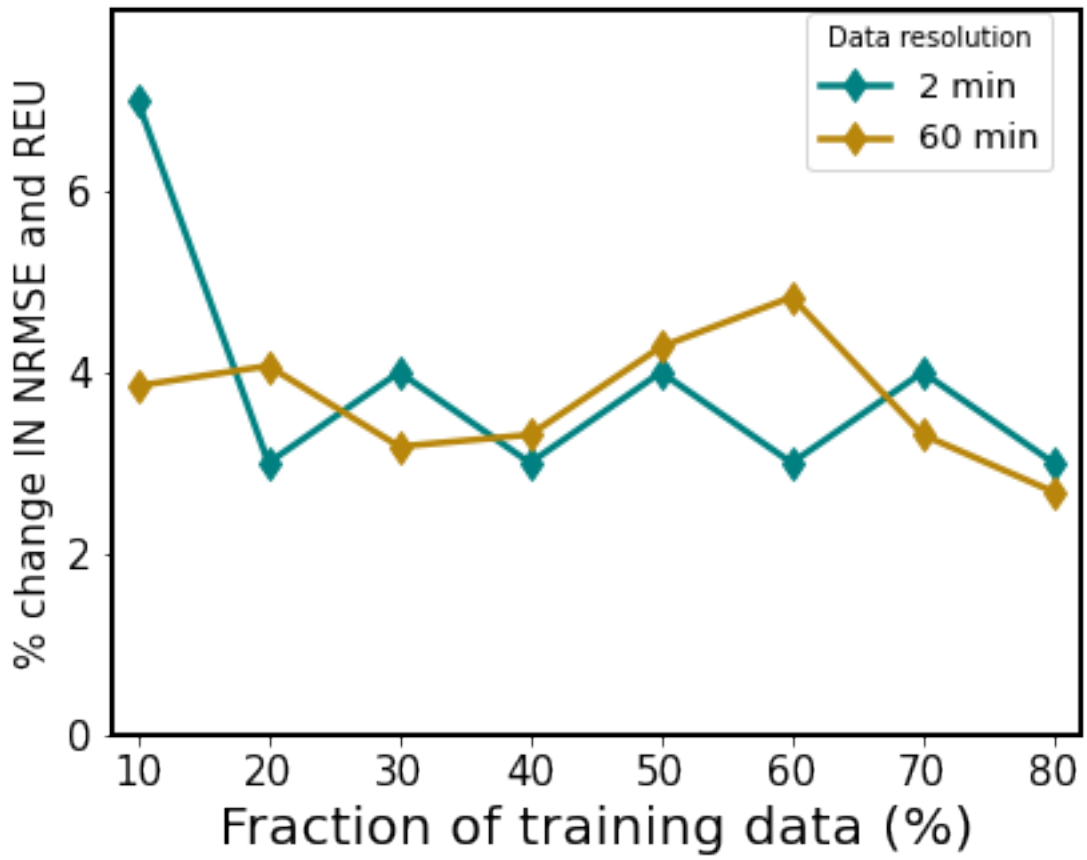
[15]:
```
np.array(RMSE_3_CO_2)*100-np.array(RMSE_3_CO)*100
```

[15]:
```
array([7., 3., 4., 3., 4., 3., 4., 3.])
```

[16]:
```
X=[10,20,30,40,50,60,70,80]
SUB = str.maketrans("0123456789", "      ")
SUP = str.maketrans("0123456789", " ¹²³    ")
import numpy as np
import matplotlib.pyplot as plt
A=[i for i in range(1,41)]
#Diff=[Diff[i] for i in range(16) if i%2==0]
#Diff2=[Diff2[i] for i in range(16) if i%2==0]
#Diff3=[Diff3[i] for i in range(16) if i%2==0]
Y_Test=[i for i in np.arange(0,41,1)]
fig= plt.figure(figsize=(6.5,5))
ax = fig.add_subplot(111)
```

```
#plt.scatter(A[1],Corr_mean[1], marker="d",s=200,color='darkgoldenrod', alpha=1)
#plt.scatter(A[1],Corr_mean2[1], marker="d",s=200,color='#CD5B45', alpha=0.9)
#plt.legend(['r','R^2'],loc = 2, bbox_to_anchor = (0.7,0.7), fontsize=16)



#plt.plot(X,np.
 ↪array(RMSE_6_CO)*100,color='darkgoldenrod',marker="d",markersize=9,linewidth=2.
 ↪5, alpha=1)
#plt.plot(X,RMSE_1_CO_2, color='darkgoldenrod',marker="o",markersize=9, alpha=1)
#plt.plot(X,RMSE_2_CO_2, color='teal',marker="d",markersize=9, alpha=1)
#plt.plot(X,np.array(RMSE_3_CO_2)*100,␣
 ↪color='#CD3333',marker="d",markersize=9,linewidth=2.5, alpha=1)
plt.plot(X,np.array(RMSE_3_CO_2)*100-np.array(RMSE_3_CO)*100,
         color='teal',marker="d",markersize=9,linewidth=2.5, alpha=1)
plt.plot(X,np.array(REU_3_CO_2)-np.array(REU_3_CO),
         color='darkgoldenrod',marker="d",markersize=9,linewidth=2.5, alpha=1)
plt.legend(['2 min','60 min'],
           title='Data resolution',loc = 2,
           bbox_to_anchor = (0.7,1.02), fontsize=13)

ax.set_ylim(bottom=0)
ax.set_xlim(left=0)
plt.xticks(fontsize=15)
plt.yticks(fontsize=15)
plt.show
#plt.xlim(0,40)
plt.ylim(0,8)
plt.xlim(8,82)
plt.yticks(np.arange(0,8, step=2))
#plt.xticks(np.arange(0,40, step=5))
#ax.set_xticks([0,5,10,15,20,25,30,35,40,45])
#ax.
 ↪set_xticklabels(['0','10','20','30','40','50','60','70','80','90'],fontsize=16)
plt.ylabel('% change IN NRMSE and REU',fontsize=15)
plt.setp(ax.spines.values(), linewidth=2)
#plt.xlabel('Tolerance, Tc (%)',fontsize=20)
plt.xlabel('Fraction of training data (%)',fontsize=20)
#plt.text(33,2, 'Tc=2',fontsize=17)
#plt.text(33,4, 'Tc=4',fontsize=17)
#plt.text(33,6, 'Tc=6',fontsize=17)
#plt.text(33,8, 'Tc=8',fontsize=17)
#plt.text(33,10, 'Tc=10',fontsize=17)
#plt.title('CO'.translate(SUB),fontsize=16)
textstr = 'CO-'.translate(SUB) +'Monthly'
props = dict(boxstyle='round', facecolor='white', alpha=1)
#plt.text(0.717, 0.975, textstr, transform=ax.transAxes, fontsize=15,
```

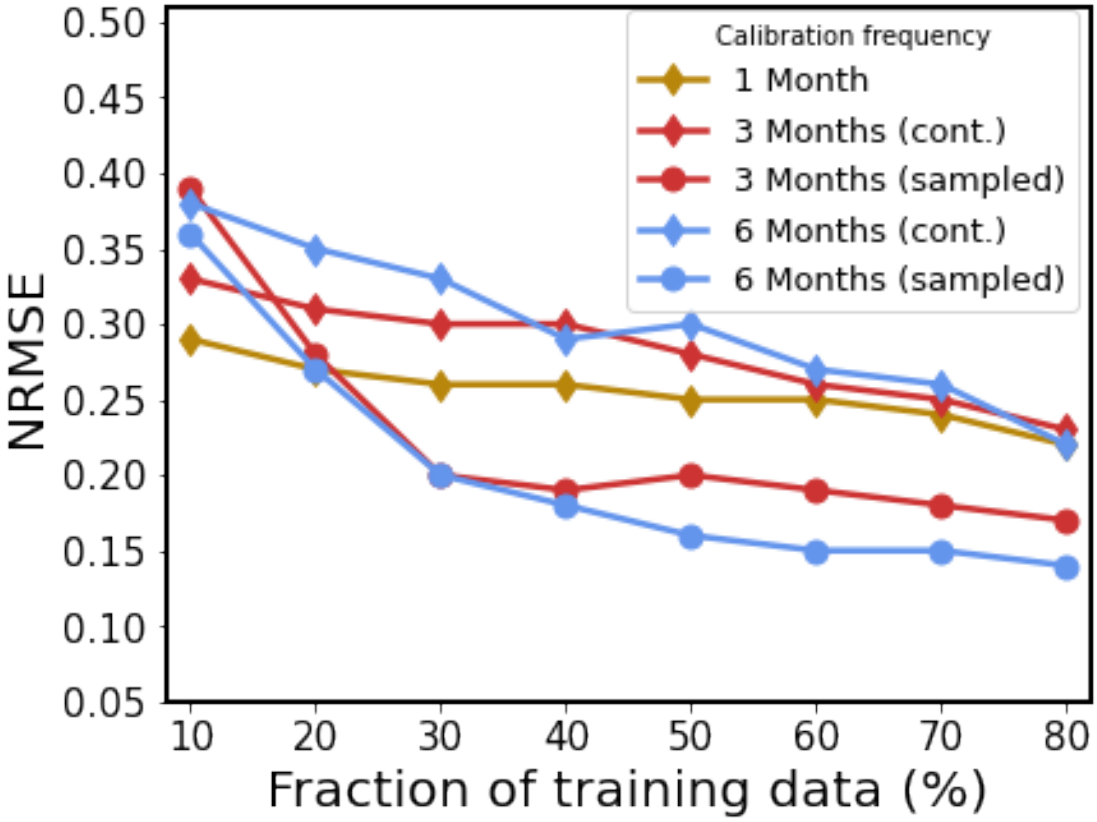
```
        #verticalalignment='top', bbox=props)
plt.savefig("Resolution_CO_S.pdf",format="pdf", bbox_inches="tight",dpi=1000)
plt.show()
```



[17]:
```
X=[10,20,30,40,50,60,70,80]
SUB = str.maketrans("0123456789", "      ")
SUP = str.maketrans("0123456789", " ¹²³    ")
import numpy as np
import matplotlib.pyplot as plt
A=[i for i in range(1,41)]
#Diff=[Diff[i] for i in range(16) if i%2==0]
#Diff2=[Diff2[i] for i in range(16) if i%2==0]
#Diff3=[Diff3[i] for i in range(16) if i%2==0]
Y_Test=[i for i in np.arange(0,41,1)]
fig= plt.figure(figsize=(6.5,5))
ax = fig.add_subplot(111)
#plt.scatter(A[1],Corr_mean[1], marker="d",s=200,color='darkgoldenrod', alpha=1)
#plt.scatter(A[1],Corr_mean2[1], marker="d",s=200,color='#CD5B45', alpha=0.9)
#plt.legend(['r','R^2'],loc = 2, bbox_to_anchor = (0.7,0.7), fontsize=16)
```

```
plt.plot(X,RMSE_1_CO,color='darkgoldenrod',marker="d",markersize=9,linewidth=2.
 ↪5, alpha=1)
#plt.plot(X,RMSE_1_CO_2, color='darkgoldenrod',marker="o",markersize=9, alpha=1)
#plt.plot(X,RMSE_2_CO_2, color='teal',marker="d",markersize=9, alpha=1)
plt.plot(X,RMSE_3_CO_C, color='#CD3333',marker="d",markersize=9,linewidth=2.5,␣
 ↪alpha=1)
plt.plot(X,RMSE_3_CO,color='#CD3333',marker="o",markersize=9,linewidth=2.5,␣
 ↪alpha=1)
#plt.plot(X,RMSE_3_CO_2, color='#CD3333',marker="o",markersize=9, alpha=1)
plt.plot(X,RMSE_6_CO_C, color='#6495ED',marker="d",markersize=9,linewidth=2.5,␣
 ↪alpha=1)
plt.plot(X,RMSE_6_CO,color='#6495ED',marker="o",markersize=9,linewidth=2.5,␣
 ↪alpha=1)
#plt.plot(X,RMSE_6_CO_2, color='#6495ED',marker="o",markersize=9, alpha=1)
plt.legend(['1 Month','3 Months (cont.)','3 Months (sampled)','6 Months (cont.
 ↪)',
            '6 Months (sampled)'],
           title='Calibration frequency',loc = 2,
           bbox_to_anchor = (0.48,1.02), fontsize=13)

ax.set_ylim(bottom=0)
ax.set_xlim(left=0)
plt.xticks(fontsize=15)
plt.yticks(fontsize=15)
plt.show
#plt.xlim(0,40)
plt.ylim(0.05,0.51)
plt.xlim(8,82)
plt.yticks(np.arange(0.05,0.51, step=0.05))
#plt.xticks(np.arange(0,40, step=5))
#ax.set_xticks([0,5,10,15,20,25,30,35,40,45])
#ax.
 ↪set_xticklabels(['0','10','20','30','40','50','60','70','80','90'],fontsize=16)
plt.ylabel('NRMSE',fontsize=20)
plt.setp(ax.spines.values(), linewidth=2)
#plt.xlabel('Tolerance, Tc (%)',fontsize=20)
plt.xlabel('Fraction of training data (%)',fontsize=20)
#plt.text(33,2, 'Tc=2',fontsize=17)
#plt.text(33,4, 'Tc=4',fontsize=17)
#plt.text(33,6, 'Tc=6',fontsize=17)
#plt.text(33,8, 'Tc=8',fontsize=17)
#plt.text(33,10, 'Tc=10',fontsize=17)
#plt.title('CO'.translate(SUB),fontsize=16)
```

