Findings 10/04

Understanding the effect of weather on simulation results

Intepolate between clear-sunset and rain-sunset

- cloudiness
- precipitation
- precipitation_deposits
- wetness
- fog_density
- wind_intensity

In [1]:

```
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
# %matplotlib inline
import scipy.stats
from scipy.stats import norm, binom, poisson
from dtaidistance import dtw

import json
```

Layered folders, parse into 2d lists of dicts

In [2]:

```
import os

txt_lists = [[], [], [], [], [], []]

for root, dirs, files in os.walk("./campaign_results_new/route_highway_epoch24_clear-s
    for file in files:
        if file.endswith(".txt"):
            with open(os.path.join(root, file), encoding = 'utf-8') as f:
               read_string = f.read()
               json_object = json.loads(read_string)
               txt_lists[0].append(json_object)

for root, dirs, files in os.walk("./campaign_results_new/route_highway_epoch24_rain-su
    for file in files:
        if file.endswith(".txt"):
            with open(os.path.join(root, file), encoding = 'utf-8') as f:
               read_string = f.read()
               json_object = json.loads(read_string)
               txt_lists[1].append(json_object)

for root, dirs, files in os.walk("./campaign_results_new/route_highway_epoch24_rain-su
    for file in files:
        if file.endswith(".txt"):
            with open(os.path.join(root, file), encoding = 'utf-8') as f:
               read_string = f.read()
               json_object = json.loads(read_string)
```

```
                txt_lists[2].append(json_object)

for root, dirs, files in os.walk("./campaign_results_new/route_highway_epoch24_rain-su
    for file in files:
        if file.endswith(".txt"):
            with open(os.path.join(root, file), encoding = 'utf-8') as f:
                read_string = f.read()
                json_object = json.loads(read_string)
                txt_lists[3].append(json_object)

for root, dirs, files in os.walk("./campaign_results_new/route_highway_epoch24_rain-su
    for file in files:
        if file.endswith(".txt"):
            with open(os.path.join(root, file), encoding = 'utf-8') as f:
                read_string = f.read()
                json_object = json.loads(read_string)
                txt_lists[4].append(json_object)

for root, dirs, files in os.walk("./campaign_results_new/route_highway_epoch24_rain-su
    for file in files:
        if file.endswith(".txt"):
            with open(os.path.join(root, file), encoding = 'utf-8') as f:
                read_string = f.read()
                json_object = json.loads(read_string)
                txt_lists[5].append(json_object)
```

Examining results: No crashes, all perfect score

In [3]:

```
count_array = []
for txt_list in txt_lists:
    count = 0
    for txt in txt_list:
        if txt['_checkpoint']['records'][0]['status'] == 'Completed':
            count += 1
    count_array.append(count/100)

print(count_array)
```

```
[1.0, 1.0, 1.0, 1.0, 1.0, 1.0]
```

In [4]:

```
count_array = []
for txt_list in txt_lists:
    count = 0
    for txt in txt_list:
        count += txt['_checkpoint']['records'][0]['scores']['score_route']
    count_array.append(count/100)

print(count_array)
```

```
[100.0, 100.0, 100.0, 100.0, 100.0, 100.0]
```

Setting up into 2d lists of DataFrames

In [5]:

```
dim = (6, 100)
df_array = np.ndarray(dim, dtype=object)
```

In [6]:

```
dir_path = './campaign_results_new'
```

```python
# list to store files
res = []
count = 0

# Iterate directory
for path in os.listdir(dir_path):
    # check if current path is a file

#     print(path)
    if not os.path.isfile(os.path.join(dir_path, path)):
        folder_list = []
        for folder in os.listdir(os.path.join(dir_path, path)):
            folder_list.append(folder)
        folder_list.sort()
#         print(folder_list)
        for i in range(len(folder_list)):
            folder = folder_list[i]
            temp = os.path.join(os.path.join(dir_path, path), folder)
            file_array = []
            for file in os.listdir(temp):
                file_array.append(file)
            file_array.sort()
            df = pd.DataFrame()
            for file in file_array:
                if "_ctl.csv" in file:
                    df = pd.read_csv(temp + '/' + file)
                elif "_cvip.csv" in file:
                    df = pd.concat([df, pd.read_csv(temp + '/' + file)], axis=1)
                elif "_traj.csv" in file:
                    df = pd.concat([df, pd.read_csv(temp + '/' + file)], axis=1)
            df_array[count][i%100]=df
    count += 1
```

In [7]:

```python
df_array[5][50]
```

Out[7]:

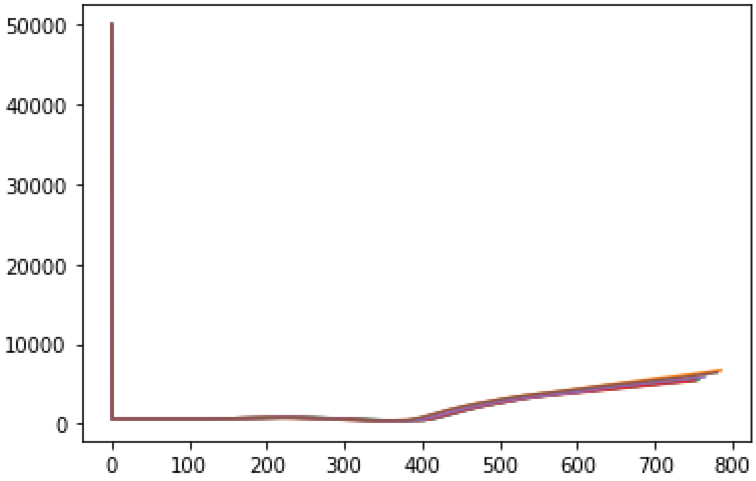
	ts	agent_id	throttle	steer	brake	ts	agent_id	cvip	cvip_x	cvi
0	567030	0	0.900000	0.015107	0.0	567030	0	500.491189	198.767441	-95.832
1	567031	0	0.900000	-0.004060	0.0	567031	0	5.595580	195.567444	-90.832
2	567032	0	0.900000	0.014287	0.0	567032	0	5.592365	195.567444	-90.832
3	567033	0	0.900000	0.002533	0.0	567033	0	5.589578	195.567444	-90.832
4	567034	0	0.900000	0.013922	0.0	567034	0	5.587154	195.567444	-90.832
...	...	...	...	...	...	...	...	...	...	
780	567810	0	0.900000	-0.000011	0.0	567810	0	64.163853	192.164413	107.317
781	567811	0	0.000000	0.000289	1.0	567811	0	64.301973	192.135345	107.649
782	567812	0	0.512673	-0.000414	0.0	567812	0	64.439160	192.105118	107.982
783	567813	0	0.900000	-0.001558	0.0	567813	0	64.575262	192.073746	108.315
784	567814	0	0.900000	0.000048	0.0	567814	0	64.710321	192.040955	108.648

785 rows × 17 columns

Since no accident, check cvip

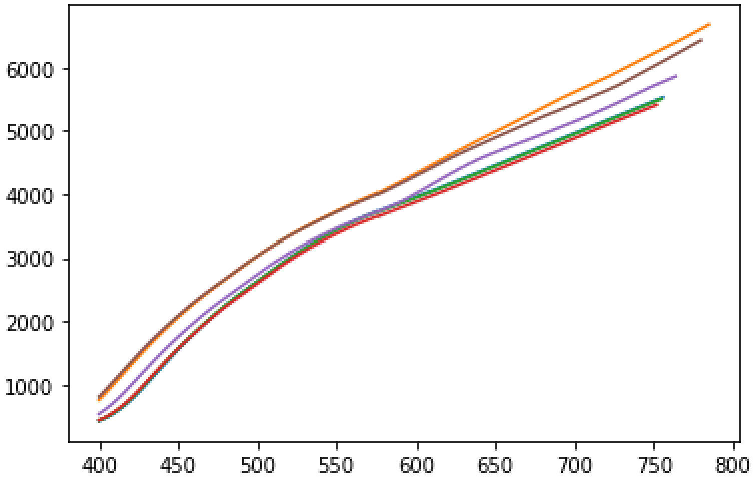
In [8]:

```
for weather in range(6):
    df_avg = df_array[weather][0]['cvip']
    count = 0
    for i in range(1, len(df_array[weather])):
        if not df_array[weather][i]['cvip'].isnull().values.any():
            df_avg += df_array[weather][i]['cvip']
            count += 1
    df_avg.interpolate().dropna()/count
    df_avg.plot()
```



