In [ ]:

```python

```

executed in 11.6s, finished 16:21:01 2020-10-07

In [1]:

```python
from GlucoCheck.glucoCheck import glucoCheckOps
import pandas as pd
import random
import numpy as np
from tqdm.auto import tqdm

from scipy import stats

import random
import re
from dateutil.parser import parse

import warnings
warnings.filterwarnings('ignore')

import os

```

executed in 12.8s, finished 12:22:58 2020-10-28

```
Using TensorFlow backend.
```

In [2]:

```python
def createGap(df,start,end):
    """
    Creating a Gap
    input:
        start: seed
        end: seed + gap
    output:
        df: dataframe with index => DisplayTime value => GlucoseValue
    """

    #df = readData()
    l = len(df.index)
    if end>l:
        end = l

    for i in range(start,end):
        df['GlucoseValue'][i]=float("NaN")

    return df
```

executed in 43ms, finished 12:22:58 2020-10-28

In [3]:

```
#Extract Data
data = pd.read_csv("~/Desktop/NCSA_genomics/Python - notebooks/Data/O
data = data[data['subjectId']=='OD552']
data = data.reset_index(drop=True)
data
```

executed in 524ms, finished 12:22:58 2020-10-28

Out[3]:

	subjectId	Display Time	GlucoseValue
0	OD552	4/16/25 11:17	95
1	OD552	4/16/25 11:22	86
2	OD552	4/16/25 11:27	81
3	OD552	4/16/25 11:32	81
4	OD552	4/16/25 11:37	82
...	...	...	...
11439	OD552	6/7/25 16:49	238
11440	OD552	6/7/25 16:54	233
11441	OD552	6/7/25 16:59	229
11442	OD552	6/7/25 17:04	224
11443	OD552	6/7/25 17:09	215

11444 rows × 3 columns

In [ ]:

```

```

executed in 50ms, finished 15:27:50 2020-10-16

In [4]:

```
#1 week after: 1890,1974,2003,2196,2378,2581,2751,3190,3223,3301

seed_points = [1890,1974,2003,2196,2378,2581,2751,3190,3223,3301]

#
```

executed in 8ms, finished 12:22:58 2020-10-28

In [5]:

```
# obj = glucoCheckOps()
```

executed in 5ms, finished 12:22:58 2020-10-28

In [ ]:

```

```

executed in 5ms, finished 00:11:30 2020-10-23

In [6]:

```

#for gap size 50
ioa_gap50 = list()


for seed in tqdm(seed_points):
    start = seed
    end = seed+49

    dataWithMissing = data.copy()
    dataWithMissing = createGap(dataWithMissing,start,end)

    dataBeforeGap = dataWithMissing[:seed]

    obj = glucoCheckOps()
#     obj.train(dataBeforeGap);
    imputed_data = obj.impute(dataWithMissing)

    ioa = obj.index_agreement(np.asarray(imputed_data['GlucoseValue']

    del obj

    ioa_gap50.append(ioa)

ioa_gap50
```

executed in 6m 37s, finished 12:29:35 2020-10-28

```
100%                                    10/10 [07:47<00:00, 46.72s/it]
```

```
Gap < 50; We use LSTM imputations
Training Model...


Model trained successfully!
Gap < 50; We use LSTM imputations
Training Model...


Model trained successfully!
Gap < 50; We use LSTM imputations
Training Model...


Model trained successfully!
Gap < 50; We use LSTM imputations
Training Model...


Model trained successfully!
Gap < 50; We use LSTM imputations
Training Model...


Model trained successfully!
```

```
Gap < 50; We use LSTM imputations
Training Model...


Model trained successfully!
Gap < 50; We use LSTM imputations
Training Model...


Model trained successfully!
Gap < 50; We use LSTM imputations
Training Model...


Model trained successfully!
Gap < 50; We use LSTM imputations
Training Model...


Model trained successfully!
Gap < 50; We use LSTM imputations
Training Model...


Model trained successfully!
```

```
Out[6]: [0.501322180512248,
         0.3514284527648076,
         0.9404086824960632,
         0.21513311245878852,
         0.00010339886072030513,
         0.7750068577387832,
         0.1466092249533928,
         0.46736588002738455,
         0.2876924169801187,
         0.2393198658634842]
```

