In [ ]:
```
1
```
executed in 11.6s, finished 16:21:01 2020-10-07

In [1]:
```
1  from GlucoCheck.glucoCheck import glucoCheckOps
2  import pandas as pd
3  import random
4  import numpy as np
5  from tqdm.auto import tqdm
6
7  from scipy import stats
8
9  import random
10 import re
11 from dateutil.parser import parse
12
13 import warnings
14 warnings.filterwarnings('ignore')
15
16 import os
17
```
executed in 11.6s, finished 12:17:34 2020-10-27

```
Using TensorFlow backend.
```

In [13]:
```
1  def createGap(df,start,end):
2      """
3      Creating a Gap
4      input:
5          start: seed
6          end: seed + gap
7      output:
8          df: dataframe with index => DisplayTime value => GlucoseValue
9      """
10
11     #df = readData()
12     l = len(df.index)
13     if end>l:
14         end = l
15
16     for i in range(start,end):
17         df['GlucoseValue'][i]=float("NaN")
18
19     return df
```
executed in 40ms, finished 12:49:20 2020-10-27

In [9]:
```
#Extract Data
data = pd.read_csv("~/Desktop/NCSA_genomics/Python - notebooks/GlucoC
data = data[data['subjectId']=='OD552']
data = data.reset_index(drop=True)
data
```

executed in 194ms, finished 12:49:08 2020-10-27

Out[9]:

	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 [ ]:
```
1
```

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

In [10]:
```
1
2
3  # good -> 9500
4  # bad -> 9000,10000,10500
5
6  seed_points = [9250,9320,9500,9750,9890,10100,10600,10890,11200,11220
7
8  #
```

executed in 6ms, finished 12:49:09 2020-10-27

In [11]:
```
1 # obj = glucoCheckOps()
```

executed in 14ms, finished 12:49:09 2020-10-27

In [ ]:
```
1
```

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

In [14]:

```
#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 47m 32s, finished 13:36:55 2020-10-27

```
Training Model...


Model trained successfully!
Object Created!
Training Model...


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


Model trained successfully!
Object Created!
Training Model...


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

In [15]:

```
#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 49m 51s, finished 14:26:46 2020-10-27

100% 10/10 [51:41<00:00, 310.13s/it]

In [17]:
```
#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 46.3s, finished 14:28:33 2020-10-27

```
100%                                              10/10 [00:46<00:00, 4.62s/it]

Object Created!
Gap < 15; We use the spline imputations
Object Created!
Gap < 15; We use the spline imputations
Object Created!
Gap < 15; We use the spline imputations
Object Created!
Gap < 15; We use the spline imputations
Object Created!
Gap < 15; We use the spline imputations
Object Created!
Gap < 15; We use the spline imputations
Object Created!
Gap < 15; We use the spline imputations
Object Created!
Gap < 15; We use the spline imputations
Object Created!
Gap < 15; We use the spline imputations
Object Created!
Gap < 15; We use the spline imputations
```

Out[17]:
```
[0.901047422587044,
 0.8355096781920831,
 0.9084209045348344,
 0.3998047906322084,
 0.42304458677753587,
```

```
0.9668900387262722,
0.8844472656272231,
0.4824631657548668,
0.4403790006745705,
0.9727535389219776]
```

In [22]:
```
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'][sta
        ioa_gap100.append(ioa)
    else:
        ioa_gap100.append(0)

    del obj


ioa_gap100
```

executed in 189ms, finished 14:33:51 2020-10-27

```
100%                                              10/10 [00:43<00:00, 4.32s/it]

Object Created!
We cannot impute this data
Object Created!
We cannot impute this data
Object Created!
We cannot impute this data
Object Created!
We cannot impute this data
Object Created!
We cannot impute this data
Object Created!
We cannot impute this data
Object Created!
We cannot impute this data
Object Created!
We cannot impute this data
Object Created!
We cannot impute this data
Object Created!
We cannot impute this data
```

Out[22]:
```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

In [23]:

```
#for gap size 5
ioa_gap5 = list()
# fb_gap5 = list()
# mad_gap5 = list()
# rmse_gap5 = list()
# mape_gap5 = list()

for seed in tqdm(seed_points):
    start = seed
    end = start+4

    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_gap5.append(ioa)



ioa_gap5
```

executed in 247ms, finished 14:33:52 2020-10-27

100% 10/10 [00:00<00:00, 47.84it/s]