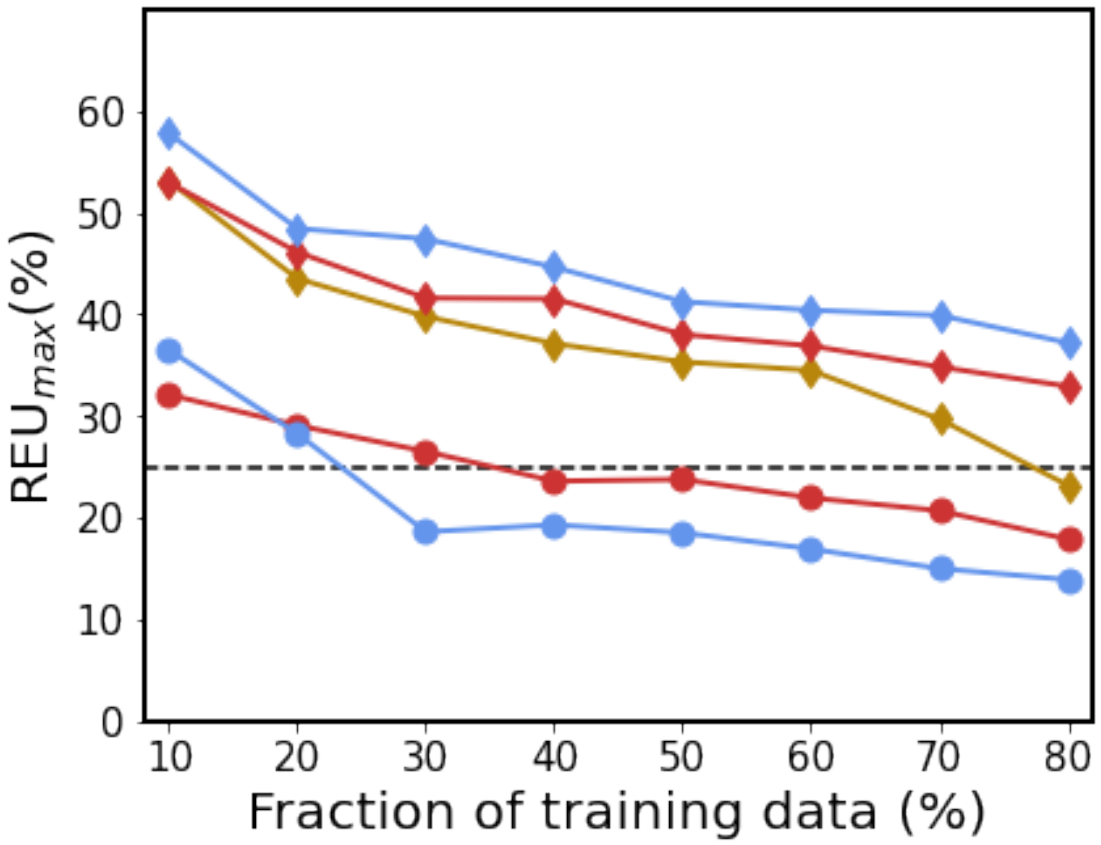
```
textstr = 'CO-'.translate(SUB) +'Monthly'
props = dict(boxstyle='round', facecolor='white', alpha=1)
#plt.text(0.717, 0.975, textstr, transform=ax.transAxes, fontsize=15,
        #verticalalignment='top', bbox=props)
plt.savefig("rmse_CO_S.pdf",format="pdf", bbox_inches="tight",dpi=1000)
plt.show()
```



[18]:
```
X=[10,20,30,40,50,60,70,80]
SUB = str.maketrans("0123456789", "₀₁₂₃₄₅₆₇₈₉")
SUP = str.maketrans("0123456789", "⁰¹²³⁴⁵⁶⁷⁸⁹")
sub = str.maketrans("max", "ₘₐₓ")
import numpy as np
import matplotlib.pyplot as plt
A=[i for i in range(1,41)]
#Diff=[Diff[i] for i in range(16) if i%2==0]
#Diff2=[Diff2[i] for i in range(16) if i%2==0]
#Diff3=[Diff3[i] for i in range(16) if i%2==0]
Y_Test=[i for i in np.arange(0,41,1)]
fig= plt.figure(figsize=(6.5,5))
ax = fig.add_subplot(111)
```

```
#plt.scatter(A[1],Corr_mean[1], marker="d",s=200,color='darkgoldenrod', alpha=1)
#plt.scatter(A[1],Corr_mean2[1], marker="d",s=200,color='#CD5B45', alpha=0.9)
#plt.legend(['r','R^2'],loc = 2, bbox_to_anchor = (0.7,0.7), fontsize=16)


plt.hlines([25], 0, 100, linestyles='dashed', color='black', linewidth=1.5)
plt.plot(X,REU_1_CO,color='darkgoldenrod',marker="d",markersize=9,linewidth=2,␣
 ↪alpha=1)
#plt.plot(X,RMSE_1_CO_2, color='darkgoldenrod',marker="o",markersize=9, alpha=1)
#plt.plot(X,RMSE_2_CO_2, color='teal',marker="d",markersize=9, alpha=1)
plt.plot(X,REU_3_CO_C, color='#CD3333',marker="d",markersize=9,linewidth=2,␣
 ↪alpha=1)
plt.plot(X,REU_3_CO,color='#CD3333',marker="o",markersize=9,linewidth=2,␣
 ↪alpha=1)
#plt.plot(X,RMSE_3_CO_2, color='#CD3333',marker="o",markersize=9, alpha=1)
plt.plot(X,REU_6_CO_C, color='#6495ED',marker="d",markersize=9,linewidth=2,␣
 ↪alpha=1)
plt.plot(X,REU_6_CO,color='#6495ED',marker="o",markersize=9,linewidth=2,␣
 ↪alpha=1)
#plt.plot(X,RMSE_6_CO_2, color='#6495ED',marker="o",markersize=9, alpha=1)
ax.set_ylim(bottom=0)
ax.set_xlim(left=0)
plt.xticks(fontsize=15)
plt.yticks(fontsize=15)
plt.show
#plt.xlim(0,40)
plt.ylim(0,70)
plt.xlim(8,82)
plt.yticks(np.arange(0,70, step=10))
#plt.xticks(np.arange(0,40, step=5))
#ax.set_xticks([0,5,10,15,20,25,30,35,40,45])
#ax.
 ↪set_xticklabels(['0','10','20','30','40','50','60','70','80','90'],fontsize=16)
plt.ylabel('REU$_{max}$(%)',fontsize=20)
plt.setp(ax.spines.values(), linewidth=2)
#plt.xlabel('Tolerance, Tc (%)',fontsize=20)
plt.xlabel('Fraction of training data (%)',fontsize=20)
#plt.text(33,2, 'Tc=2',fontsize=17)
#plt.text(33,4, 'Tc=4',fontsize=17)
#plt.text(33,6, 'Tc=6',fontsize=17)
#plt.text(33,8, 'Tc=8',fontsize=17)
#plt.text(33,10, 'Tc=10',fontsize=17)
#plt.title('CO'.translate(SUB),fontsize=16)
textstr = 'CO-'.translate(SUB) +'Monthly'
props = dict(boxstyle='round', facecolor='white', alpha=1)
#plt.text(0.717, 0.975, textstr, transform=ax.transAxes, fontsize=15,
```

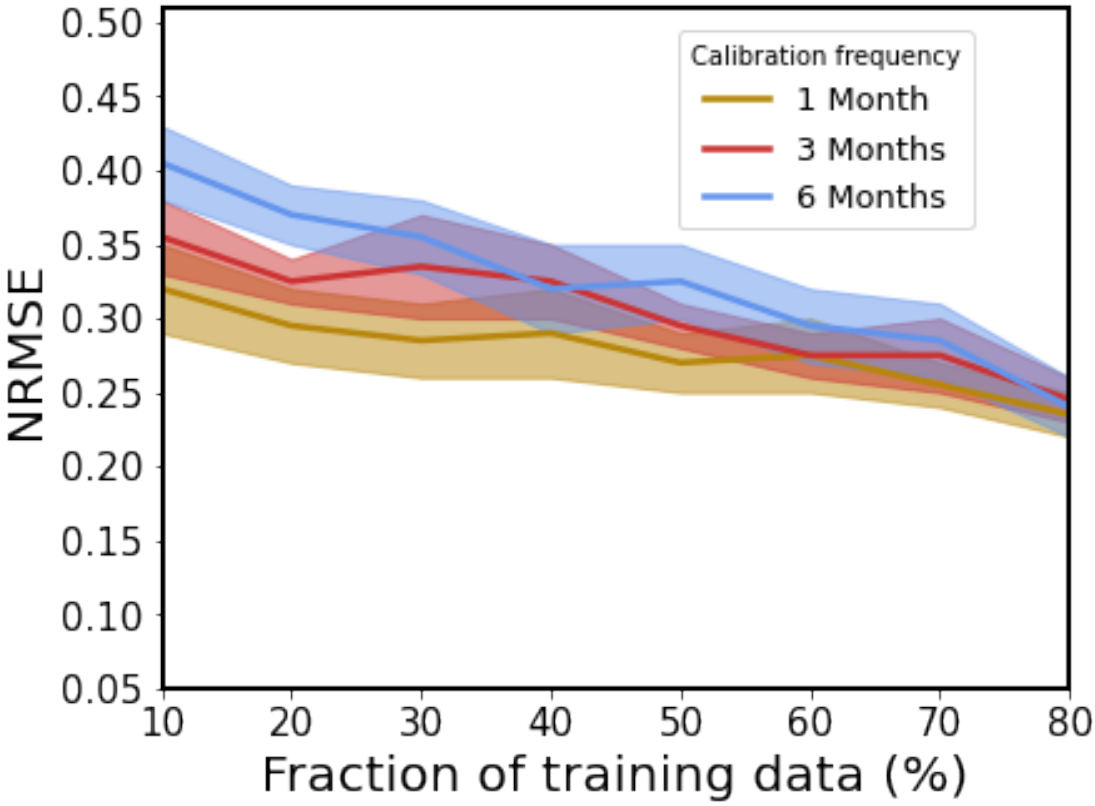
```
        #verticalalignment='top', bbox=props)
plt.savefig("reu_CO_S.pdf",format="pdf", bbox_inches="tight",dpi=1000)
plt.show()
```



[19]:
```
X=[10,20,30,40,50,60,70,80]
SUB = str.maketrans("0123456789", "      ")
SUP = str.maketrans("0123456789", " ¹²³    ")
import numpy as np
import matplotlib.pyplot as plt
A=[i for i in range(1,41)]
#Diff=[Diff[i] for i in range(16) if i%2==0]
#Diff2=[Diff2[i] for i in range(16) if i%2==0]
#Diff3=[Diff3[i] for i in range(16) if i%2==0]
Y_Test=[i for i in np.arange(0,41,1)]
fig= plt.figure(figsize=(6.5,5))
ax = fig.add_subplot(111)
#plt.scatter(A[1],Corr_mean[1], marker="d",s=200,color='darkgoldenrod', alpha=1)
#plt.scatter(A[1],Corr_mean2[1], marker="d",s=200,color='#CD5B45', alpha=0.9)
#plt.legend(['r','R^2'],loc = 2, bbox_to_anchor = (0.7,0.7), fontsize=16)
```

```
#plt.legend(['1 Month','3 Months','6 Months'],title='Calibration frequency',loc␣
↪= 2,
           #bbox_to_anchor = (0.55,0.99), fontsize=13)

plt.plot(X,0.5*(np.array(RMSE_1_CO)+np.array(RMSE_1_CO_2)),
         color='darkgoldenrod', linewidth=2.5,alpha=1)
#plt.plot(X,RMSE_1_CO_2, color='darkgoldenrod',marker="o",markersize=9, alpha=1)
#plt.plot(X,RMSE_2_CO_2, color='teal',marker="d",markersize=9, alpha=1)
plt.plot(X,0.5*(np.array(RMSE_3_CO_C)+np.array(RMSE_3_CO_C_2)),
         color='#CD3333',linewidth=2.5, alpha=1)
#plt.plot(X,RMSE_3_CO_2, color='#CD3333',marker="o",markersize=9, alpha=1)
plt.plot(X,0.5*(np.array(RMSE_6_CO_C)+np.array(RMSE_6_CO_C_2)),
         color='#6495ED',linewidth=2.5, alpha=1)
#plt.plot(X,RMSE_6_CO_2, color='#6495ED',marker="o",markersize=9, alpha=1)
plt.legend(['1 Month','3 Months','6 Months'],
           title='Calibration frequency',loc = 2,
           bbox_to_anchor = (0.55,0.99), fontsize=13)

plt.fill_between(X,RMSE_1_CO ,RMSE_1_CO_2 , color='darkgoldenrod',
                 interpolate=True, alpha=0.5)
plt.fill_between(X,RMSE_3_CO_C ,RMSE_3_CO_C_2 , color='#CD3333',
                  interpolate=True,alpha=0.5)
plt.fill_between(X,RMSE_6_CO_C ,RMSE_6_CO_C_2 , color='#6495ED',
                 interpolate=True, alpha=0.5)

ax.set_ylim(bottom=0)
ax.set_xlim(left=0)
plt.xticks(fontsize=15)
plt.yticks(fontsize=15)
plt.show
#plt.xlim(0,40)
plt.ylim(0.05,0.51)
plt.xlim(10,80)
plt.yticks(np.arange(0.05,0.51, step=0.05))
#plt.xticks(np.arange(0,40, step=5))
#ax.set_xticks([0,5,10,15,20,25,30,35,40,45])
#ax.
↪set_xticklabels(['0','10','20','30','40','50','60','70','80','90'],fontsize=16)
plt.ylabel('NRMSE',fontsize=20)
plt.setp(ax.spines.values(), linewidth=2)
#plt.xlabel('Tolerance, Tc (%)',fontsize=20)
plt.xlabel('Fraction of training data (%)',fontsize=20)
#plt.text(33,2, 'Tc=2',fontsize=17)
#plt.text(33,4, 'Tc=4',fontsize=17)
#plt.text(33,6, 'Tc=6',fontsize=17)
```

```
#plt.text(33,8, 'Tc=8',fontsize=17)
#plt.text(33,10, 'Tc=10',fontsize=17)
#plt.title('CO'.translate(SUB),fontsize=16)
textstr = 'CO-'.translate(SUB) +'Monthly'
props = dict(boxstyle='round', facecolor='white', alpha=1)
#plt.text(0.717, 0.975, textstr, transform=ax.transAxes, fontsize=15,
        #verticalalignment='top', bbox=props)
plt.savefig("rmse_CO_C.pdf",format="pdf", bbox_inches="tight",dpi=1000)
plt.show()
```