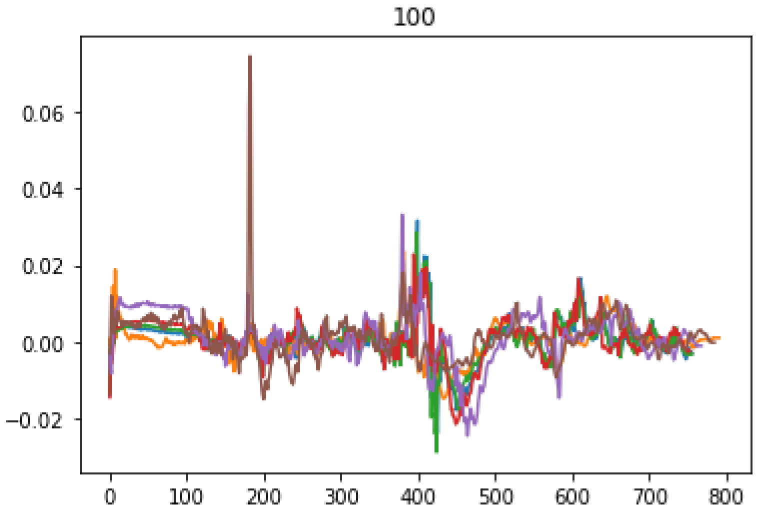
In [9]:

```
for weather in range(6):
    df_avg = df_array[weather][0]['cvip']
    count = 0
    for i in range(1, len(df_array[weather])):
        if not df_array[weather][i]['cvip'].isnull().values.any():
            df_avg += df_array[weather][i]['cvip']
            count += 1
    df_avg.interpolate().dropna()/count
    df_avg = df_avg[400:]
    df_avg.plot()
```



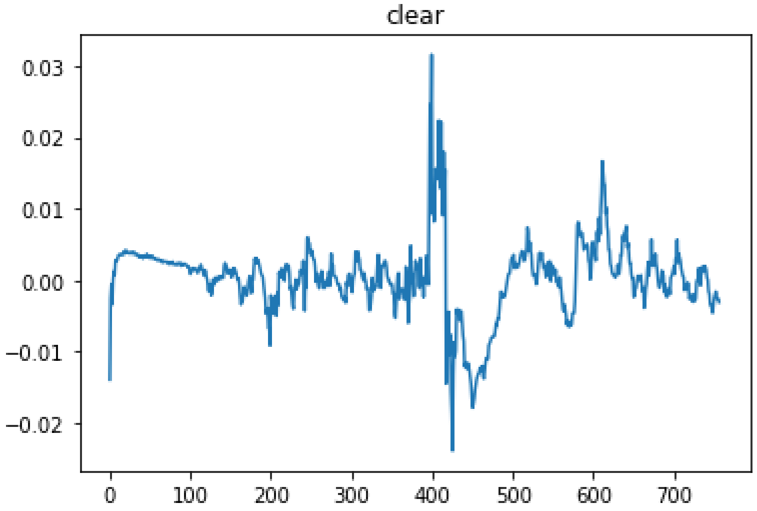
In [10]:

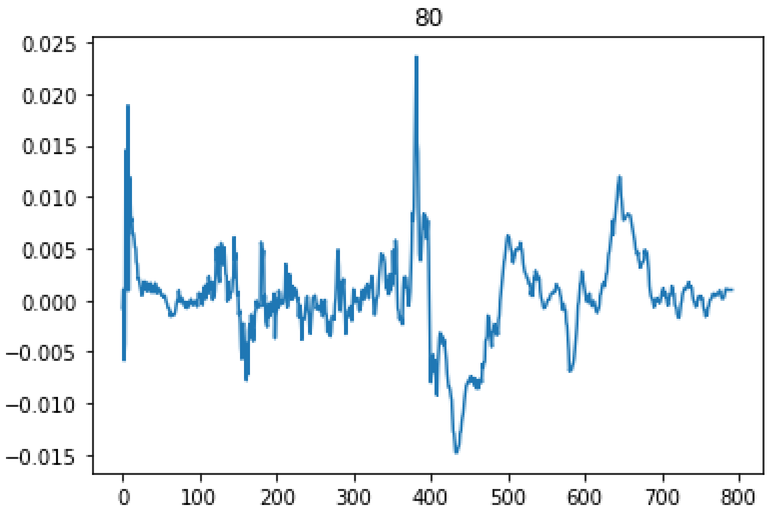
```
legend = ['clear', '80', '60', '20', '40', '100']
df_avg_dict = {}
for weather in range(6):
    df_avg = df_array[weather][0]['steer']
    for i in range(1, len(df_array[weather])):
        df_avg += df_array[weather][i]['steer']
    df_avg = df_avg.interpolate().dropna()/100
    df_avg_dict.update({legend[weather]: df_avg.copy()})
    df_avg.plot()
plt.title(legend[weather])
plt.show()
```