```
Object Created!
Gap < 5; We use the linear imputations
Object Created!
Gap < 5; We use the linear imputations
Object Created!
Gap < 5; We use the linear imputations
Object Created!
Gap < 5; We use the linear imputations
Object Created!
Gap < 5; We use the linear imputations
Object Created!
Gap < 5; We use the linear imputations
Object Created!
Gap < 5; We use the linear imputations
Object Created!
Gap < 5; We use the linear imputations
Object Created!
```

```
Gap < 5; We use the linear imputations
Object Created!
Gap < 5; We use the linear imputations
```

Out[23]:
```
[0.9954158480681075,
 0.5290697674418621,
 0.5048675571654959,
 0.9333333333333335,
 0.31883609348398023,
 0.5696846388606309,
 0.4073898626243483,
 0.5106697708037422,
 0.39981224951558103,
 0.8741035290776108]
```

In [ ]:
```
1
```

In [25]:
```
1 IOA = pd.DataFrame({'Seeds':seed_points, 'Gap:5':ioa_gap5, 'Gap:12':i
2 IOA
3
```

executed in 21ms, finished 14:34:20 2020-10-27

Out[25]:

	Seeds	Gap:5	Gap:12	Gap:30	Gap:50	Gap:100
0	9250	0.995416	0.901047	0.433157	0.453610	0
1	9320	0.529070	0.835510	0.229344	0.891468	0
2	9500	0.504868	0.908421	0.772168	0.520654	0
3	9750	0.933333	0.399805	0.966751	0.785829	0
4	9890	0.318836	0.423045	0.597910	0.468042	0
5	10100	0.569685	0.966890	0.805871	0.427142	0
6	10600	0.407390	0.884447	0.486732	0.461335	0
7	10890	0.510670	0.482463	0.287861	0.393222	0
8	11200	0.399812	0.440379	0.785963	0.891709	0
9	11220	0.874104	0.972754	0.198728	0.598565	0

In [ ]:
```
1
```

In [ ]:
```
1 IOA.to_csv("~/Desktop/IOA.csv")
```

executed in 46ms, finished 19:07:38 2020-10-26

In [ ]:
```
1
```

In [ ]:
```
1
```

In [29]:
```
1  import matplotlib.pyplot as plt
2  import matplotlib.ticker as ticker
3  import seaborn as sns
```
executed in 8ms, finished 14:39:42 2020-10-27

In [30]:
```
1  # IOA
```
executed in 10ms, finished 14:39:42 2020-10-27

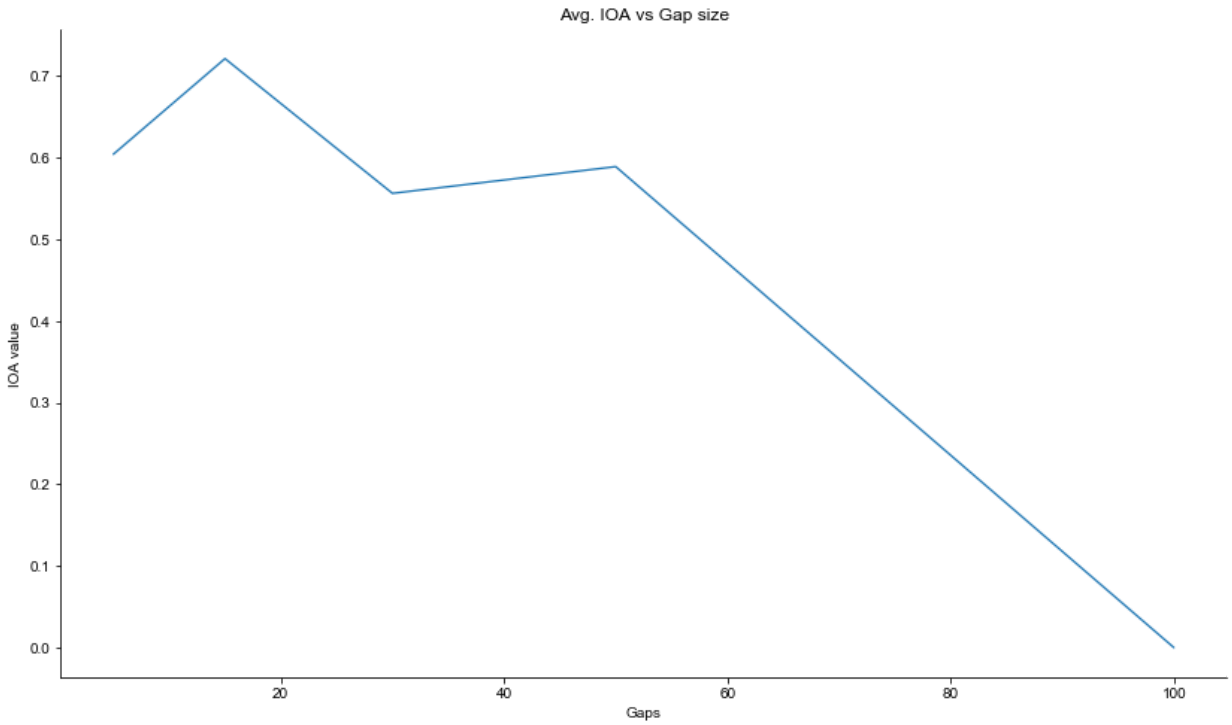
In [31]:
```
1  gaps = [5,15,30,50,100]
2  ioa = []
3  ioa.append(IOA['Gap:5'].mean())
4  ioa.append(IOA['Gap:12'].mean())
5  ioa.append(IOA['Gap:30'].mean())
6  ioa.append(IOA['Gap:50'].mean())
7  ioa.append(IOA['Gap:100'].mean())
8  ioa
```
executed in 18ms, finished 14:39:42 2020-10-27

Out[31]:
```
[0.6043182650374692,
 0.7214760392428616,
 0.5564486502658464,
 0.5891576338840601,
 0.0]
```

In [32]:
```
1  plt.figure(figsize=(14,8))
2  plt.title("Avg. IOA vs Gap size")
3  sns.set(style="white")
4  fig = sns.lineplot(x = gaps, y = ioa, palette="tab10", linewidth=1.25
5  sns.despine()
6
7  fig.set_xlabel('Gaps')
8  fig.set_ylabel('IOA value')
```

executed in 533ms, finished 14:39:43 2020-10-27

Out[32]: Text(0, 0.5, 'IOA value')



In [ ]:
```
1
```

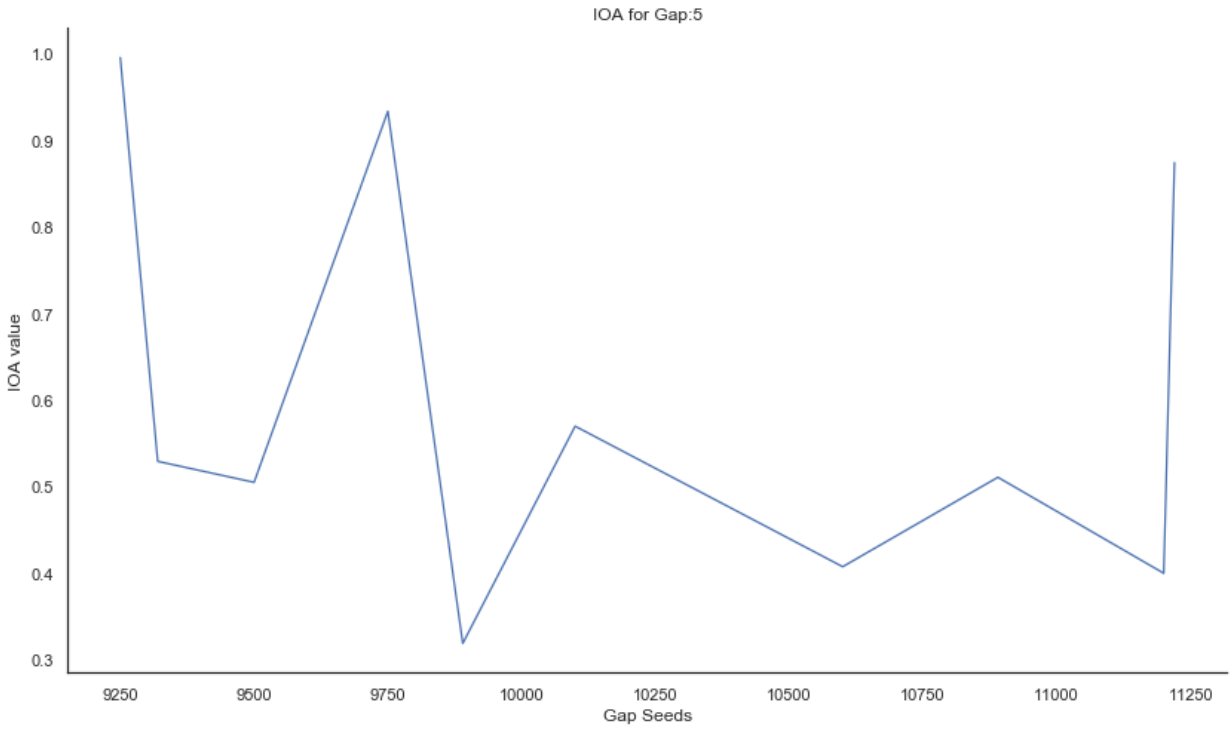