5 NO2

[20]:
```
REU_6_NO2_C=[55.48, 44.92, 42.44, 41.89, 39.33, 38.2, 33.26, 28.92]
REU_3_NO2_C=[52.42, 43.79, 41.9, 36.83, 34.3, 34.2, 33.02, 26.81]
REU_6_NO2=[51.4, 34.11, 24.84, 23.9, 20.3, 20.48, 15.89, 15.53]
REU_3_NO2=[49.2, 34.51, 28.07, 25.38, 21.24, 21.23, 18.28, 17.92]
REU_1_NO2=[77.47, 49.3, 40.7, 34.7, 31.81, 30.45, 23.76, 21.9]
```

```
RMSE_6_NO2_C=[0.36, 0.33, 0.3, 0.28, 0.26, 0.26, 0.26, 0.25]
RMSE_3_NO2_C=[0.34, 0.3, 0.29, 0.27, 0.25, 0.24, 0.24, 0.22]
RMSE_6_NO2=[0.26, 0.24, 0.19, 0.18, 0.17, 0.165, 0.16, 0.15]
RMSE_3_NO2=[0.28, 0.25, 0.2, 0.19, 0.18, 0.18, 0.17, 0.16]
RMSE_1_NO2=[0.38, 0.3, 0.28, 0.26, 0.25, 0.24, 0.23, 0.21]

REU_1_NO2_2=[93.74, 54.23, 45.18, 39.21, 37.22, 37.15, 26.37, 24.75]
REU_3_NO2_2=[57.07, 42.1 , 32.28, 30.46, 24.21, 25.26, 20.29, 20.07]
REU_6_NO2_2=[60.14, 38.54, 27.57, 26.29, 24.16, 22.53, 19.07, 18.79]
REU_3_NO2_C_2=[63.43, 48.17, 46.09, 44.56, 40.13, 41.04, 36.65, 31.64]
REU_6_NO2_C_2=[64.91, 53.45, 48.81, 47.34, 46.8 , 46.22, 36.59, 34.7 ]

RMSE_1_NO2_2=[0.43, 0.34, 0.32, 0.3 , 0.28, 0.26, 0.26, 0.23]
RMSE_3_NO2_2=[0.32, 0.3 , 0.23, 0.22, 0.22, 0.21, 0.2 , 0.18]
RMSE_6_NO2_2=[0.31, 0.28, 0.22, 0.2 , 0.2 , 0.18, 0.18, 0.17]
RMSE_3_NO2_C_2=[0.37, 0.34, 0.33, 0.33, 0.3 , 0.29, 0.29, 0.25]
RMSE_6_NO2_C_2=[0.42, 0.4 , 0.34, 0.31, 0.29, 0.29, 0.31, 0.28]
```

[21]:
```
np.mean(REU_6_NO2_2)-np.mean(REU_6_NO2)
```

[21]:
```
3.8300000000000054
```

[22]:
```
#!pip install -U kaleido
import plotly.graph_objects as go
from functools import reduce
from itertools import product

z=[list(np.ceil(np.array(RMSE_6_NO2_C_2)*100).astype(int)),list(np.ceil(np.
 ↪array(RMSE_6_NO2_C)*100).astype(int)),
   list(np.ceil(np.array(RMSE_3_NO2_C_2)*100).astype(int)),list(np.ceil(np.
 ↪array(RMSE_3_NO2_C)*100).astype(int)),
  list(np.ceil(np.array(RMSE_1_NO2_2)*100).astype(int)),list(np.ceil(np.
 ↪array(RMSE_1_NO2)*100).astype(int))]
x=['<b>10</b>', '<b>20</b>', '<b>30</b>', '<b>40</b>', '<b>50</b>','<b>60</b>',
   '<b>70</b>','<b>80</b>']
y=['<b>6 month (1h)</b>','<b>6 month (2min)</b>','<b>3 month (1h)</b>','<b>3␣
 ↪month (2min)</b>',
   '<b>1 month (1h)','<b>1 month (2min)</b>']

def get_anno_text(z_value):
    annotations=[]
    a, b = len(z_value), len(z_value[0])
    flat_z = reduce(lambda x,y: x+y, z_value) # z_value.flat if you deal with␣
 ↪numpy
    coords = product(range(a), range(b))
    for pos, elem in zip(coords, flat_z):
        annotations.append({'font': {'color': 'black'},
```

```
                    'showarrow': False,
                    'text': str(elem),
                    'x': pos[1],
                    'y': pos[0],
                        'font.size':24   })
    return annotations

fig = go.Figure(data=go.Heatmap(
                    z=z,
                    x=x,
                    y=y,
                    hoverongaps = True, colorscale ='turbid',
    opacity=0.6,colorbar=dict(tickfont=dict(size=24)) ))#matter#

fig.update_layout(title={'text': "",
        'y':0.8,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},
          plot_bgcolor='rgba(0,0,0,0)',
    annotations = get_anno_text(z),
                  width=1000,
height=400,xaxis={'side': 'top'},margin=dict(l=20, r=20, t=20, b=20))

fig.update_xaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.update_yaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.write_image("RMSE_NO2_C_POLAR.pdf",engine="kaleido")
#plt.savefig("table2a.pdf", format="pdf", bbox_inches="tight")
fig.show()
```

[23]:
```
#!pip install -U kaleido
import plotly.graph_objects as go
from functools import reduce
from itertools import product

z=[list(np.ceil(np.array(RMSE_6_NO2_2)*100).astype(int)),list(np.ceil(np.
 ↪array(RMSE_6_NO2)*100).astype(int)),
   list(np.ceil(np.array(RMSE_3_NO2_2)*100).astype(int)),list(np.ceil(np.
 ↪array(RMSE_3_NO2)*100).astype(int)),
  list(np.ceil(np.array(RMSE_1_NO2_2)*100).astype(int)),list(np.ceil(np.
 ↪array(RMSE_1_NO2)*100).astype(int))]
x=['<b>10</b>', '<b>20</b>', '<b>30</b>', '<b>40</b>', '<b>50</b>','<b>60</b>',
   '<b>70</b>','<b>80</b>']
```

```
y=['<b>6 month (1h)</b>','<b>6 month (2min)</b>','<b>3 month (1h)</b>','<b>3␣
 ↪month (2min)</b>',
   '<b>1 month (1h)','<b>1 month (2min)</b>']

def get_anno_text(z_value):
    annotations=[]
    a, b = len(z_value), len(z_value[0])
    flat_z = reduce(lambda x,y: x+y, z_value) # z_value.flat if you deal with␣
 ↪numpy
    coords = product(range(a), range(b))
    for pos, elem in zip(coords, flat_z):
        annotations.append({'font': {'color': 'black'},
                    'showarrow': False,
                    'text': str(elem),
                    'x': pos[1],
                    'y': pos[0],
                        'font.size':24   })
    return annotations

fig = go.Figure(data=go.Heatmap(
                    z=z,
                    x=x,
                    y=y,
                    hoverongaps = True, colorscale ='turbid',
    opacity=0.6,colorbar=dict(tickfont=dict(size=24)) ))#matter#

fig.update_layout(title={'text': "",
        'y':0.8,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},
           plot_bgcolor='rgba(0,0,0,0)',
    annotations = get_anno_text(z),
                width=1000,
height=400,xaxis={'side': 'top'},margin=dict(l=20, r=20, t=20, b=20))

fig.update_xaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.update_yaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.write_image("RMSE_NO2_S_POLAR.pdf",engine="kaleido")
#plt.savefig("table2a.pdf", format="pdf", bbox_inches="tight")
fig.show()
```

[24]:
```
#!pip install -U kaleido
import plotly.graph_objects as go
from functools import reduce
from itertools import product

z=[list(np.ceil(np.array(REU_6_NO2_C_2)).astype(int)),list(np.ceil(np.
 ↪array(REU_6_NO2_C)).astype(int)),
   list(np.ceil(np.array(REU_3_NO2_C_2)).astype(int)),list(np.ceil(np.
 ↪array(REU_3_NO2_C)).astype(int)),
  list(np.ceil(np.array(REU_1_NO2_2)).astype(int)),list(np.ceil(np.
 ↪array(REU_1_NO2)).astype(int))]
x=['<b>10</b>', '<b>20</b>', '<b>30</b>', '<b>40</b>', '<b>50</b>','<b>60</b>',
   '<b>70</b>','<b>80</b>']
y=['<b>6 month (1h)</b>','<b>6 month (2min)</b>','<b>3 month (1h)</b>','<b>3␣
 ↪month (2min)</b>',
   '<b>1 month (1h)','<b>1 month (2min)</b>']

def get_anno_text(z_value):
    annotations=[]
    a, b = len(z_value), len(z_value[0])
    flat_z = reduce(lambda x,y: x+y, z_value) # z_value.flat if you deal with␣
 ↪numpy
    coords = product(range(a), range(b))
    for pos, elem in zip(coords, flat_z):
        annotations.append({'font': {'color': 'black'},
                    'showarrow': False,
                    'text': str(elem),
                    'x': pos[1],
                    'y': pos[0],
                        'font.size':24   })
    return annotations

fig = go.Figure(data=go.Heatmap(
                    z=z,
                    x=x,
                    y=y,
                    hoverongaps = True, colorscale ='turbid',
    opacity=0.6,colorbar=dict(tickfont=dict(size=24)) ))#matter#

fig.update_layout(title={'text': "",
        'y':0.8,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},
          plot_bgcolor='rgba(0,0,0,0)',
    annotations = get_anno_text(z),
                width=1000,
```

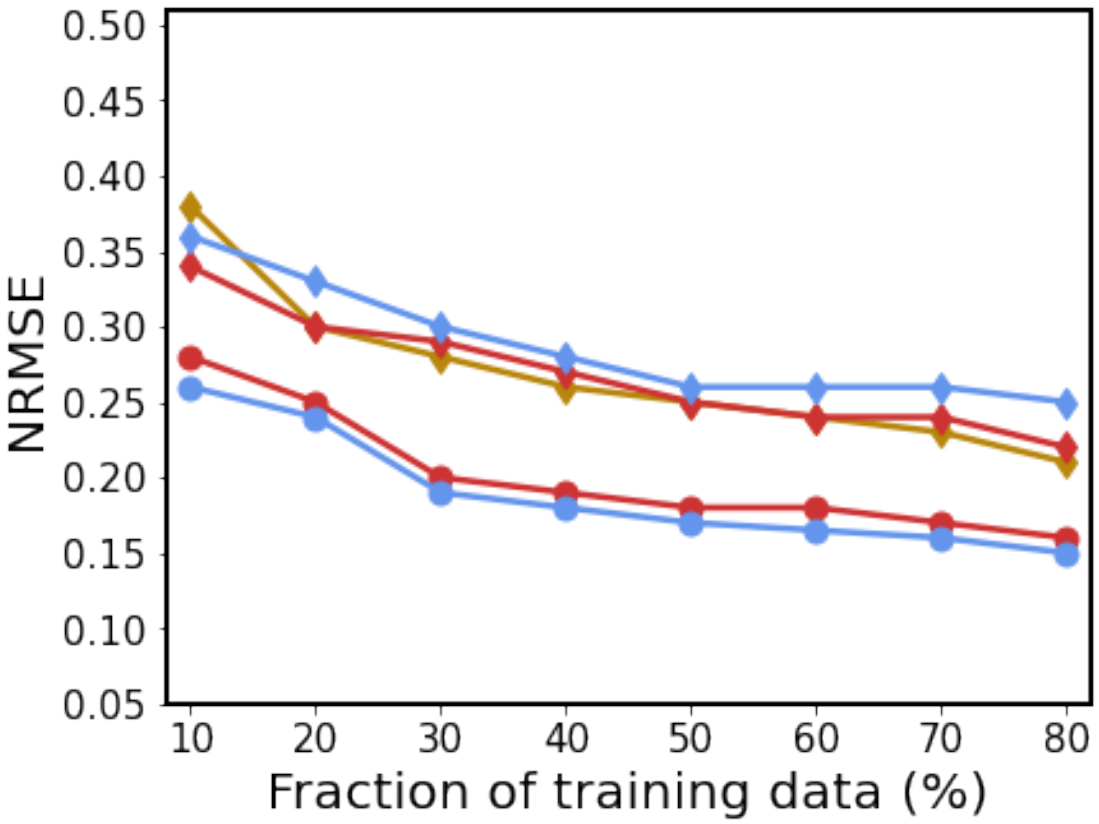
```
height=400,xaxis={'side': 'top'},margin=dict(l=20, r=20, t=20, b=20))

fig.update_xaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.update_yaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.write_image("REU_NO2_C_POLAR.pdf",engine="kaleido")
#plt.savefig("table2a.pdf", format="pdf", bbox_inches="tight")
fig.show()
```