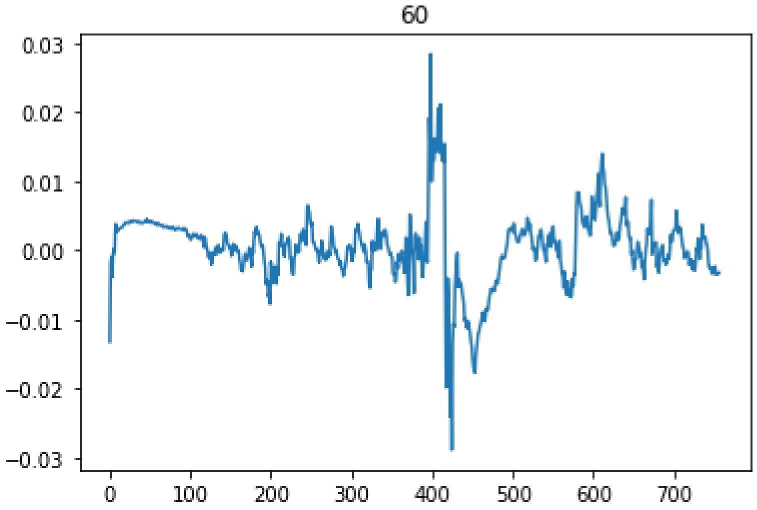


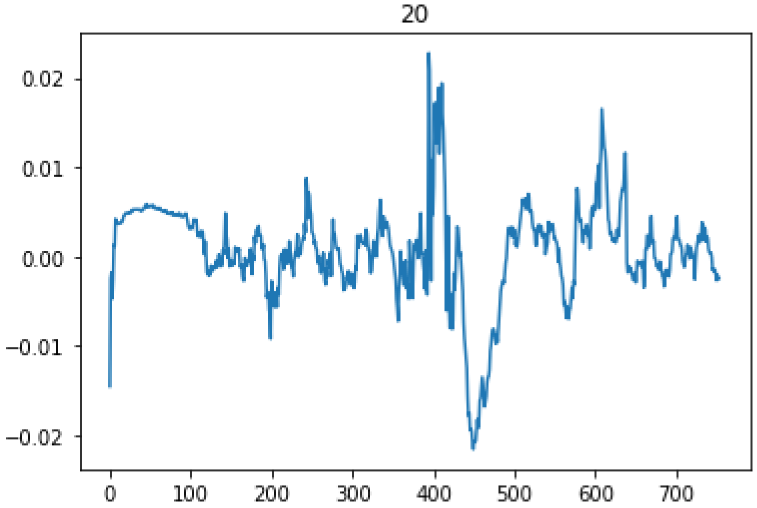
In [11]:

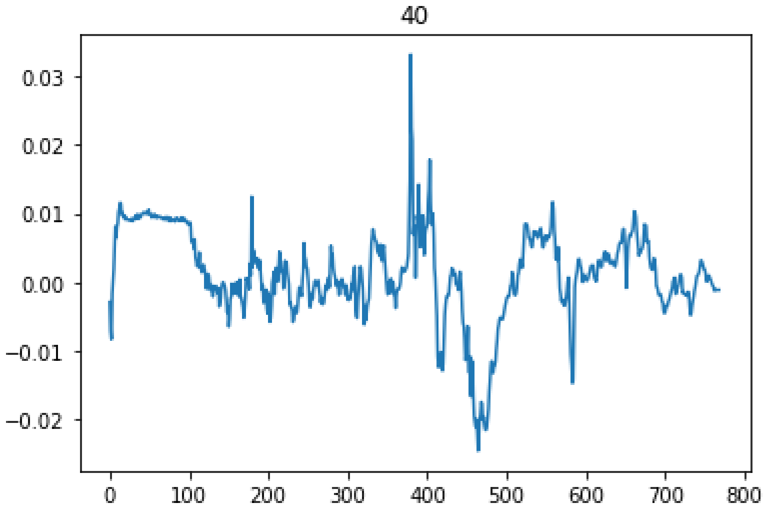
```
legend = ['clear', '80', '60', '20', '40', '100']
df_avg_dict = {}
for weather in range(6):
    df_avg = df_array[weather][0]['steer']
    for i in range(1, len(df_array[weather])):
        df_avg += df_array[weather][i]['steer']
    df_avg = df_avg.interpolate().dropna()/100
    df_avg_dict.update({legend[weather]: df_avg.copy()})
    df_avg.plot(y="Age")
    plt.title(legend[weather])
    plt.show()
```