In [7]:

```
#for gap size 30
ioa_gap30 = list()

for seed in tqdm(seed_points):
    start = seed
    end = start+29

    dataWithMissing = data.copy()
    dataWithMissing = createGap(dataWithMissing,start,end)

    dataBeforeGap = dataWithMissing[:seed]

    obj = glucoCheckOps()
#     obj.train(dataBeforeGap);
    imputed_data = obj.impute(dataWithMissing)

    ioa = obj.index_agreement(np.asarray(imputed_data['GlucoseValue'][

    del obj

    ioa_gap30.append(ioa)

ioa_gap30

```

executed in 6m 40s, finished 12:36:14 2020-10-28

```
100%                                    10/10 [06:39<00:00, 39.97s/it]
```

```
Gap < 50; We use LSTM imputations
Training Model...


Model trained successfully!
Gap < 50; We use LSTM imputations
Training Model...


Model trained successfully!
Gap < 50; We use LSTM imputations
Training Model...


Model trained successfully!
Gap < 50; We use LSTM imputations
Training Model...


Model trained successfully!
Gap < 50; We use LSTM imputations
Training Model...


Model trained successfully!
Gap < 50; We use LSTM imputations
```

```
Training Model...


Model trained successfully!
Gap < 50; We use LSTM imputations
Training Model...


Model trained successfully!
Gap < 50; We use LSTM imputations
Training Model...


Model trained successfully!
Gap < 50; We use LSTM imputations
Training Model...


Model trained successfully!
Gap < 50; We use LSTM imputations
Training Model...


Model trained successfully!
```

```
Out[7]: [0.309784549478317 24,
```

```
Out[7]: [0.30978454947831724,
         0.4792044789412776,
         0.3379008664834321,
         0.4189059469155322,
         0.17637675343657444,
         0.6279498956260033,
         0.19239509020081258,
         0.4177682176504518,
         0.310971683767826,
         0.7526675667436087]
```

In [8]:

```
#for gap size 12
ioa_gap15 = list()


for seed in tqdm(seed_points):
    start = seed
    end = start+12

    dataWithMissing = data.copy()
    dataWithMissing = createGap(dataWithMissing,start,end)

    dataBeforeGap = dataWithMissing[:seed]

    obj = glucoCheckOps()
#     obj.train(dataBeforeGap);
    imputed_data = obj.impute(dataWithMissing)

    ioa = obj.index_agreement(np.asarray(imputed_data['GlucoseValue']

    del obj

    ioa_gap15.append(ioa)

ioa_gap15
```

executed in 38.7s, finished 12:36:53 2020-10-28

```
100%                                    10/10 [00:39<00:00, 3.97s/it]
```

```
Gap < 15; We use the spline imputations
Gap < 15; We use the spline imputations
Gap < 15; We use the spline imputations
Gap < 15; We use the spline imputations
Gap < 15; We use the spline imputations
Gap < 15; We use the spline imputations
Gap < 15; We use the spline imputations
Gap < 15; We use the spline imputations
Gap < 15; We use the spline imputations
Gap < 15; We use the spline imputations
```

Out[8]:

```
[0.8225174137987098,
 0.9374015908773021,
 0.8307417323953903,
 0.6371247396644975,
 0.7692234986326327,
 0.5505256745687919,
 0.9082767150763599,
 0.7008917378638158,
 0.9836676941882905,
 0.3388264061663081]
```

In [9]:

```
#for gap size 100
ioa_gap100 = list()

for seed in tqdm(seed_points):
    start = seed
    end = seed+99

    dataWithMissing = data.copy()
    dataWithMissing = createGap(dataWithMissing,start,end)

    dataBeforeGap = dataWithMissing[:seed]

    obj = glucoCheckOps()
#     obj.train(dataBeforeGap);
    imputed_data = obj.impute(dataWithMissing)

    if isinstance(imputed_data, pd.DataFrame):
        ioa = obj.index_agreement(np.asarray(imputed_data['GlucoseValue'][s
        ioa_gap100.append(ioa)
    else:
        ioa_gap100.append(0)

    del obj



ioa_gap100
```

executed in 360ms, finished 12:36:54 2020-10-28

```
100%                                    10/10 [00:00<00:00, 29.85it/s]

We cannot impute this data
We cannot impute this data
We cannot impute this data
We cannot impute this data
We cannot impute this data
We cannot impute this data
We cannot impute this data
We cannot impute this data
We cannot impute this data
We cannot impute this data
```