[25]:
```
#!pip install -U kaleido
import plotly.graph_objects as go
from functools import reduce
from itertools import product

z=[list(np.ceil(np.array(REU_6_NO2_2)).astype(int)),list(np.ceil(np.
 ↪array(REU_6_NO2)).astype(int)),
   list(np.ceil(np.array(REU_3_NO2_2)).astype(int)),list(np.ceil(np.
 ↪array(REU_3_NO2)).astype(int)),
  list(np.ceil(np.array(REU_1_NO2_2)).astype(int)),list(np.ceil(np.
 ↪array(REU_1_NO2)).astype(int))]
x=['<b>10</b>', '<b>20</b>', '<b>30</b>', '<b>40</b>', '<b>50</b>','<b>60</b>',
   '<b>70</b>','<b>80</b>']
y=['<b>6 month (1h)</b>','<b>6 month (2min)</b>','<b>3 month (1h)</b>','<b>3␣
 ↪month (2min)</b>',
   '<b>1 month (1h)','<b>1 month (2min)</b>']

def get_anno_text(z_value):
    annotations=[]
    a, b = len(z_value), len(z_value[0])
    flat_z = reduce(lambda x,y: x+y, z_value) # z_value.flat if you deal with␣
 ↪numpy
    coords = product(range(a), range(b))
    for pos, elem in zip(coords, flat_z):
        annotations.append({'font': {'color': 'black'},
                    'showarrow': False,
                    'text': str(elem),
                    'x': pos[1],
                    'y': pos[0],
                        'font.size':24   })
    return annotations

fig = go.Figure(data=go.Heatmap(
                   z=z,
                   x=x,
```

```
                    y=y,
                    hoverongaps = True, colorscale ='turbid',
    opacity=0.6,colorbar=dict(tickfont=dict(size=24)) ))#matter#

fig.update_layout(title={'text': "",
        'y':0.8,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},
          plot_bgcolor='rgba(0,0,0,0)',
    annotations = get_anno_text(z),
                  width=1000,
height=400,xaxis={'side': 'top'},margin=dict(l=20, r=20, t=20, b=20))

fig.update_xaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.update_yaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.write_image("REU_NO2_S_POLAR.pdf",engine="kaleido")
#plt.savefig("table2a.pdf", format="pdf", bbox_inches="tight")
fig.show()
```

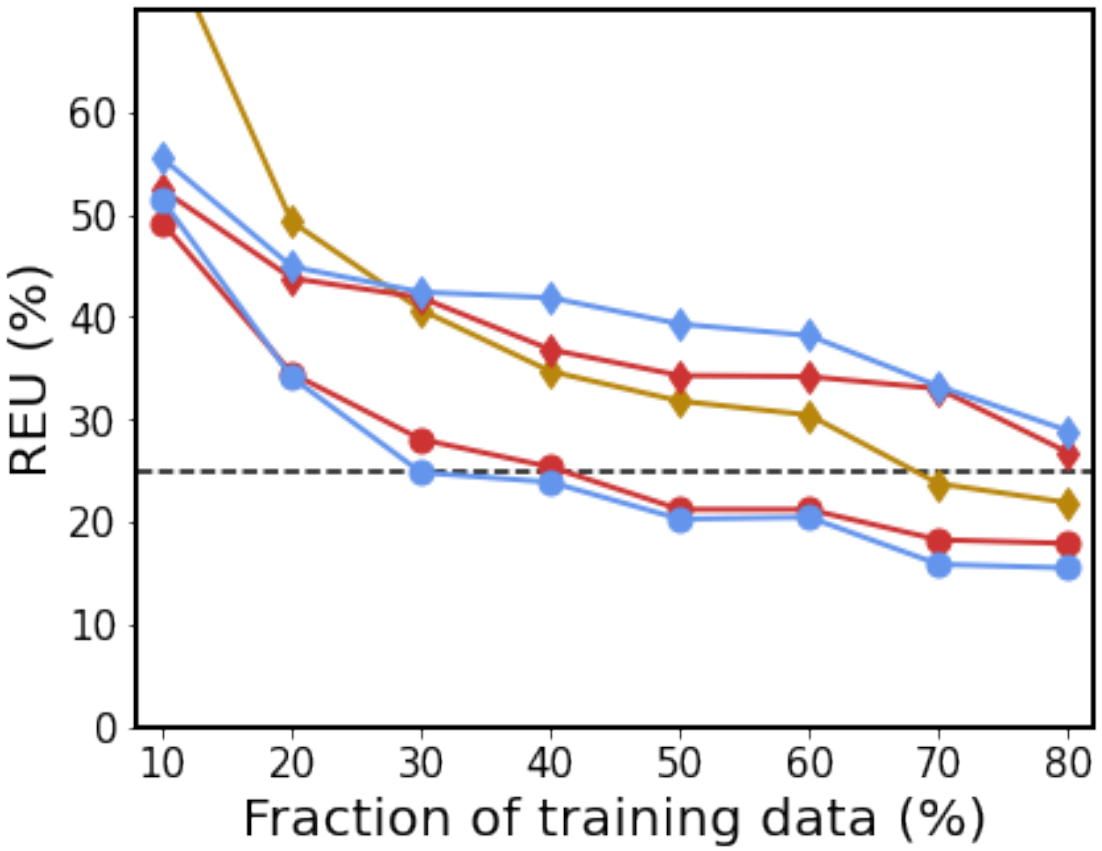
[26]:
```
import numpy as np
import matplotlib.pyplot as plt
A=[i for i in range(1,41)]
#Diff=[Diff[i] for i in range(16) if i%2==0]
#Diff2=[Diff2[i] for i in range(16) if i%2==0]
#Diff3=[Diff3[i] for i in range(16) if i%2==0]
Y_Test=[i for i in np.arange(0,41,1)]
fig= plt.figure(figsize=(6.5,5))
ax = fig.add_subplot(111)
#plt.scatter(A[1],Corr_mean[1], marker="d",s=200,color='darkgoldenrod', alpha=1)
#plt.scatter(A[1],Corr_mean2[1], marker="d",s=200,color='#CD5B45', alpha=0.9)
#plt.legend(['r','R^2'],loc = 2, bbox_to_anchor = (0.7,0.7), fontsize=16)



plt.plot(X,RMSE_1_NO2,color='darkgoldenrod',marker="d",markersize=9,linewidth=2.
 ↪5, alpha=1)
#plt.plot(X,RMSE_1_CO_2, color='darkgoldenrod',marker="o",markersize=9, alpha=1)
#plt.plot(X,RMSE_2_CO_2, color='teal',marker="d",markersize=9, alpha=1)
plt.plot(X,RMSE_3_NO2_C, color='#CD3333',marker="d",markersize=9,linewidth=2.5,␣
 ↪alpha=1)
plt.plot(X,RMSE_3_NO2,color='#CD3333',marker="o",markersize=9,linewidth=2.5,␣
 ↪alpha=1)
```

```
#plt.plot(X,RMSE_3_CO_2, color='#CD3333',marker="o",markersize=9, alpha=1)
plt.plot(X,RMSE_6_NO2_C, color='#6495ED',marker="d",markersize=9,linewidth=2.5,␣
 ↪alpha=1)
plt.plot(X,RMSE_6_NO2,color='#6495ED',marker="o",markersize=9,linewidth=2.5,␣
 ↪alpha=1)
#plt.plot(X,RMSE_6_CO_2, color='#6495ED',marker="o",markersize=9, alpha=1)

ax.set_ylim(bottom=0)
ax.set_xlim(left=0)
plt.xticks(fontsize=15)
plt.yticks(fontsize=15)
plt.show
#plt.xlim(0,40)
plt.ylim(0.05,0.51)
plt.xlim(8,82)
plt.yticks(np.arange(0.05,0.51, step=0.05))
#plt.xticks(np.arange(0,40, step=5))
#ax.set_xticks([0,5,10,15,20,25,30,35,40,45])
#ax.
 ↪set_xticklabels(['0','10','20','30','40','50','60','70','80','90'],fontsize=16)
plt.ylabel('NRMSE',fontsize=20)
plt.setp(ax.spines.values(), linewidth=2)
#plt.xlabel('Tolerance, Tc (%)',fontsize=20)
plt.xlabel('Fraction of training data (%)',fontsize=20)
#plt.text(33,2, 'Tc=2',fontsize=17)
#plt.text(33,4, 'Tc=4',fontsize=17)
#plt.text(33,6, 'Tc=6',fontsize=17)
#plt.text(33,8, 'Tc=8',fontsize=17)
#plt.text(33,10, 'Tc=10',fontsize=17)
#plt.title('CO'.translate(SUB),fontsize=16)
textstr = 'CO-'.translate(SUB) +'Monthly'
props = dict(boxstyle='round', facecolor='white', alpha=1)
#plt.text(0.717, 0.975, textstr, transform=ax.transAxes, fontsize=15,
        #verticalalignment='top', bbox=props)
plt.savefig("rmse_NO2_S.pdf",format="pdf", bbox_inches="tight",dpi=1000)
plt.show()
```



[27]:
```
X=[10,20,30,40,50,60,70,80]
SUB = str.maketrans("0123456789", "      ")
SUP = str.maketrans("0123456789", " ¹²³    ")
import numpy as np
import matplotlib.pyplot as plt
A=[i for i in range(1,41)]
#Diff=[Diff[i] for i in range(16) if i%2==0]
#Diff2=[Diff2[i] for i in range(16) if i%2==0]
#Diff3=[Diff3[i] for i in range(16) if i%2==0]
Y_Test=[i for i in np.arange(0,41,1)]
fig= plt.figure(figsize=(6.5,5))
ax = fig.add_subplot(111)
#plt.scatter(A[1],Corr_mean[1], marker="d",s=200,color='darkgoldenrod', alpha=1)
#plt.scatter(A[1],Corr_mean2[1], marker="d",s=200,color='#CD5B45', alpha=0.9)
#plt.legend(['r','R^2'],loc = 2, bbox_to_anchor = (0.7,0.7), fontsize=16)


plt.hlines([25], 0, 100, linestyles='dashed', color='black', linewidth=1.5)
plt.plot(X,REU_1_NO2,color='darkgoldenrod',marker="d",markersize=9,linewidth=2,␣
 ↪alpha=1)
```

```
#plt.plot(X,RMSE_1_CO_2, color='darkgoldenrod',marker="o",markersize=9, alpha=1)
#plt.plot(X,RMSE_2_CO_2, color='teal',marker="d",markersize=9, alpha=1)
plt.plot(X,REU_3_NO2_C, color='#CD3333',marker="d",markersize=9,linewidth=2,␣
 ↪alpha=1)
plt.plot(X,REU_3_NO2,color='#CD3333',marker="o",markersize=9,linewidth=2,␣
 ↪alpha=1)
#plt.plot(X,RMSE_3_CO_2, color='#CD3333',marker="o",markersize=9, alpha=1)
plt.plot(X,REU_6_NO2_C, color='#6495ED',marker="d",markersize=9,linewidth=2,␣
 ↪alpha=1)
plt.plot(X,REU_6_NO2,color='#6495ED',marker="o",markersize=9,linewidth=2,␣
 ↪alpha=1)
#plt.plot(X,RMSE_6_CO_2, color='#6495ED',marker="o",markersize=9, alpha=1)
ax.set_ylim(bottom=0)
ax.set_xlim(left=0)
plt.xticks(fontsize=15)
plt.yticks(fontsize=15)
plt.show
#plt.xlim(0,40)
plt.ylim(0,70)
plt.xlim(8,82)
plt.yticks(np.arange(0,70, step=10))
#plt.xticks(np.arange(0,40, step=5))
#ax.set_xticks([0,5,10,15,20,25,30,35,40,45])
#ax.
 ↪set_xticklabels(['0','10','20','30','40','50','60','70','80','90'],fontsize=16)
plt.ylabel('REU (%)',fontsize=20)
plt.setp(ax.spines.values(), linewidth=2)
#plt.xlabel('Tolerance, Tc (%)',fontsize=20)
plt.xlabel('Fraction of training data (%)',fontsize=20)
#plt.text(33,2, 'Tc=2',fontsize=17)
#plt.text(33,4, 'Tc=4',fontsize=17)
#plt.text(33,6, 'Tc=6',fontsize=17)
#plt.text(33,8, 'Tc=8',fontsize=17)
#plt.text(33,10, 'Tc=10',fontsize=17)
#plt.title('CO'.translate(SUB),fontsize=16)
textstr = 'CO-'.translate(SUB) +'Monthly'
props = dict(boxstyle='round', facecolor='white', alpha=1)
#plt.text(0.717, 0.975, textstr, transform=ax.transAxes, fontsize=15,
        #verticalalignment='top', bbox=props)
plt.savefig("reu_NO2_S.pdf",format="pdf", bbox_inches="tight",dpi=1000)
plt.show()
```