In [ ]:
```
1
```

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

In [33]:
```
1
2  plt.figure(figsize=(14,8))
3  plt.title("IOA for Gap:5")
4  sns.set(style="white")
5  fig = sns.lineplot(x = seed_points, y = IOA['Gap:5'], data = IOA, pal
6  sns.despine()
7
8  fig.set_xlabel('Gap Seeds')
9  fig.set_ylabel('IOA value')
```

executed in 525ms, finished 14:39:50 2020-10-27

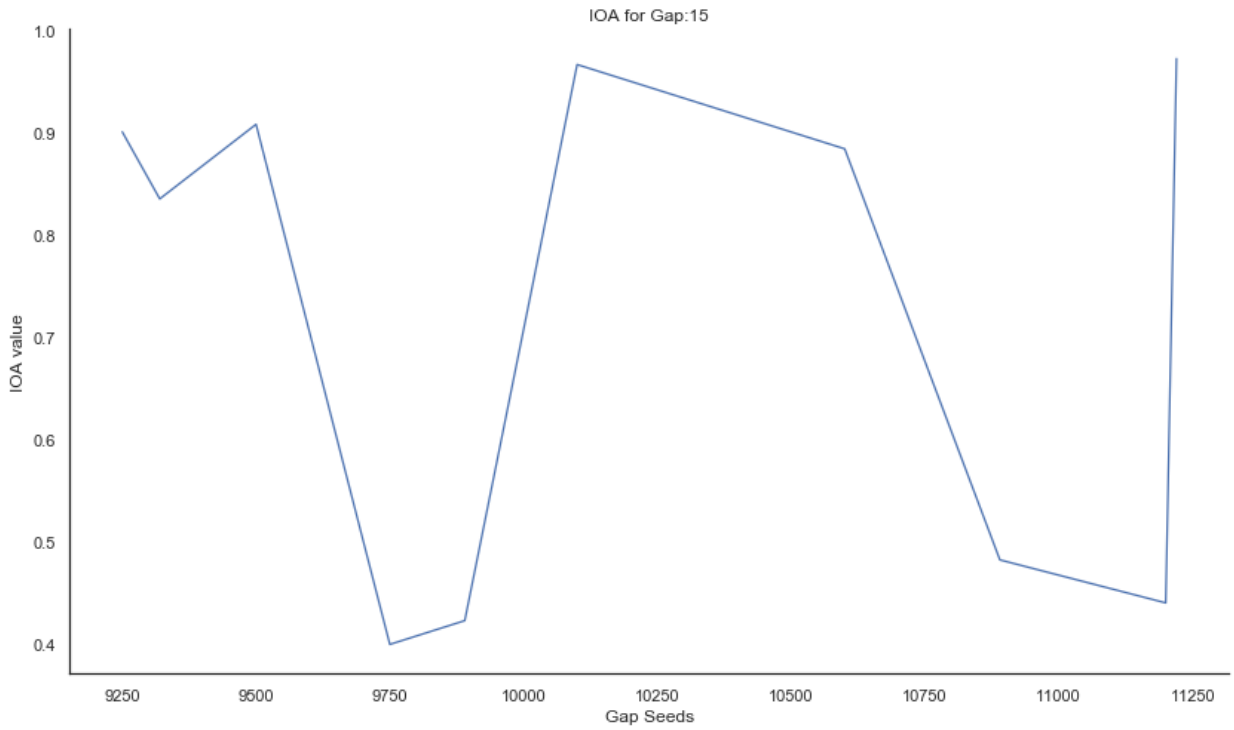
Out[33]: Text(0, 0.5, 'IOA value')



In [35]:
```
1
2  plt.figure(figsize=(14,8))
3  plt.title("IOA for Gap:15")
4  sns.set(style="white")
5  fig = sns.lineplot(x = seed_points, y = IOA['Gap:12'], data = IOA, pa
6  sns.despine()
7
8  fig.set_xlabel('Gap Seeds')
9  fig.set_ylabel('IOA value')
```

executed in 558ms, finished 14:40:01 2020-10-27

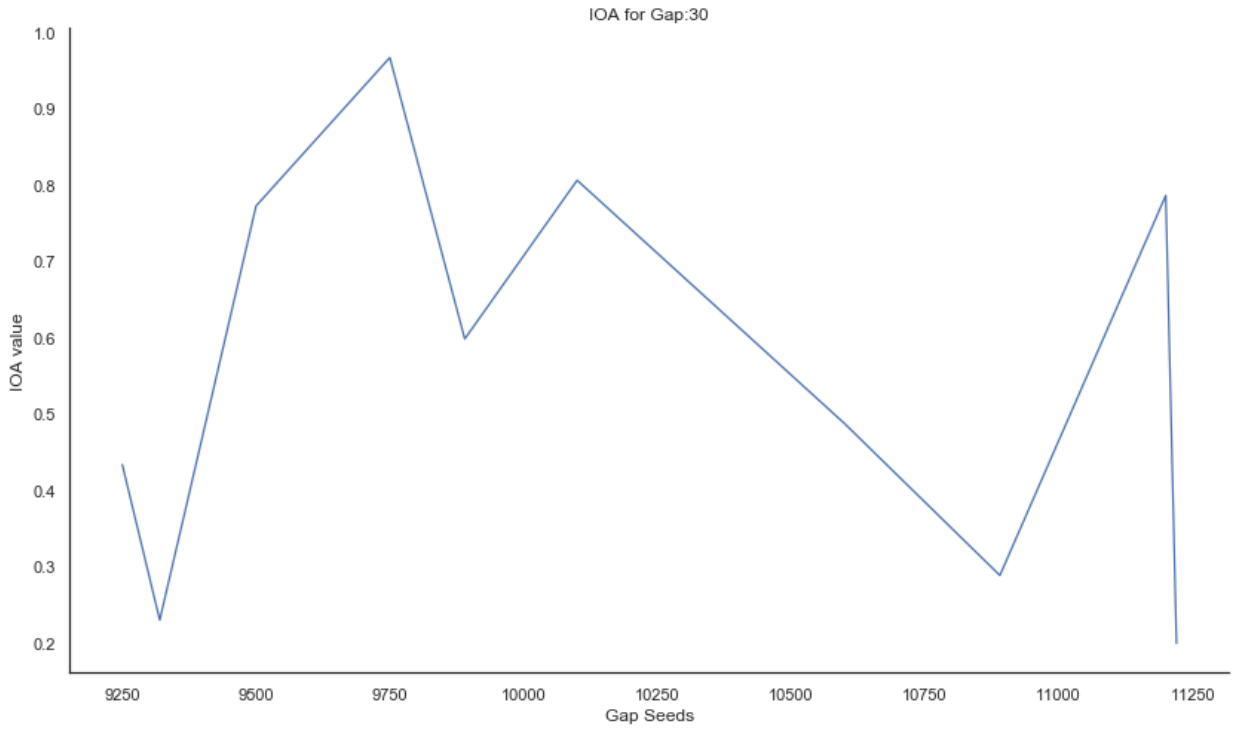
Out[35]: Text(0, 0.5, 'IOA value')



In [36]:
```
1
2  plt.figure(figsize=(14,8))