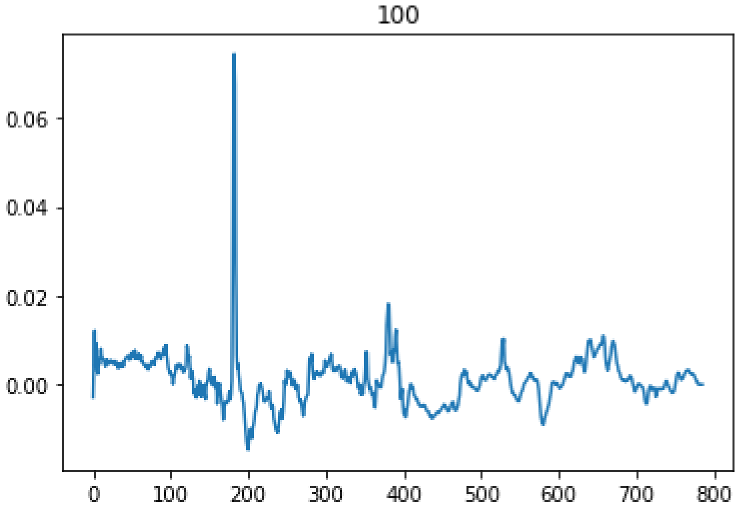



80
0.025
0.020
0.015
0.010
0.005
0.000
−0.005
−0.010
−0.015
0
100
200
300
400
500
600
700
800


60
0.03
0.02
0.01
0.00
−0.01
−0.02
−0.03
0
100
200
300
400
500
600
700


20
0.02
0.01
0.00
−0.01
−0.02
0
100
200
300
400
500
600
700





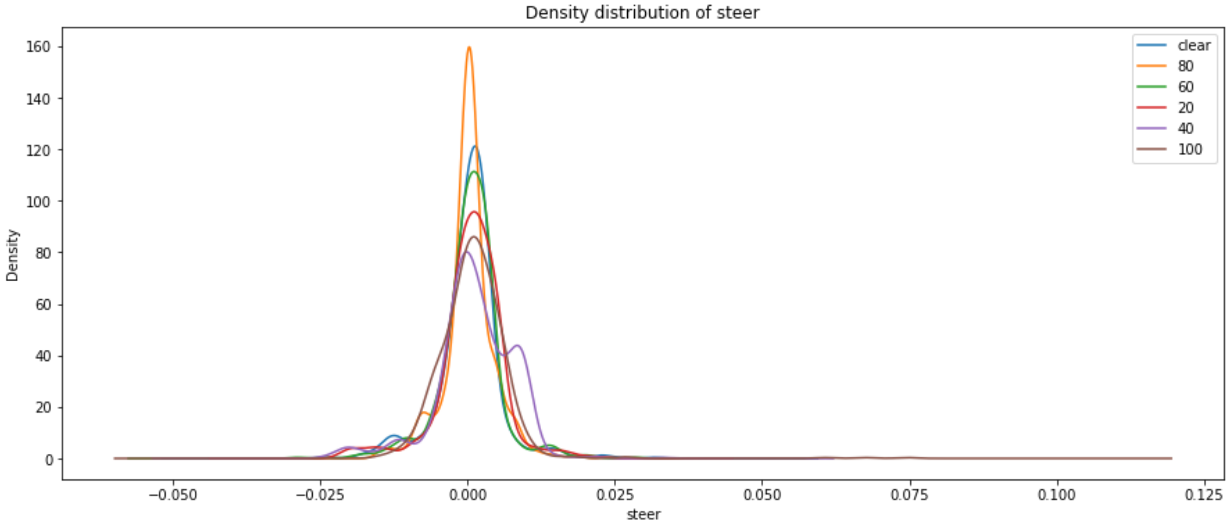
In [12]:

```
legend = ['clear', '80', '60', '20', '40', '100']
# for i in legend:
#     print(df_avg_dict[i])
```

In [13]:

```
fig = plt.figure(figsize=(15, 6))

for i in legend:
    df_avg_dict[i].plot.density()
plt.xlabel("steer")
plt.title('Density distribution of steer')
plt.legend(legend)
plt.show()
```



Dynamic time warping (DTW): measuring similarity between two temporal sequences

```
In [14]: result = []
         for i in legend:
             distance = dtw.distance(df_avg_dict['clear'], df_avg_dict[i])
             result.append((distance, i))
```

```
In [15]: result.sort(key=lambda y: y[0])
         print(result)
```

```
[(0.0, 'clear'), (0.02779703458004904, '60'), (0.04738881121655588, '20'), (0.0582760
8941554895, '80'), (0.0770693574504234, '40'), (0.09782514915937235, '100')]
```