6 O3

[28]:
```
REU_6_O3_C=[53.56, 42.75, 39.17, 38.02, 29.96, 25.94, 26.06, 23.06]
REU_3_O3_C=[56.11, 47.25, 50.84, 45.93, 36.8, 39.52, 34.97, 28.45]
REU_6_O3=[36.72, 31.35, 21.46, 17.0, 14.71, 13.72, 12.57, 11.25]
REU_3_O3=[44.12, 32.62, 26.41, 22.37, 19.32, 17.81, 18.93, 11.63]
REU_1_O3=[57.14, 41.32, 37.36, 32.05, 28.26, 26.03, 26.75, 17.16]



RMSE_6_O3_C=[0.41, 0.35, 0.33, 0.32, 0.29, 0.28, 0.24, 0.23]
RMSE_3_O3_C=[0.46, 0.38, 0.37, 0.36, 0.34, 0.36, 0.31, 0.3]
RMSE_6_O3=[0.34, 0.27, 0.26, 0.24, 0.21, 0.2, 0.2, 0.18]
RMSE_3_O3=[0.35, 0.29, 0.28, 0.25, 0.24, 0.26, 0.23, 0.22]
RMSE_1_O3=[0.33, 0.31, 0.31, 0.27, 0.28, 0.29, 0.25, 0.22]

REU_1_O3_2=[62.85, 49.17, 42.96, 37.5 , 33.06, 29.67, 30.5 , 19.39]
REU_3_O3_2=[50.3 , 37.19, 30.37, 27.29, 23.38, 20.66, 22.72, 13.61]
REU_6_O3_2=[41.49, 29.41, 21.99, 19.04, 17.36, 16.63, 16.15, 12.49]
```

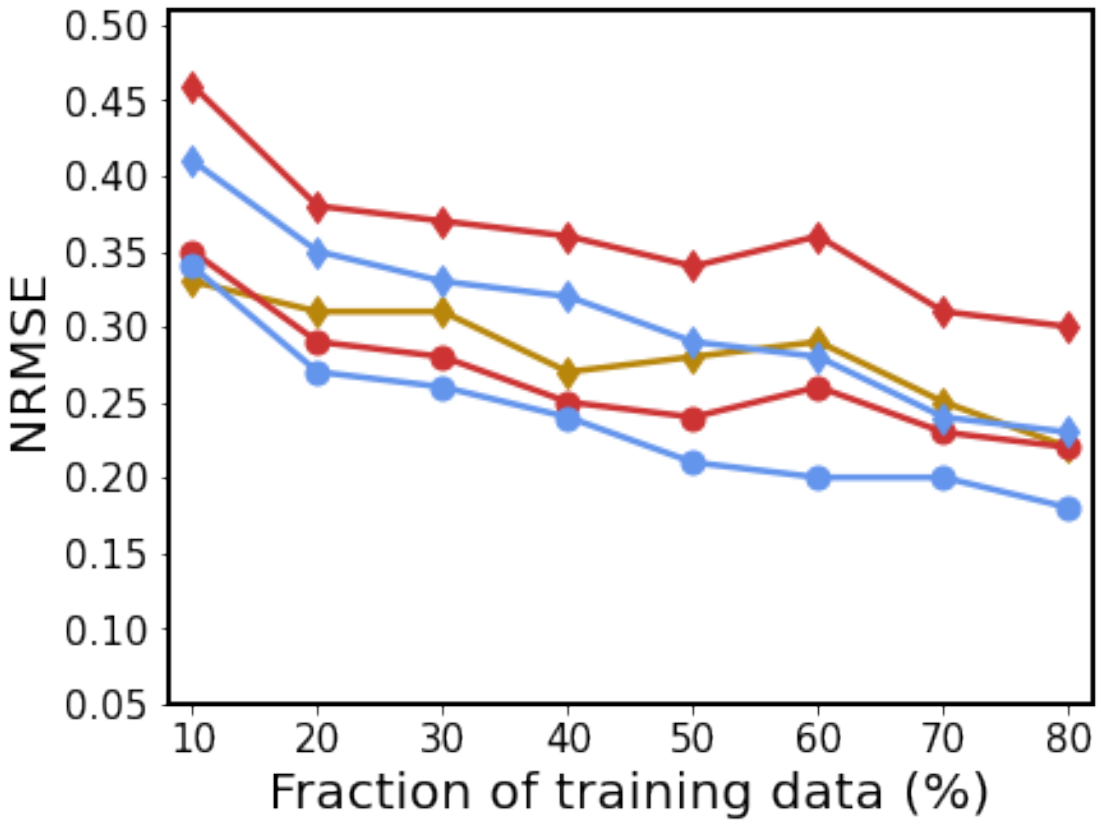
```
REU_3_O3_C_2=[65.65, 53.39, 56.94, 56.03, 40.85, 48.21, 38.82, 33.29]
REU_6_O3_C_2=[61.06, 52.16, 45.44, 42.58, 33.85, 30.61, 30.49, 26.29]

RMSE_1_O3_2=[0.39, 0.37, 0.36, 0.33, 0.31, 0.34, 0.29, 0.26]
RMSE_3_O3_2=[0.39, 0.35, 0.33, 0.3 , 0.29, 0.29, 0.25, 0.24]
RMSE_6_O3_2=[0.4 , 0.32, 0.31, 0.28, 0.24, 0.23, 0.24, 0.2 ]
RMSE_3_O3_C_2=[0.55, 0.43, 0.43, 0.44, 0.39, 0.42, 0.36, 0.34]
RMSE_6_O3_C_2=[0.49, 0.39, 0.4 , 0.39, 0.35, 0.34, 0.29, 0.27]
```

[29]:
```
np.mean(RMSE_6_O3_2)-np.mean(RMSE_6_O3)
```

[29]:
```
0.03999999999999995
```

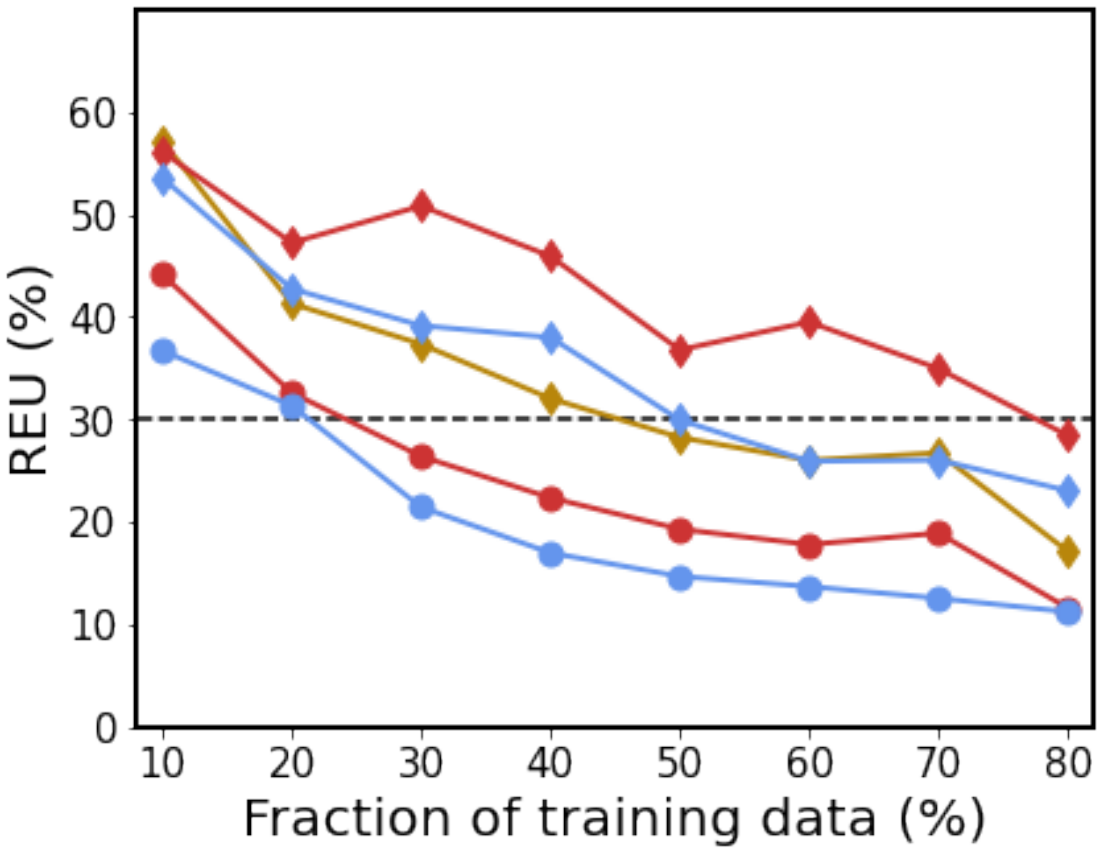
[30]:
```
import numpy as np
import matplotlib.pyplot as plt
A=[i for i in range(1,41)]
#Diff=[Diff[i] for i in range(16) if i%2==0]
#Diff2=[Diff2[i] for i in range(16) if i%2==0]
#Diff3=[Diff3[i] for i in range(16) if i%2==0]
Y_Test=[i for i in np.arange(0,41,1)]
fig= plt.figure(figsize=(6.5,5))
ax = fig.add_subplot(111)
#plt.scatter(A[1],Corr_mean[1], marker="d",s=200,color='darkgoldenrod', alpha=1)
#plt.scatter(A[1],Corr_mean2[1], marker="d",s=200,color='#CD5B45', alpha=0.9)
#plt.legend(['r','R^2'],loc = 2, bbox_to_anchor = (0.7,0.7), fontsize=16)



plt.plot(X,RMSE_1_O3,color='darkgoldenrod',marker="d",markersize=9,linewidth=2.
 ↪5, alpha=1)
#plt.plot(X,RMSE_1_CO_2, color='darkgoldenrod',marker="o",markersize=9, alpha=1)
#plt.plot(X,RMSE_2_CO_2, color='teal',marker="d",markersize=9, alpha=1)
plt.plot(X,RMSE_3_O3_C, color='#CD3333',marker="d",markersize=9,linewidth=2.5,␣
 ↪alpha=1)
plt.plot(X,RMSE_3_O3,color='#CD3333',marker="o",markersize=9,linewidth=2.5,␣
 ↪alpha=1)
#plt.plot(X,RMSE_3_CO_2, color='#CD3333',marker="o",markersize=9, alpha=1)
plt.plot(X,RMSE_6_O3_C, color='#6495ED',marker="d",markersize=9,linewidth=2.5,␣
 ↪alpha=1)
plt.plot(X,RMSE_6_O3,color='#6495ED',marker="o",markersize=9,linewidth=2.5,␣
 ↪alpha=1)
#plt.plot(X,RMSE_6_CO_2, color='#6495ED',marker="o",markersize=9, alpha=1)

ax.set_ylim(bottom=0)
ax.set_xlim(left=0)
plt.xticks(fontsize=15)
plt.yticks(fontsize=15)
```

```
plt.show
#plt.xlim(0,40)
plt.ylim(0.05,0.51)
plt.xlim(8,82)
plt.yticks(np.arange(0.05,0.51, step=0.05))
#plt.xticks(np.arange(0,40, step=5))
#ax.set_xticks([0,5,10,15,20,25,30,35,40,45])
#ax.
 ↪set_xticklabels(['0','10','20','30','40','50','60','70','80','90'],fontsize=16)
plt.ylabel('NRMSE',fontsize=20)
plt.setp(ax.spines.values(), linewidth=2)
#plt.xlabel('Tolerance, Tc (%)',fontsize=20)
plt.xlabel('Fraction of training data (%)',fontsize=20)
#plt.text(33,2, 'Tc=2',fontsize=17)
#plt.text(33,4, 'Tc=4',fontsize=17)
#plt.text(33,6, 'Tc=6',fontsize=17)
#plt.text(33,8, 'Tc=8',fontsize=17)
#plt.text(33,10, 'Tc=10',fontsize=17)
#plt.title('CO'.translate(SUB),fontsize=16)
textstr = 'CO-'.translate(SUB) +'Monthly'
props = dict(boxstyle='round', facecolor='white', alpha=1)
#plt.text(0.717, 0.975, textstr, transform=ax.transAxes, fontsize=15,
        #verticalalignment='top', bbox=props)
plt.savefig("rmse_O3_S.pdf",format="pdf", bbox_inches="tight",dpi=1000)
plt.show()
```



[31]:
```
X=[10,20,30,40,50,60,70,80]
SUB = str.maketrans("0123456789", "      ")
SUP = str.maketrans("0123456789", " ¹²³    ")
import numpy as np
import matplotlib.pyplot as plt
A=[i for i in range(1,41)]
#Diff=[Diff[i] for i in range(16) if i%2==0]
#Diff2=[Diff2[i] for i in range(16) if i%2==0]
#Diff3=[Diff3[i] for i in range(16) if i%2==0]
Y_Test=[i for i in np.arange(0,41,1)]
fig= plt.figure(figsize=(6.5,5))
ax = fig.add_subplot(111)
#plt.scatter(A[1],Corr_mean[1], marker="d",s=200,color='darkgoldenrod', alpha=1)
#plt.scatter(A[1],Corr_mean2[1], marker="d",s=200,color='#CD5B45', alpha=0.9)
#plt.legend(['r','R^2'],loc = 2, bbox_to_anchor = (0.7,0.7), fontsize=16)


plt.hlines([30], 0, 100, linestyles='dashed', color='black', linewidth=1.5)
plt.plot(X,REU_1_03,color='darkgoldenrod',marker="d",markersize=9,linewidth=2,␣
 ↪alpha=1)
```

```
#plt.plot(X,RMSE_1_CO_2, color='darkgoldenrod',marker="o",markersize=9, alpha=1)
#plt.plot(X,RMSE_2_CO_2, color='teal',marker="d",markersize=9, alpha=1)
plt.plot(X,REU_3_O3_C, color='#CD3333',marker="d",markersize=9,linewidth=2,␣
 ↪alpha=1)
plt.plot(X,REU_3_O3,color='#CD3333',marker="o",markersize=9,linewidth=2,␣
 ↪alpha=1)
#plt.plot(X,RMSE_3_CO_2, color='#CD3333',marker="o",markersize=9, alpha=1)
plt.plot(X,REU_6_O3_C, color='#6495ED',marker="d",markersize=9,linewidth=2,␣
 ↪alpha=1)
plt.plot(X,REU_6_O3,color='#6495ED',marker="o",markersize=9,linewidth=2,␣
 ↪alpha=1)
#plt.plot(X,RMSE_6_CO_2, color='#6495ED',marker="o",markersize=9, alpha=1)
ax.set_ylim(bottom=0)
ax.set_xlim(left=0)
plt.xticks(fontsize=15)
plt.yticks(fontsize=15)
plt.show
#plt.xlim(0,40)
plt.ylim(0,70)
plt.xlim(8,82)
plt.yticks(np.arange(0,70, step=10))
#plt.xticks(np.arange(0,40, step=5))
#ax.set_xticks([0,5,10,15,20,25,30,35,40,45])
#ax.
 ↪set_xticklabels(['0','10','20','30','40','50','60','70','80','90'],fontsize=16)
plt.ylabel('REU (%)',fontsize=20)
plt.setp(ax.spines.values(), linewidth=2)
#plt.xlabel('Tolerance, Tc (%)',fontsize=20)
plt.xlabel('Fraction of training data (%)',fontsize=20)
#plt.text(33,2, 'Tc=2',fontsize=17)
#plt.text(33,4, 'Tc=4',fontsize=17)
#plt.text(33,6, 'Tc=6',fontsize=17)
#plt.text(33,8, 'Tc=8',fontsize=17)
#plt.text(33,10, 'Tc=10',fontsize=17)
#plt.title('CO'.translate(SUB),fontsize=16)
textstr = 'CO-'.translate(SUB) +'Monthly'
props = dict(boxstyle='round', facecolor='white', alpha=1)
#plt.text(0.717, 0.975, textstr, transform=ax.transAxes, fontsize=15,
        #verticalalignment='top', bbox=props)
plt.savefig("reu_O3_S.pdf",format="pdf", bbox_inches="tight",dpi=1000)
plt.show()
```