3  plt.title("IOA for Gap:30")
4  sns.set(style="white")
5  fig = sns.lineplot(x = seed_points, y = IOA['Gap:30'], data = IOA, pa
6  sns.despine()
7
8  fig.set_xlabel('Gap Seeds')
9  fig.set_ylabel('IOA value')
```

executed in 520ms, finished 14:40:07 2020-10-27

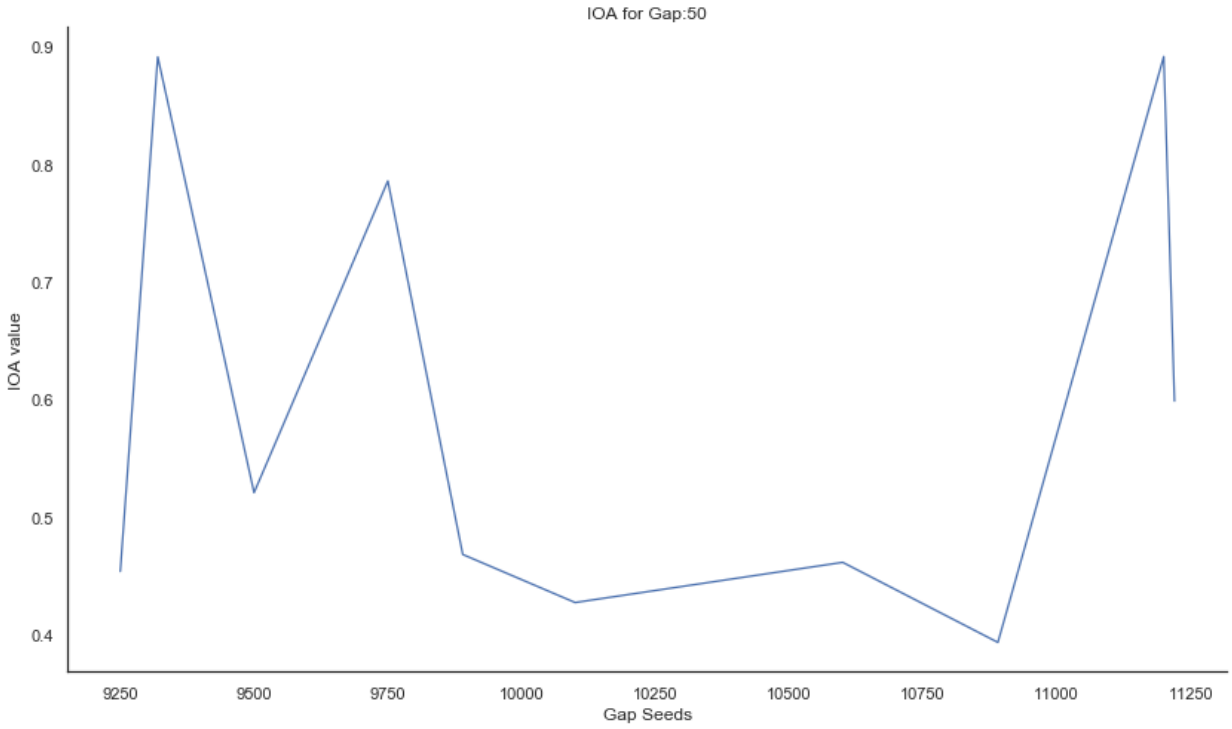
Out[36]: Text(0, 0.5, 'IOA value')



In [37]:
```
1
2  plt.figure(figsize=(14,8))
3  plt.title("IOA for Gap:50")
4  sns.set(style="white")
5  fig = sns.lineplot(x = seed_points, y = IOA['Gap:50'], data = IOA, pa
6  sns.despine()
7
8  fig.set_xlabel('Gap Seeds')
9  fig.set_ylabel('IOA value')
```

executed in 566ms, finished 14:40:10 2020-10-27

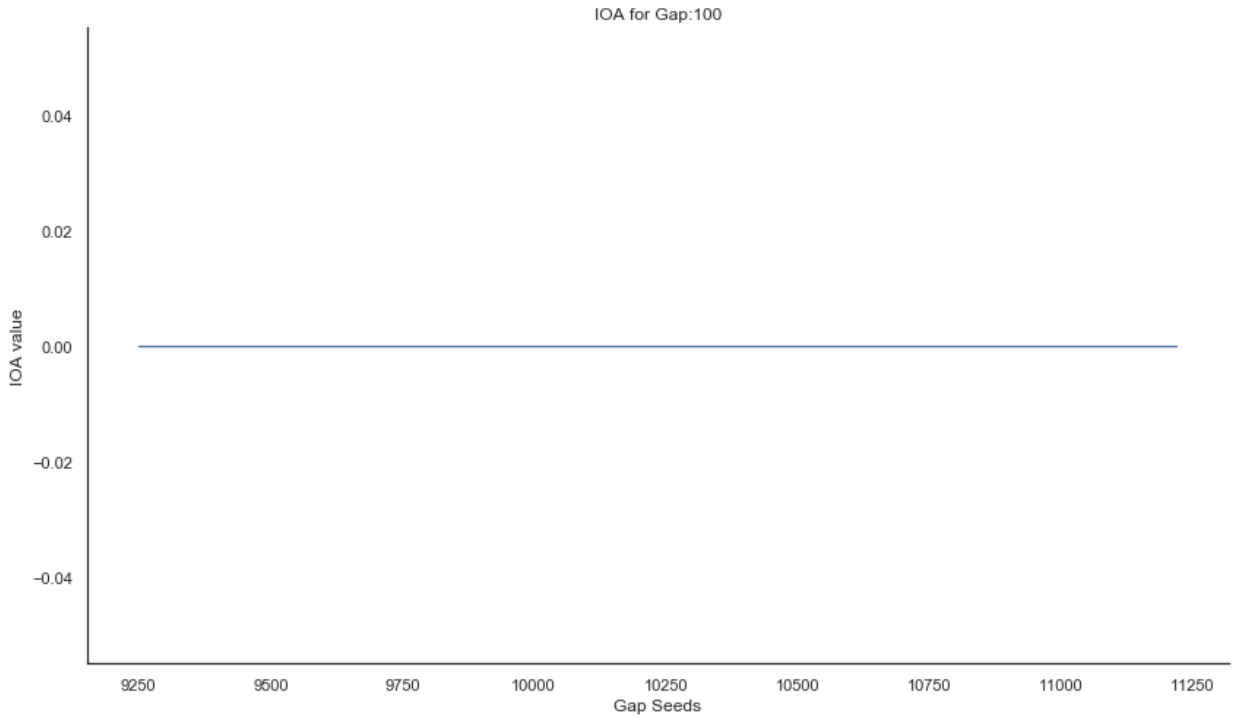
Out[37]: Text(0, 0.5, 'IOA value')



In [38]:
```
1
2  plt.figure(figsize=(14,8))
3  plt.title("IOA for Gap:100")
4  sns.set(style="white")
5  fig = sns.lineplot(x = seed_points, y = IOA['Gap:100'], data = IOA, p