Out[9]:

```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

In [10]:

```
_points):



 data.copy()
 createGap(dataWithMissing,start,end)

ataWithMissing[:seed]

ps()
eforeGap);
jimpute(dataWithMissing)

greement(np.asarray(imputed_data['GlucoseValue'][start:end-1].tolist()),np.a



oa)
```

executed in 289ms, finished 12:36:54 2020-10-28

```
100%                                    10/10 [00:00<00:00, 27.73it/s]

Gap < 5; We use the linear imputations
Gap < 5; We use the linear imputations
Gap < 5; We use the linear imputations
Gap < 5; We use the linear imputations
Gap < 5; We use the linear imputations
Gap < 5; We use the linear imputations
Gap < 5; We use the linear imputations
Gap < 5; We use the linear imputations
Gap < 5; We use the linear imputations
Gap < 5; We use the linear imputations
```

Out[10]:

```
[0.0,
 0.8695652173913043,
 0.1950263612186861,
 0.6756226527904932,
 0.9830310122878874,
 0.3587518771900524,
 0.698414270329153,
 0.883601806295073,
 0.8503937007874014,
 0.47058823529411054]
```

In [ ]:

In [11]:

```
IOA = pd.DataFrame({'Seeds':seed_points, 'Gap:5':ioa_gap5, 'Gap:12':i
IOA
```

executed in 29ms, finished 12:36:54 2020-10-28

Out[11]:

	Seeds	Gap:5	Gap:12	Gap:30	Gap:50	Gap:100
0	1890	0.000000	0.822517	0.309785	0.501322	0
1	1974	0.869565	0.937402	0.479204	0.351428	0
2	2003	0.195026	0.830742	0.337901	0.940409	0
3	2196	0.675623	0.637125	0.418906	0.215133	0
4	2378	0.983031	0.769223	0.176377	0.000103	0
5	2581	0.358752	0.550526	0.627950	0.775007	0
6	2751	0.698414	0.908277	0.192395	0.146609	0
7	3190	0.883602	0.700892	0.417768	0.467366	0
8	3223	0.850394	0.983668	0.310972	0.287692	0
9	3301	0.470588	0.338826	0.752668	0.239320	0

In [ ]:

In [12]:

```
IOA.to_csv("~/Desktop/1week.csv")
```

executed in 16ms, finished 12:36:54 2020-10-28

In [ ]:

In [ ]:

In [13]:

```
import matplotlib.pyplot as plt
import matplotlib.ticker as ticker
import seaborn as sns
```

executed in 5ms, finished 12:36:54 2020-10-28

In [14]:

```
# IOA
```

executed in 7ms, finished 12:36:54 2020-10-28

In [15]:

```
gaps = [5,15,30,50,100]
ioa = []
ioa.append(IOA['Gap:5'].mean())
ioa.append(IOA['Gap:12'].mean())
ioa.append(IOA['Gap:30'].mean())
ioa.append(IOA['Gap:50'].mean())
ioa.append(IOA['Gap:100'].mean())
ioa
```

executed in 15ms, finished 12:36:54 2020-10-28

Out[15]:

```
[0.5984995133584162,
 0.7479197203232097,
 0.4023925049243836,
 0.39243900726557907,
 0.0]
```

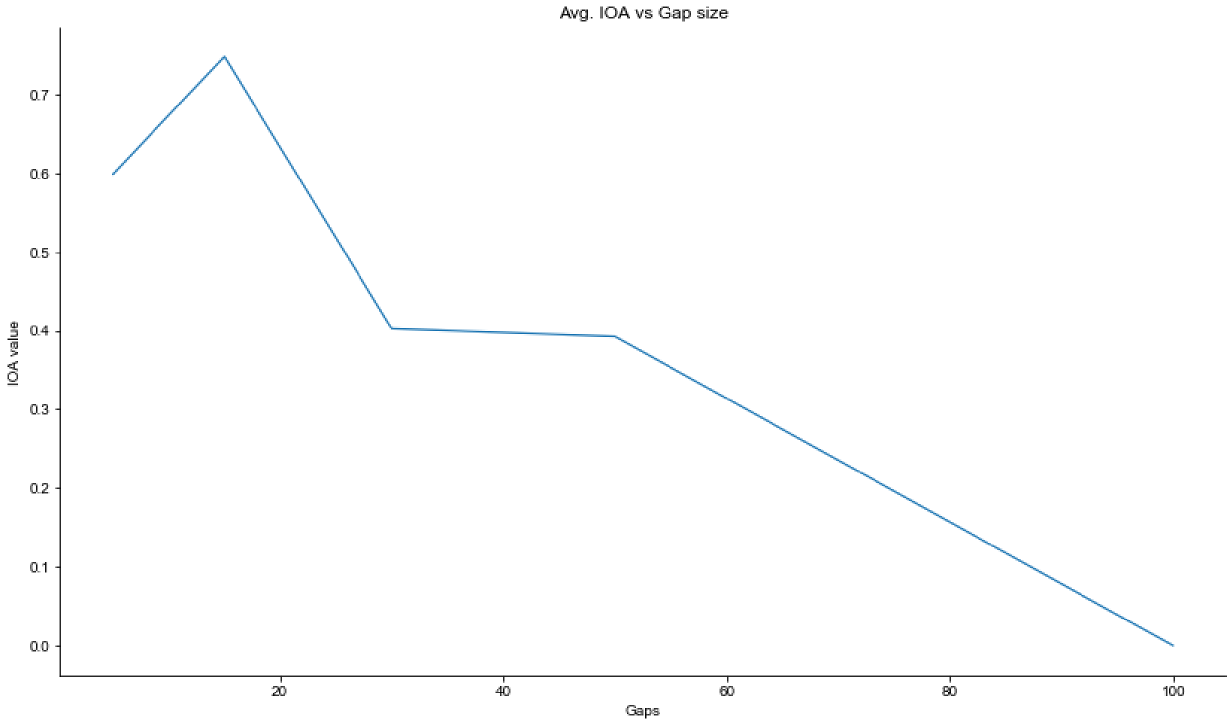
In [16]:

```
plt.figure(figsize=(14,8))
plt.title("Avg. IOA vs Gap size")
sns.set(style="white")
fig = sns.lineplot(x = gaps, y = ioa, palette="tab10", linewidth=1.25
sns.despine()

fig.set_xlabel('Gaps')
fig.set_ylabel('IOA value')
```

executed in 504ms, finished 12:36:54 2020-10-28

Out[16]:

```
Text(0, 0.5, 'IOA value')
```



In [ ]:

```

```

In [ ]:

```

```

executed in 8ms, finished 14:05:45 2020-10-14

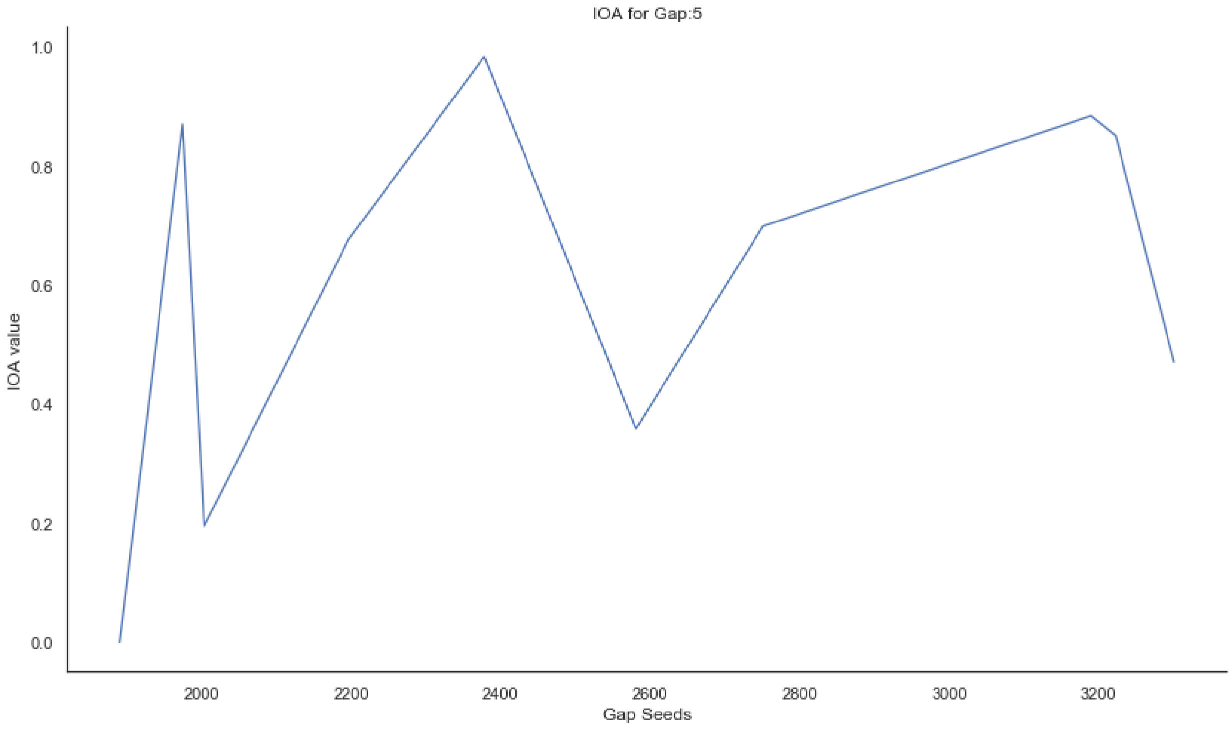
In [17]:

```

plt.figure(figsize=(14,8))
plt.title("IOA for Gap:5")
sns.set(style="white")
fig = sns.lineplot(x = seed_points, y = IOA['Gap:5'], data = IOA, pal
sns.despine()

fig.set_xlabel('Gap Seeds')
fig.set_ylabel('IOA value')
```

executed in 658ms, finished 12:36:55 2020-10-28

Out[17]:

```
Text(0, 0.5, 'IOA value')
```



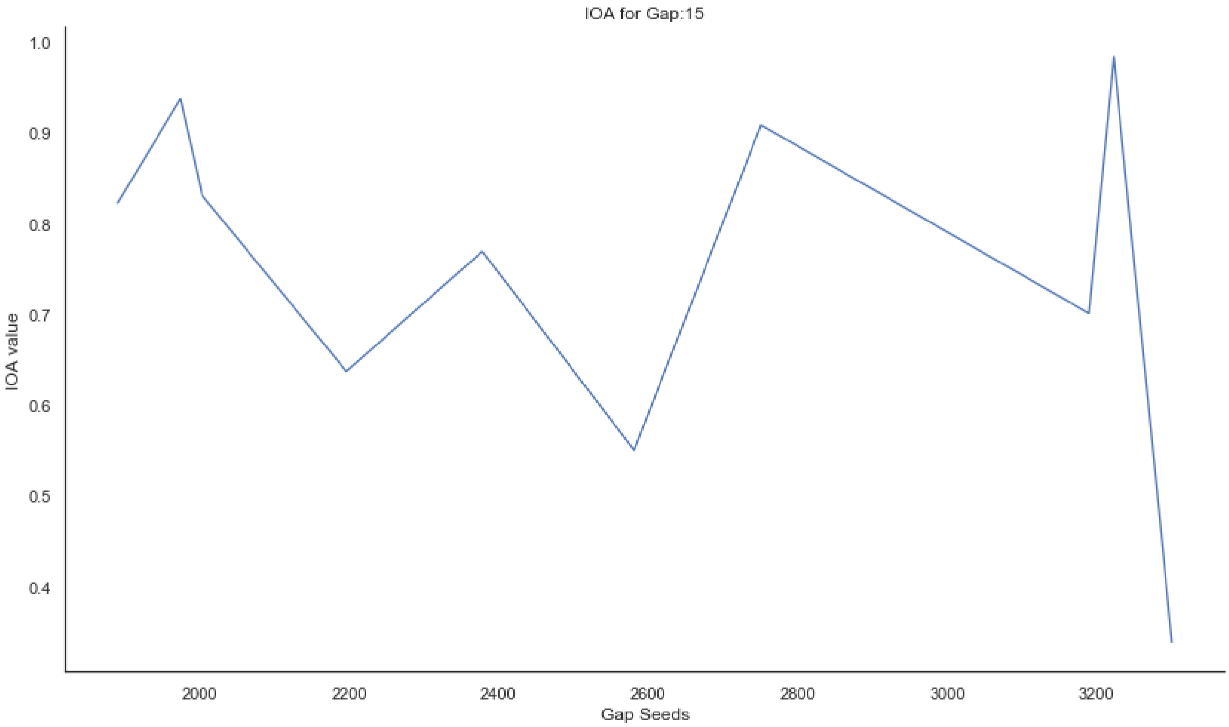
In [18]:

```python

plt.figure(figsize=(14,8))
plt.title("IOA for Gap:15")
sns.set(style="white")
fig = sns.lineplot(x = seed_points, y = IOA['Gap:12'], data = IOA, pa
sns.despine()

fig.set_xlabel('Gap Seeds')
fig.set_ylabel('IOA value')
```

executed in 604ms, finished 12:36:56 2020-10-28

Out[18]:

```
Text(0, 0.5, 'IOA value')
```



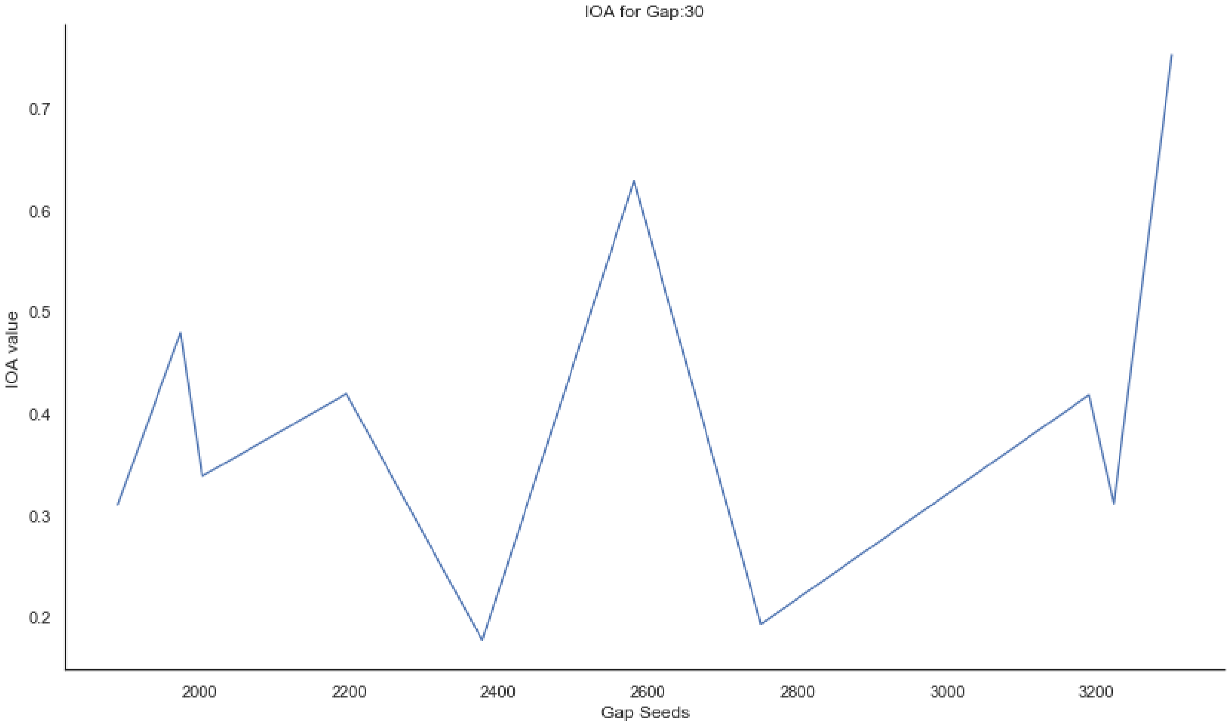
In [19]:

```

plt.figure(figsize=(14,8))
plt.title("IOA for Gap:30")
sns.set(style="white")
fig = sns.lineplot(x = seed_points, y = IOA['Gap:30'], data = IOA, pa
sns.despine()

fig.set_xlabel('Gap Seeds')
fig.set_ylabel('IOA value')
```

executed in 548ms, finished 12:36:56 2020-10-28

Out[19]:

```
Text(0, 0.5, 'IOA value')
```



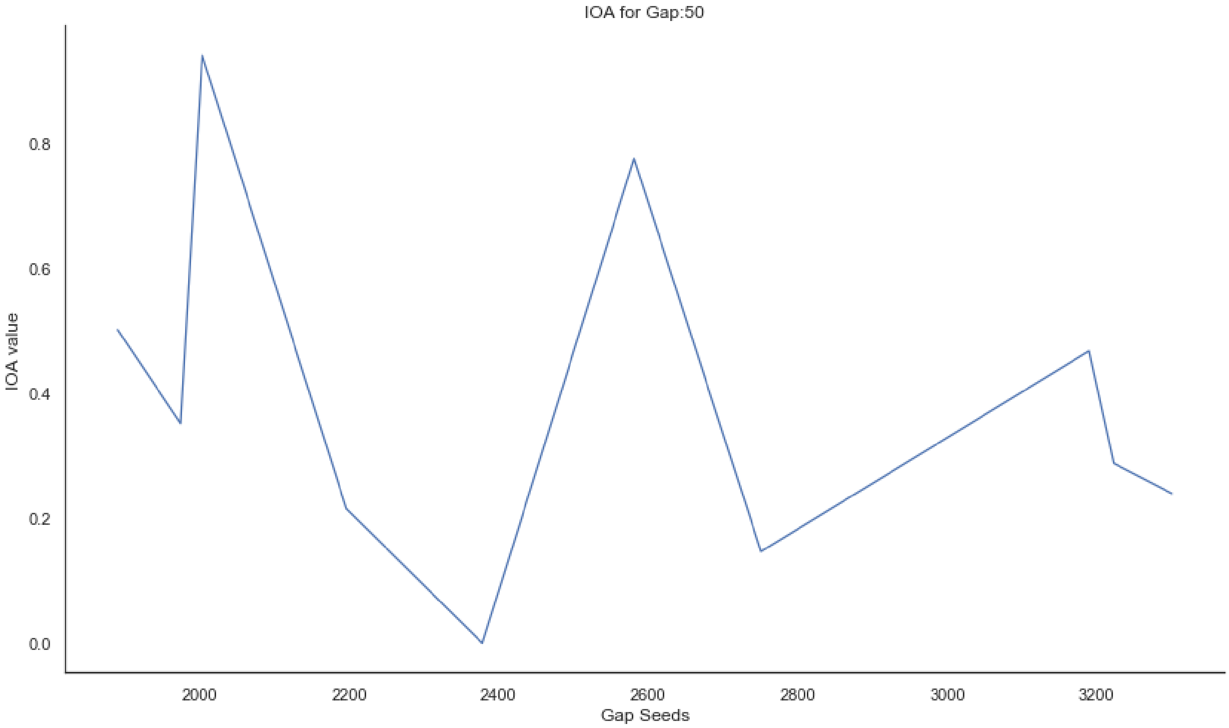
In [20]:

```

plt.figure(figsize=(14,8))
plt.title("IOA for Gap:50")
sns.set(style="white")
fig = sns.lineplot(x = seed_points, y = IOA['Gap:50'], data = IOA, pa
sns.despine()

fig.set_xlabel('Gap Seeds')
fig.set_ylabel('IOA value')
```

executed in 549ms, finished 12:36:57 2020-10-28

Out[20]:

```
Text(0, 0.5, 'IOA value')
```



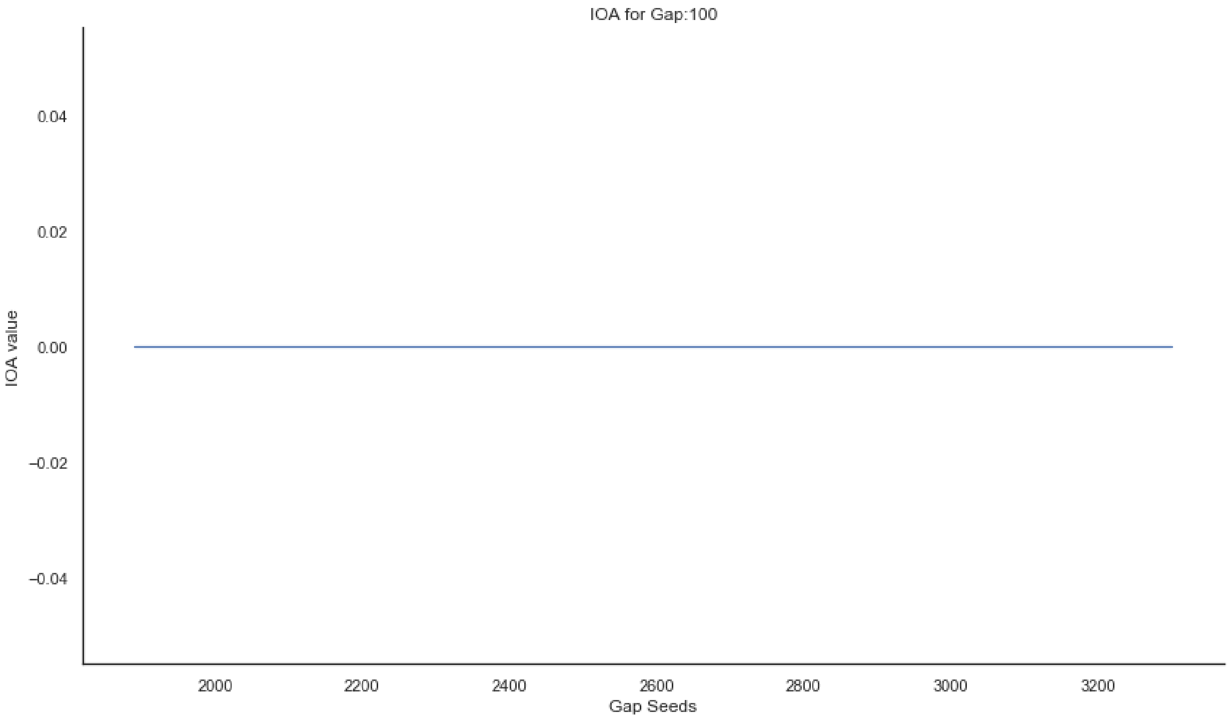
In [21]:

```

plt.figure(figsize=(14,8))
plt.title("IOA for Gap:100")
sns.set(style="white")
fig = sns.lineplot(x = seed_points, y = IOA['Gap:100'], data = IOA, p
sns.despine()
fig.set_xlabel('Gap Seeds')
fig.set_ylabel('IOA value')
```

executed in 601ms, finished 12:36:57 2020-10-28

Out[21]:

```
Text(0, 0.5, 'IOA value')
```



In [ ]:

```

```

In [ ]:

```

```

In [22]:

```
# MAD = pd.DataFrame({'Gap:5':mad_gap5, 'Gap:15':mad_gap15, 'Gap:30':m
# MAD
```

executed in 7ms, finished 12:36:57 2020-10-28

In [23]:

```
 # FB = pd.DataFrame({'Gap:5':fb_gap5, 'Gap:15':fb_gap15, 'Gap:30':fb_
 # FB
```

executed in 7ms, finished 12:36:57 2020-10-28

In [24]:

```
 # RMSE = pd.DataFrame({'Gap:5':rmse_gap5, 'Gap:15':rmse_gap15, 'Gap:3
 # RMSE
```

executed in 8ms, finished 12:36:57 2020-10-28

In [25]:

```
 # MAPE = pd.DataFrame({'Gap:5':mape_gap5, 'Gap:15':mape_gap15, 'Gap:3
 # MAPE
```

executed in 55ms, finished 12:36:58 2020-10-28

In [26]:

```
 # IOA.to_csv("~/Desktop/NCSA_genomics/Python - notebooks/GlucoCheck/M
 # FB.to_csv("~/Desktop/NCSA_genomics/Python - notebooks/GlucoCheck/Me
 # RMSE.to_csv("~/Desktop/NCSA_genomics/Python - notebooks/GlucoCheck/
 # MAPE.to_csv("~/Desktop/NCSA_genomics/Python - notebooks/GlucoCheck/
 # MAD.to_csv("~/Desktop/NCSA_genomics/Python - notebooks/GlucoCheck/M
```

executed in 11ms, finished 12:36:58 2020-10-28

In [ ]:

```

```