[32]:
```
#!pip install -U kaleido
import plotly.graph_objects as go
from functools import reduce
from itertools import product

z=[list(np.ceil(np.array(RMSE_6_03_C_2)*100).astype(int)),list(np.ceil(np.
 ↪array(RMSE_6_03_C)*100).astype(int)),
   list(np.ceil(np.array(RMSE_3_03_C_2)*100).astype(int)),list(np.ceil(np.
 ↪array(RMSE_3_03_C)*100).astype(int)),
  list(np.ceil(np.array(RMSE_1_03_2)*100).astype(int)),list(np.ceil(np.
 ↪array(RMSE_1_03)*100).astype(int))]
x=['<b>10</b>', '<b>20</b>', '<b>30</b>', '<b>40</b>', '<b>50</b>','<b>60</b>',
   '<b>70</b>','<b>80</b>']
y=['<b>6 month (1h)</b>','<b>6 month (2min)</b>','<b>3 month (1h)</b>','<b>3␣
 ↪month (2min)</b>',
   '<b>1 month (1h)','<b>1 month (2min)</b>']

def get_anno_text(z_value):
    annotations=[]
```

```
    a, b = len(z_value), len(z_value[0])
    flat_z = reduce(lambda x,y: x+y, z_value) # z_value.flat if you deal with␣
 ↪numpy
    coords = product(range(a), range(b))
    for pos, elem in zip(coords, flat_z):
        annotations.append({'font': {'color': 'black'},
                    'showarrow': False,
                    'text': str(elem),
                    'x': pos[1],
                    'y': pos[0],
                        'font.size':24   })
    return annotations

fig = go.Figure(data=go.Heatmap(
                    z=z,
                    x=x,
                    y=y,
                    hoverongaps = True, colorscale ='turbid',
    opacity=0.6,colorbar=dict(tickfont=dict(size=24)) ))#matter#

fig.update_layout(title={'text': "",
        'y':0.8,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},
          plot_bgcolor='rgba(0,0,0,0)',
    annotations = get_anno_text(z),
                   width=1000,
height=400,xaxis={'side': 'top'},margin=dict(l=20, r=20, t=20, b=20))

fig.update_xaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.update_yaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.write_image("RMSE_O3_C_POLAR.pdf",engine="kaleido")
#plt.savefig("table2a.pdf", format="pdf", bbox_inches="tight")
fig.show()
```

[33]:
```
#!pip install -U kaleido
import plotly.graph_objects as go
from functools import reduce
from itertools import product

z=[list(np.ceil(np.array(RMSE_6_O3_2)*100).astype(int)),list(np.ceil(np.
 ↪array(RMSE_6_O3)*100).astype(int)),
```

```
   list(np.ceil(np.array(RMSE_3_O3_2)*100).astype(int)),list(np.ceil(np.
 ↪array(RMSE_3_O3)*100).astype(int)),
  list(np.ceil(np.array(RMSE_1_O3_2)*100).astype(int)),list(np.ceil(np.
 ↪array(RMSE_1_O3)*100).astype(int))]
x=['<b>10</b>', '<b>20</b>', '<b>30</b>', '<b>40</b>', '<b>50</b>','<b>60</b>',
   '<b>70</b>','<b>80</b>']
y=['<b>6 month (1h)</b>','<b>6 month (2min)</b>','<b>3 month (1h)</b>','<b>3␣
 ↪month (2min)</b>',
   '<b>1 month (1h)','<b>1 month (2min)</b>']

def get_anno_text(z_value):
    annotations=[]
    a, b = len(z_value), len(z_value[0])
    flat_z = reduce(lambda x,y: x+y, z_value) # z_value.flat if you deal with␣
 ↪numpy
    coords = product(range(a), range(b))
    for pos, elem in zip(coords, flat_z):
        annotations.append({'font': {'color': 'black'},
                    'showarrow': False,
                    'text': str(elem),
                    'x': pos[1],
                    'y': pos[0],
                        'font.size':24   })
    return annotations

fig = go.Figure(data=go.Heatmap(
                    z=z,
                    x=x,
                    y=y,
                    hoverongaps = True, colorscale ='turbid',
    opacity=0.6,colorbar=dict(tickfont=dict(size=24)) ))#matter#

fig.update_layout(title={'text': "",
        'y':0.8,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},
          plot_bgcolor='rgba(0,0,0,0)',
    annotations = get_anno_text(z),
                width=1000,
height=400,xaxis={'side': 'top'},margin=dict(l=20, r=20, t=20, b=20))

fig.update_xaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                mirror=True)
fig.update_yaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',
```

```
                  mirror=True)
fig.write_image("RMSE_O3_S_POLAR.pdf",engine="kaleido")
#plt.savefig("table2a.pdf", format="pdf", bbox_inches="tight")
fig.show()
```

[34]:
```
#!pip install -U kaleido
import plotly.graph_objects as go
from functools import reduce
from itertools import product

z=[list(np.ceil(np.array(REU_6_O3_C_2)).astype(int)),list(np.ceil(np.
 ↪array(REU_6_O3_C)).astype(int)),
   list(np.ceil(np.array(REU_3_O3_C_2)).astype(int)),list(np.ceil(np.
 ↪array(REU_3_O3_C)).astype(int)),
  list(np.ceil(np.array(REU_1_O3_2)).astype(int)),list(np.ceil(np.
 ↪array(REU_1_O3)).astype(int))]
x=['<b>10</b>', '<b>20</b>', '<b>30</b>', '<b>40</b>', '<b>50</b>','<b>60</b>',
   '<b>70</b>','<b>80</b>']
y=['<b>6 month (1h)</b>','<b>6 month (2min)</b>','<b>3 month (1h)</b>','<b>3␣
 ↪month (2min)</b>',
   '<b>1 month (1h)','<b>1 month (2min)</b>']

def get_anno_text(z_value):
    annotations=[]
    a, b = len(z_value), len(z_value[0])
    flat_z = reduce(lambda x,y: x+y, z_value) # z_value.flat if you deal with␣
 ↪numpy
    coords = product(range(a), range(b))
    for pos, elem in zip(coords, flat_z):
        annotations.append({'font': {'color': 'black'},
                    'showarrow': False,
                    'text': str(elem),
                    'x': pos[1],
                    'y': pos[0],
                        'font.size':24   })
    return annotations

fig = go.Figure(data=go.Heatmap(
                    z=z,
                    x=x,
                    y=y,
                    hoverongaps = True, colorscale ='turbid',
    opacity=0.6,colorbar=dict(tickfont=dict(size=24)) ))#matter#

fig.update_layout(title={'text': "",
        'y':0.8,
        'x':0.5,
```

```
        'xanchor': 'center',
        'yanchor': 'top'},
          plot_bgcolor='rgba(0,0,0,0)',
    annotations = get_anno_text(z),
                  width=1000,
height=400,xaxis={'side': 'top'},margin=dict(l=20, r=20, t=20, b=20))

fig.update_xaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.update_yaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.write_image("REU_03_C_POLAR.pdf",engine="kaleido")
#plt.savefig("table2a.pdf", format="pdf", bbox_inches="tight")
fig.show()
```

[35]:
```
#!pip install -U kaleido
import plotly.graph_objects as go
from functools import reduce
from itertools import product

z=[list(np.ceil(np.array(REU_6_03_2)).astype(int)),list(np.ceil(np.
 ↪array(REU_6_03)).astype(int)),
   list(np.ceil(np.array(REU_3_03_2)).astype(int)),list(np.ceil(np.
 ↪array(REU_3_03)).astype(int)),
  list(np.ceil(np.array(REU_1_03_2)).astype(int)),list(np.ceil(np.
 ↪array(REU_1_03)).astype(int))]
x=['<b>10</b>', '<b>20</b>', '<b>30</b>', '<b>40</b>', '<b>50</b>','<b>60</b>',
   '<b>70</b>','<b>80</b>']
y=['<b>6 month (1h)</b>','<b>6 month (2min)</b>','<b>3 month (1h)</b>','<b>3␣
 ↪month (2min)</b>',
   '<b>1 month (1h)','<b>1 month (2min)</b>']

def get_anno_text(z_value):
    annotations=[]
    a, b = len(z_value), len(z_value[0])
    flat_z = reduce(lambda x,y: x+y, z_value) # z_value.flat if you deal with␣
 ↪numpy
    coords = product(range(a), range(b))
    for pos, elem in zip(coords, flat_z):
        annotations.append({'font': {'color': 'black'},
                    'showarrow': False,
                    'text': str(elem),
                    'x': pos[1],
                    'y': pos[0],
                        'font.size':24   })
```

```
    return annotations

fig = go.Figure(data=go.Heatmap(
                    z=z,
                    x=x,
                    y=y,
                    hoverongaps = True, colorscale ='turbid',
    opacity=0.6,colorbar=dict(tickfont=dict(size=24)) ))#matter#

fig.update_layout(title={'text': "",
        'y':0.8,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},
           plot_bgcolor='rgba(0,0,0,0)',
    annotations = get_anno_text(z),
                  width=1000,
height=400,xaxis={'side': 'top'},margin=dict(l=20, r=20, t=20, b=20))

fig.update_xaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.update_yaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.write_image("REU_O3_S_POLAR.pdf",engine="kaleido")
#plt.savefig("table2a.pdf", format="pdf", bbox_inches="tight")
fig.show()
```

[ ]:

7 Resolution

[36]:
```
SUB = str.maketrans("0123456789", "      ")
SUP = str.maketrans("0123456789", " ¹²³    ")

#!pip install -U kaleido
import plotly.graph_objects as go
from functools import reduce
from itertools import product

z=[[4.9,4.3,5.9],[4.3,4.4,6.0],[4.6,3.4,4.9]]
x=['<b>CO</b>', '<b>NO2</b>'.translate(SUB), '<b>O3</b>'.translate(SUB)]
y=['<b>6 month</b>','<b>3 month </b>','<b>1 month</b>']

def get_anno_text(z_value):
```

```
    annotations=[]
    a, b = len(z_value), len(z_value[0])
    flat_z = reduce(lambda x,y: x+y, z_value) # z_value.flat if you deal with␣
 ↪numpy
    coords = product(range(a), range(b))
    for pos, elem in zip(coords, flat_z):
        annotations.append({'font': {'color': 'black'},
                    'showarrow': False,
                    'text': str(elem),
                    'x': pos[1],
                    'y': pos[0],
                        'font.size':30   })
    return annotations

fig = go.Figure(data=go.Heatmap(
                    z=z,
                    x=x,
                    y=y,
                    hoverongaps = True, colorscale ='turbid',
    opacity=0.7,colorbar=dict(tickfont=dict(size=24)), zmin=2,zmax=6))#matter#

fig.update_layout(title={'text': "",
        'y':0.8,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},
          plot_bgcolor='rgba(0,0,0,0)',
    annotations = get_anno_text(z),
                  width=1000,
height=400,xaxis={'side': 'top'},margin=dict(l=20, r=20, t=20, b=20))

fig.update_xaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.update_yaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.write_image("Resolution_RMSE1.pdf",engine="kaleido")
#plt.savefig("table2a.pdf", format="pdf", bbox_inches="tight")
fig.show()
```