6  sns.despine()
7  fig.set_xlabel('Gap Seeds')
8  fig.set_ylabel('IOA value')
```

executed in 472ms, finished 14:40:16 2020-10-27

Out[38]: Text(0, 0.5, 'IOA value')



In [ ]:
```
1
```

In [ ]:
```
1
```

In [ ]:
```
# MAD = pd.DataFrame({'Gap:5':mad_gap5, 'Gap:15':mad_gap15, 'Gap:30':m
# MAD
```
executed in 16ms, finished 16:48:50 2020-10-16

In [ ]:
```
# FB = pd.DataFrame({'Gap:5':fb_gap5, 'Gap:15':fb_gap15, 'Gap:30':fb_
# FB
```
executed in 23ms, finished 16:48:50 2020-10-16

In [ ]:
```
# RMSE = pd.DataFrame({'Gap:5':rmse_gap5, 'Gap:15':rmse_gap15, 'Gap:3
# RMSE
```
executed in 24ms, finished 16:48:51 2020-10-16

In [ ]:
```
# MAPE = pd.DataFrame({'Gap:5':mape_gap5, 'Gap:15':mape_gap15, 'Gap:3
# MAPE
```
executed in 20ms, finished 16:48:51 2020-10-16

In [ ]:
```
# 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 16:48:52 2020-10-16

In [ ]:
```

```