[37]:
```
z=[[2.7,2.8,4.0],[3.9,3.4,4.0],[4.6,3.4,4.9]]
x=['<b>CO</b>', '<b>NO2</b>'.translate(SUB), '<b>O3</b>'.translate(SUB)]
y=['<b>6 month</b>','<b>3 month </b>','<b>1 month</b>']

def get_anno_text(z_value):
    annotations=[]
```

```
    a, b = len(z_value), len(z_value[0])
    flat_z = reduce(lambda x,y: x+y, z_value) # z_value.flat if you deal with␣
 ↪numpy
    coords = product(range(a), range(b))
    for pos, elem in zip(coords, flat_z):
        annotations.append({'font': {'color': 'black'},
                    'showarrow': False,
                    'text': str(elem),
                    'x': pos[1],
                    'y': pos[0],
                        'font.size':30   })
    return annotations

fig = go.Figure(data=go.Heatmap(
                    z=z,
                    x=x,
                    y=y,
                    hoverongaps = True, colorscale ='turbid',
    opacity=0.7,colorbar=dict(tickfont=dict(size=24)),zmin=2,zmax=6 ))#matter#

fig.update_layout(title={'text': "",
        'y':0.8,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},
          plot_bgcolor='rgba(0,0,0,0)',
    annotations = get_anno_text(z),
                   width=1000,
height=400,xaxis={'side': 'top'},margin=dict(l=20, r=20, t=20, b=20))

fig.update_xaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.update_yaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.write_image("Resolution_RMSE2.pdf",engine="kaleido")
#plt.savefig("table2a.pdf", format="pdf", bbox_inches="tight")
fig.show()
```

[38]:
```
SUB = str.maketrans("0123456789", "      ")
SUP = str.maketrans("0123456789", " ¹²³    ")

#!pip install -U kaleido
import plotly.graph_objects as go
from functools import reduce
from itertools import product
```

```
z=[[6.7,6.8,5.5],[6.5,6.1,6.7],[6.0,6.0,4.9]]
x=['<b>CO</b>', '<b>NO2</b>'.translate(SUB), '<b>O3</b>'.translate(SUB)]
y=['<b>6 month</b>','<b>3 month </b>','<b>1 month</b>']

def get_anno_text(z_value):
    annotations=[]
    a, b = len(z_value), len(z_value[0])
    flat_z = reduce(lambda x,y: x+y, z_value) # z_value.flat if you deal with␣
 ↪numpy
    coords = product(range(a), range(b))
    for pos, elem in zip(coords, flat_z):
        annotations.append({'font': {'color': 'black'},
                    'showarrow': False,
                    'text': str(elem),
                    'x': pos[1],
                    'y': pos[0],
                        'font.size':30   })
    return annotations

fig = go.Figure(data=go.Heatmap(
                    z=z,
                    x=x,
                    y=y,
                    hoverongaps = True, colorscale ='turbid',
    opacity=0.7,colorbar=dict(tickfont=dict(size=24)), zmin=2,zmax=8))#matter#

fig.update_layout(title={'text': "",
        'y':0.8,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},
           plot_bgcolor='rgba(0,0,0,0)',
    annotations = get_anno_text(z),
                   width=1000,
height=400,xaxis={'side': 'top'},margin=dict(l=20, r=20, t=20, b=20))

fig.update_xaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                   mirror=True)
fig.update_yaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                   mirror=True)
fig.write_image("Resolution_REU1.pdf",engine="kaleido")
#plt.savefig("table2a.pdf", format="pdf", bbox_inches="tight")
fig.show()
```

[39]:
```
SUB = str.maketrans("0123456789", "      ")
SUP = str.maketrans("0123456789", " ¹²³    ")

#!pip install -U kaleido
import plotly.graph_objects as go
from functools import reduce
from itertools import product

z=[[3.4,3.8,2.8],[3.7,4.5,4.0],[6.0,6.0,4.9]]
x=['<b>CO</b>', '<b>NO2</b>'.translate(SUB), '<b>O3</b>'.translate(SUB)]
y=['<b>6 month</b>','<b>3 month </b>','<b>1 month</b>']

def get_anno_text(z_value):
    annotations=[]
    a, b = len(z_value), len(z_value[0])
    flat_z = reduce(lambda x,y: x+y, z_value) # z_value.flat if you deal with␣
 ↪numpy
    coords = product(range(a), range(b))
    for pos, elem in zip(coords, flat_z):
        annotations.append({'font': {'color': 'black'},
                    'showarrow': False,
                    'text': str(elem),
                    'x': pos[1],
                    'y': pos[0],
                        'font.size':30   })
    return annotations

fig = go.Figure(data=go.Heatmap(
                    z=z,
                    x=x,
                    y=y,
                    hoverongaps = True, colorscale ='turbid',
    opacity=0.7,colorbar=dict(tickfont=dict(size=24)), zmin=2,zmax=8))#matter#

fig.update_layout(title={'text': "",
        'y':0.8,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},
           plot_bgcolor='rgba(0,0,0,0)',
    annotations = get_anno_text(z),
                  width=1000,
height=400,xaxis={'side': 'top'},margin=dict(l=20, r=20, t=20, b=20))

fig.update_xaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
```

```
fig.update_yaxes(tickfont = dict(size=24),linewidth=0.1, linecolor='black',

                 mirror=True)
fig.write_image("Resolution_REU2.pdf",engine="kaleido")
#plt.savefig("table2a.pdf", format="pdf", bbox_inches="tight")
fig.show()
```

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

SUB = str.maketrans("0123456789", "₀₁₂₃₄₅₆₇₈₉") SUP = str.maketrans("0123456789", "⁰¹²³⁴⁵⁶⁷⁸⁹") import numpy as np import matplotlib.pyplot as plt A=[i for i in range(1,41)] #Diff=[Diff[i] for i in range(16) if i%2==0] #Diff2=[Diff2[i] for i in range(16) if i%2==0] #Diff3=[Diff3[i] for i in range(16) if i%2==0] Y_Test=[i for i in np.arange(0,41,1)] fig= plt.figure(figsize=(6.5,5)) ax = fig.add_subplot(111) #plt.scatter(A[1],Corr_mean[1], marker="d",s=200,color='darkgoldenrod', alpha=1) #plt.scatter(A[1],Corr_mean2[1], marker="d",s=200,color='#CD5B45', alpha=0.9) #plt.legend(['r','R^2'],loc = 2, bbox_to_anchor = (0.7,0.7), fontsize=16)

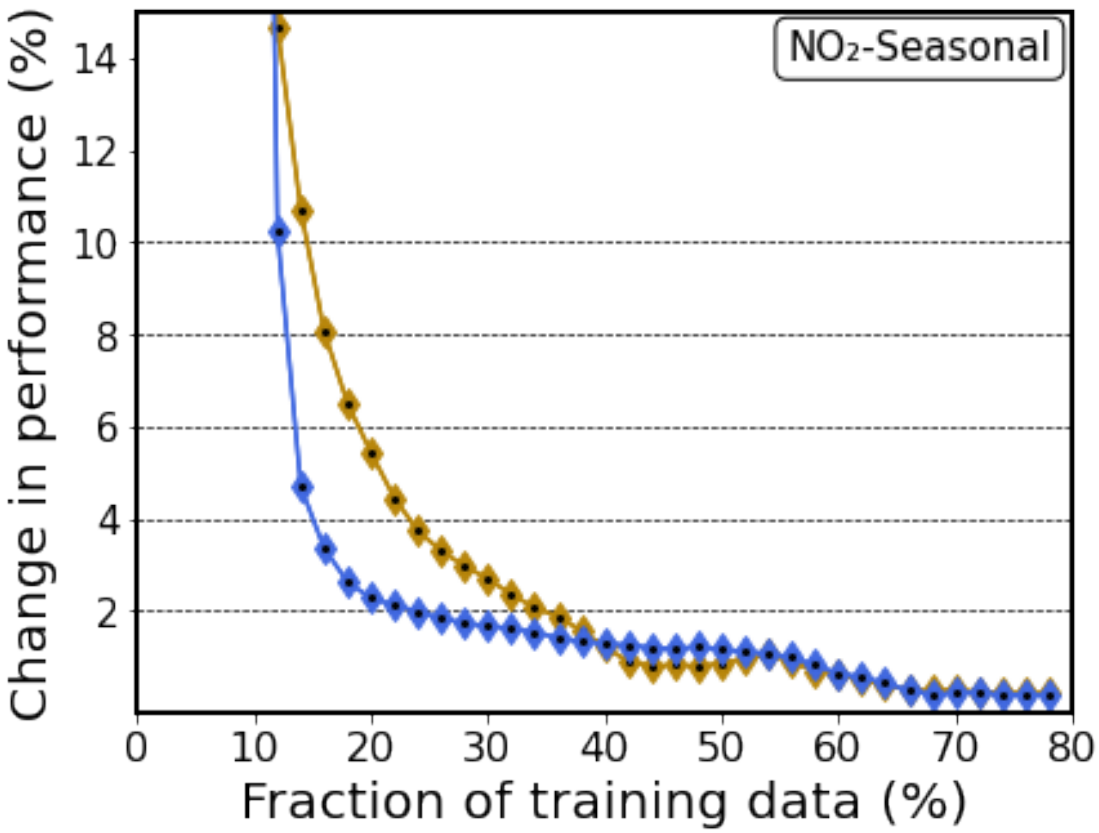
y1=[-0.5 for i in range(len(Y_Test))] y2=[2 for i in range(len(Y_Test))] y4=[4 for i in range(len(Y_Test))] y6=[6 for i in range(len(Y_Test))] y8=[8 for i in range(len(Y_Test))] y10=[10 for i in range(len(Y_Test))] plt.hlines([2], 0, 45, linestyles='dashed', color='black', linewidth=0.7) plt.hlines([4], 0, 45, linestyles='dashed', color='black', linewidth=0.7) plt.hlines([6], 0, 45, linestyles='dashed', color='black', linewidth=0.7) plt.hlines([8], 0, 45, linestyles='dashed', color='black', linewidth=0.7) plt.hlines([10], 0, 45, linestyles='dashed', color='black', linewidth=0.7) m1,=ax.plot(A[:-1],Diff_10_NO2_S[:-1], color='darkgoldenrod',marker="d",markersize=9, alpha=1) m2,=ax.plot(A[:-1],Diff_10_NO2_S[:-1], color='darkgoldenrod',marker="o",markersize=4,markerfacecolor='black', alpha=1) m3,=ax.plot(A[:-1],Diff_30_NO2_S[:-1], color='royalblue',marker="d",markersize=9, alpha=1) m4,=ax.plot(A[:-1],Diff_30_NO2_S[:-1], color='royalblue',marker="o",markersize=4,markerfacecolor= alpha=1) m5,=ax.plot(A[:-1],Diff_60_NO2_S[:-1], color='salmon',marker="d",markersize=9, alpha=1) m6,=ax.plot(A[:-1],Diff_60_NO2_S[:-1], color='salmon',marker="o",markersize=4,markerfacecolor='bla alpha=1) #plt.plot(Diff2[:-1],A[:-1], color='#CD5B45',marker="d",markersize=13, alpha=0.9) #m3,=ax.plot(A[:-1],Diff3[:-1], color='teal',marker="d",markersize=9, alpha=1) #m4,=ax.plot(A[:-1],Diff3[:-1], color='teal',marker="o",markersize=4,markerfacecolor='black', alpha=1) ax.set_ylim(bottom=0) ax.set_xlim(left=0) plt.xticks(fontsize=15) plt.yticks(fontsize=15) plt.show plt.xlim(0,40) plt.ylim(0,15) ax.set_xlim(right=40) ax.set_ylim(bottom=-0.2) plt.yticks(np.arange(0,15, step=2)) plt.xticks(np.arange(0,40, step=5)) ax.set_xticks([0,5,10,15,20,25,30,35,40]) ax.set_xticklabels(['0','10','20','30','40','50','60','70','80'],fontsize=16) plt.ylabel('Change in performance (%)',fontsize=20) plt.yticks([2,4,6,8,10,12,14]) plt.setp(ax.spines.values(), linewidth=2) #plt.xlabel('Tolerance, Tc (%)',fontsize=20) plt.xlabel('Fraction of training data (%)',fontsize=20) #plt.text(0.2,2, 'Tc=2',fontsize=14)

#plt.text(0.2,4, 'Tc=4',fontsize=14) #plt.text(0.2,6, 'Tc=6',fontsize=14) #plt.text(0.2,8, 'Tc=8',fontsize=14) #plt.text(0.2,10, 'Tc=10',fontsize=14) textstr = 'NO2-'.translate(SUB) +'Seasonal' props = dict(boxstyle='round', facecolor='white', alpha=1) plt.text(0.695, 0.975, textstr, transform=ax.transAxes, fontsize=15, verticalalignment='top', bbox=props) plt.savefig("CS_NO2_S4.pdf",format="pdf", bbox_inches="tight",dpi=1000) plt.show()

[40]:
```
RF=[61.73, 52.9 , 45.91, 42.7 , 41.37, 14.68, 10.7 ,  8.06,  6.5 ,
       5.44,  4.45,  3.75,  3.3 ,  2.98,  2.7 ,  2.36,  2.08,  1.87,
       1.58,  1.22,  0.91,  0.81,  0.83,  0.81,  0.86,  0.97,  1.05,
       0.9 ,  0.7 ,  0.61,  0.5 ,  0.4 ,  0.28,  0.27,  0.29,  0.25,
       0.22,  0.22,  0.22,  0.21]
LR=[77.45, 51.76, 47.91, 45.53, 44.29, 10.23,  4.69,  3.36,  2.63,
       2.28,  2.13,  1.99,  1.84,  1.74,  1.68,  1.63,  1.54,  1.42,
       1.34,  1.3 ,  1.26,  1.21,  1.19,  1.22,  1.16,  1.12,  1.08,
       1.  ,  0.83,  0.65,  0.57,  0.43,  0.28,  0.19,  0.22,  0.23,
       0.19,  0.19,  0.15,  0.14]
LR2=[77.446, 51.763, 47.908, 45.531, 44.288, 10.228,  4.692,  3.361,
       2.63 ,  2.278,  2.13 ,  1.993,  1.84 ,  1.738,  1.677,  1.628,
       1.541,  1.419,  1.342,  1.298,  1.264,  1.208,  1.191,  1.223,
       1.164,  1.119,  1.084,  0.996,  0.83 ,  0.647,  0.569,  0.432,
       0.281,  0.188,  0.218,  0.231,  0.191,  0.188,  0.154,  0.14 ]
```

[41]:
```
SUB = str.maketrans("0123456789", "      ")
SUP = str.maketrans("0123456789", " ¹²³    ")
import numpy as np
import matplotlib.pyplot as plt
A=[i for i in range(1,41)]
#Diff=[Diff[i] for i in range(16) if i%2==0]
#Diff2=[Diff2[i] for i in range(16) if i%2==0]
#Diff3=[Diff3[i] for i in range(16) if i%2==0]
Y_Test=[i for i in np.arange(0,41,1)]
fig= plt.figure(figsize=(6.5,5))
ax = fig.add_subplot(111)
#plt.scatter(A[1],Corr_mean[1], marker="d",s=200,color='darkgoldenrod', alpha=1)
#plt.scatter(A[1],Corr_mean2[1], marker="d",s=200,color='#CD5B45', alpha=0.9)
#plt.legend(['r','R^2'],loc = 2, bbox_to_anchor = (0.7,0.7), fontsize=16)

y1=[-0.5 for i in range(len(Y_Test))]
y2=[2 for i in range(len(Y_Test))]
y4=[4 for i in range(len(Y_Test))]
y6=[6 for i in range(len(Y_Test))]
y8=[8 for i in range(len(Y_Test))]
y10=[10 for i in range(len(Y_Test))]
plt.hlines([2], 0, 45, linestyles='dashed', color='black', linewidth=0.7)
plt.hlines([4], 0, 45, linestyles='dashed', color='black', linewidth=0.7)
plt.hlines([6], 0, 45, linestyles='dashed', color='black', linewidth=0.7)
plt.hlines([8], 0, 45, linestyles='dashed', color='black', linewidth=0.7)
```

```
plt.hlines([10], 0, 45, linestyles='dashed', color='black', linewidth=0.7)
m1,=ax.plot(A[:-1],RF[:-1], color='darkgoldenrod',marker="d",markersize=9,␣
 ↪alpha=1)
m2,=ax.plot(A[:-1],RF[:-1], color='darkgoldenrod',marker="o",markersize=4,
            markerfacecolor='black', alpha=1)
m3,=ax.plot(A[:-1],LR2[:-1], color='royalblue',marker="d",markersize=9, alpha=1)
m4,=ax.plot(A[:-1],LR2[:-1], color='royalblue',marker="o",markersize=4,
            markerfacecolor='black', alpha=1)
#m5,=ax.plot(A[:-1],Diff_60_NO2_S[:-1], color='salmon',marker="d",markersize=9,␣
 ↪alpha=1)
#m6,=ax.plot(A[:-1],Diff_60_NO2_S[:-1], color='salmon',marker="o",markersize=4
#plt.plot(Diff2[:-1],A[:-1], color='#CD5B45',marker="d",markersize=13, alpha=0.
 ↪9)
#m3,=ax.plot(A[:-1],Diff3[:-1], color='teal',marker="d",markersize=9, alpha=1)
#m4,=ax.plot(A[:-1],Diff3[:-1], color='teal',marker="o",markersize=4
ax.set_ylim(bottom=0)
ax.set_xlim(left=0)
plt.xticks(fontsize=15)
plt.yticks(fontsize=15)
plt.show
plt.xlim(0,40)
plt.ylim(0,15)
ax.set_xlim(right=40)
ax.set_ylim(bottom=-0.2)
plt.yticks(np.arange(0,15, step=2))
plt.xticks(np.arange(0,40, step=5))
ax.set_xticks([0,5,10,15,20,25,30,35,40])
ax.set_xticklabels(['0','10','20','30','40','50','60','70','80'],fontsize=16)
plt.ylabel('Change in performance (%)',fontsize=20)
plt.yticks([2,4,6,8,10,12,14])
plt.setp(ax.spines.values(), linewidth=2)
#plt.xlabel('Tolerance, Tc (%)',fontsize=20)
plt.xlabel('Fraction of training data (%)',fontsize=20)
#plt.text(0.2,2, 'Tc=2',fontsize=14)
#plt.text(0.2,4, 'Tc=4',fontsize=14)
#plt.text(0.2,6, 'Tc=6',fontsize=14)
#plt.text(0.2,8, 'Tc=8',fontsize=14)
#plt.text(0.2,10, 'Tc=10',fontsize=14)
textstr = 'NO2-'.translate(SUB) +'Seasonal'
props = dict(boxstyle='round', facecolor='white', alpha=1)
plt.text(0.695, 0.975, textstr, transform=ax.transAxes, fontsize=15,
        verticalalignment='top', bbox=props)
plt.savefig("AL.pdf",format="pdf", bbox_inches="tight",dpi=1000)
plt.show()
```



8 Bias and Precision

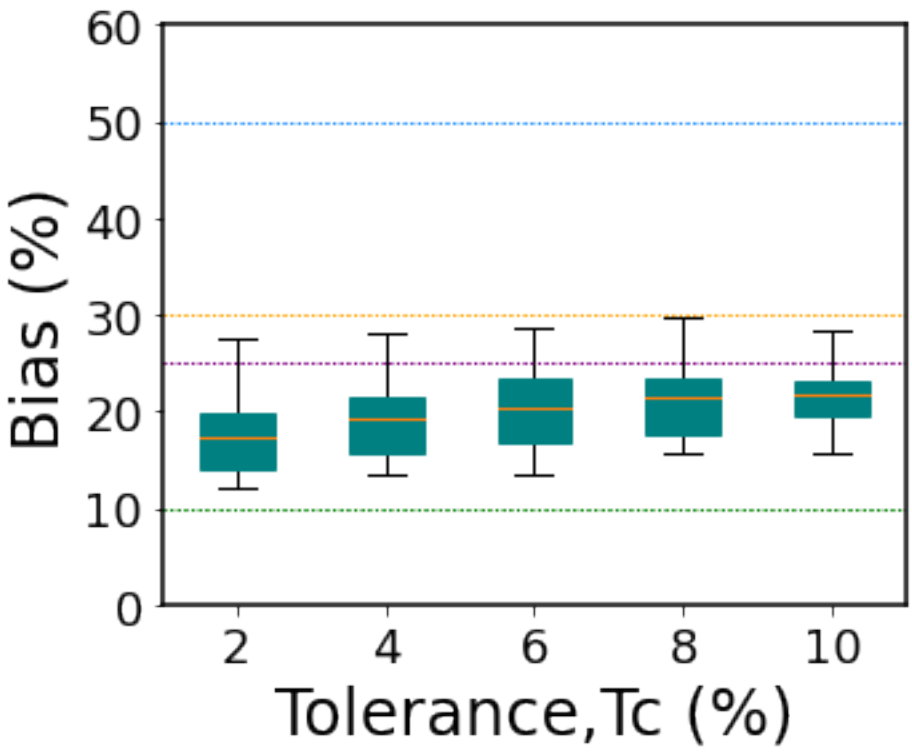
9 CO Seasonal

[ ]:

[42]:
```
B_2_15_S=[13.94, 12.72, 12.1 , 20.24, 18.83, 18.81, 16.46, 14.01, 13.41]
B_4_15_S=[15.2 , 14.22, 13.4 , 21.59, 20.26, 19.49, 17.91, 17.1 , 15.95]
B_6_15_S=[16.  , 14.53, 14.08, 22.69, 21.45, 20.59, 23.57, 20.24, 18.46]
B_8_15_S=[17.11, 16.43, 15.67, 25.1 , 23.28, 21.8 , 22.51, 21.72, 19.32]
B_10_15_S=[17.11, 16.43, 15.67, 25.1 , 23.28, 21.8 , 22.51, 21.72, 19.32]
B_2_60_S=[14.8 , 15.36, 13.33, 20.72, 18.72, 18.38, 27.6 , 22.17, 21.16]
B_4_60_S=[18.76, 15.41, 14.67, 23.09, 21.59, 19.59, 28.08, 22.22, 21.15]
B_6_60_S=[18.15, 16.23, 13.36, 24.  , 21.16, 19.75, 28.73, 26.04, 25.07]
B_8_60_S=[20.12, 16.22, 15.55, 23.36, 21.38, 20.72, 33.09, 29.62, 26.51]
B_10_60_S=[21.56, 20.73, 17.23, 22.97, 23.19, 21.85, 27.35, 28.46, 26.69]
import numpy as np
Data_B_15=[np.concatenate((np.array(B_2_15_S),np.array(B_2_60_S)),axis = 0),np.
 ↪concatenate((np.array(B_4_15_S),np.array(B_4_60_S)),axis = 0)
```

```
          ,np.concatenate((np.array(B_6_15_S),np.array(B_6_60_S)),axis = 0),np.
 ↪concatenate((np.array(B_8_15_S),np.array(B_8_60_S)),axis = 0)
          ,np.concatenate((np.array(B_10_15_S),np.array(B_10_60_S)),axis = 0)]
#Data_B_60=[B_2_60_S,B_4_60_S,B_6_60_S,B_8_60_S,B_10_60_S]
import matplotlib.pyplot as plt
import numpy as np
 # Creating dataset
np.random.seed(10)

fig = plt.figure(figsize =(5, 4))
ax = fig.add_subplot(111)
data=Data_B_15
data2=Data_B_15
#data2=data2
# Creating axes instance
plt.axhline(y= 10, color = 'green',label='RC', linestyle = ':',linewidth=1 )
plt.axhline(y= 25, color = 'purple',label='RC', linestyle = ':',linewidth=1 )
plt.axhline(y= 30, color = 'orange',label='RC', linestyle = ':',linewidth=1 )
plt.axhline(y= 50, color = 'dodgerblue',label='RC', linestyle = ':',linewidth=1␣
 ↪)
#plt.legend(['RC','SGS','IS','HA & CSP'],ncol=4, loc='lower left', fontsize=10)

bp = plt.boxplot(data, patch_artist = True,
                 vert = 1,showfliers=False)
#bp2 = ax.boxplot(data2, patch_artist = True,
                 #vert = 1)
#plt.legend(['Randomized','Non-randomized'],loc = 2, bbox_to_anchor = (0.62,1),␣
 ↪ fontsize=10)
colors= ['teal' for i in range(5)]
colors2= ['salmon' for i in range(5)]
#colors2= ['teal' for i in range(40)]

for patch, color in zip(bp['boxes'], colors):
    patch.set_color(color)

#for patch, color in zip(bp2['boxes'], colors2):
    #patch.set_color(color)

#plt.legend(['Randomized','Non-randomized'],loc = 2, bbox_to_anchor = (0.78,1),␣
 ↪ fontsize=10)

#plt.xlabel('Training Data (%)',fontsize=20)
plt.ylabel('Bias (%)',fontsize=24)
plt.xlabel('Tolerance,Tc (%)',fontsize=24)
plt.xticks(fontsize=18)
plt.yticks(fontsize=18)
#plt.tick_params(width=1,length=3)
```

```
#plt.xticks(np.arange(0,8 , step=1))
plt.yticks(np.arange(0,61, step=10))
#ax.spines["bottom"].set_linewidth(1)
#ax.spines["left"].set_linewidth(1)
#ax.spines["top"].set_linewidth(1)
#ax.spines["right"].set_linewidth(1)
plt.axhline(y= 10, color = 'green',label='RC', linestyle = ':',linewidth=1 )
plt.axhline(y= 25, color = 'purple',label='RC', linestyle = ':',linewidth=1 )
plt.axhline(y= 30, color = 'orange',label='RC', linestyle = ':',linewidth=1 )
plt.axhline(y= 50, color = 'dodgerblue',label='RC', linestyle = ':',linewidth=1
 ↪)
plt.grid(linestyle='-.',linewidth=0)
ax.set_xticks([1,2,3,4,5])
ax.set_xticklabels(['2','4','6','8','10'])
#plt.legend(['Bias','Precision'] ,fontsize=16)
#plt.title(r"$CO$",fontsize=16 )
plt.setp(ax.spines.values(), linewidth=1.5)
plt.savefig("B_S_CO.pdf",format="pdf",bbox_inches="tight",dpi=1000)
plt.show()
```



[43]:
```
P_2_15_S=[14.93, 13.98, 13.98, 21.6 , 21.17, 21.99, 24.22, 20.69, 20.21]
P_4_15_S=[16.14, 15.52, 14.96, 23.81, 23.17, 22.48, 23.04, 23.76, 22.51]
```

```
P_6_15_S=[16.65, 15.55, 15.94, 22.27, 21.52, 21.83, 30.43, 26.38, 25.28]
P_8_15_S=[18.17, 17.99, 17.12, 26.12, 24.79, 23.14, 28.09, 28.91, 27.38]
P_10_15_S=[18.17, 17.99, 17.12, 26.12, 24.79, 23.14, 28.09, 28.91, 27.38]
P_2_60_S=[14.98, 15.18, 13.57, 22.39, 19.46, 20.68, 29.4 , 26.08, 26.7 ]
P_4_60_S=[19.06, 16.88, 15.6 , 24.95, 23.5 , 23.21, 29.41, 23.31, 21.8 ]
P_6_60_S=[16.83, 15.33, 13.35, 25.85, 23.11, 21.91, 37.08, 33.47, 32.09]
P_8_60_S=[19.72, 16.07, 16.05, 21.98, 20.2 , 20.57, 43.39, 38.84, 32.7 ]
P_10_60_S=[23.11, 22.5 , 18.48, 23.77, 24.86, 23.23, 29.34, 32.91, 32.54]
```

[ ]:

[ ]: