Exploring Roller Coasters: Unveiling Thrills and Speeds

Importing the library

```
# Import pandas as pd
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline
plt.rcParams['figure.figsize'] = [10, 10]
```

Reading the data

```
df = pd.read_csv("/content/drive/MyDrive/Data Science/roller_coaster.csv")
```

```
df.shape
```

```
(1000, 15)
```

	Coaster name	Length	Speed	Location	Status	Opening date	Type	Manufacturer	Height restriction	Model	Height	Inversions	Duration	Capacity	G-force	Designer
0	Hyperion	762.0	95.0	Scandinavia Park, Sweden	Operating	June 30, 2001	Steel	Vekoma	No	Custom	76.0	0	00:01:40	1,000	3.0	Rolf Sachsse
1	Gwazi	1,676.0	80.0	Busch Gardens Tampa Bay	Operating	May 31, 2000	Wood	Gravity Group	Yes	Custom	50.0	0	00:02:00	1,000	3.0	Rolf Sachsse
2	Red Tail Ride	1,000.0	56.0	Pleasure Island	UNK	Unknown	Unknown	Intamin	Yes	Custom	14.0	0	00:01:00	1,000	2.0	Unknown
3	Cyclone	1,000.0	64.0	Coney Island, New York, United States	Operating	June 26, 1927	Steel	Vekoma	Yes	Custom	58.0	0	00:01:00	1,000	3.0	Unknown
4	Cyclone Mine Ride	1,000.0	64.0	Coney Island, New York, United States	Operating	June 26, 1927	Steel	Vekoma	Yes	Custom	58.0	0	00:01:00	1,000	3.0	Unknown
5	Cyclone Mine Ride	1,000.0	64.0	Coney Island, New York, United States	Operating	June 26, 1927	Steel	Vekoma	Yes	Custom	58.0	0	00:01:00	1,000	3.0	Unknown
6	Cyclone	1,000.0	64.0	Coney Island, New York, United States	Operating	June 26, 1927	Steel	Vekoma	Yes	Custom	58.0	0	00:01:00	1,000	3.0	Unknown
7	Cyclone	1,000.0	64.0	Coney Island, New York, United States	Operating	June 26, 1927	Steel	Vekoma	Yes	Custom	58.0	0	00:01:00	1,000	3.0	Unknown
8	Cyclone	1,000.0	64.0	Coney Island, New York, United States	Operating	June 26, 1927	Steel	Vekoma	Yes	Custom	58.0	0	00:01:00	1,000	3.0	Unknown
9	Cyclone	1,000.0	64.0	Coney Island, New York, United States	Operating	June 26, 1927	Steel	Vekoma	Yes	Custom	58.0	0	00:01:00	1,000	3.0	Unknown

1000 rows × 15 columns

```
df.dtype
```

```
KeyAttributeerror: 'DataFrame' object has no attribute 'dtype'
```

```
df.dtypes
```

```
Coaster name                   object
Length                         object
Speed                          object
Location                       object
Status                         object
Opening date                   object
Type                           object
Manufacturer                   object
Height restriction             object
Model                          object
Height                         object
Inversions                     object
Duration                       object
Capacity                       object
G-force                        object
Designer                       object
dtype: object
```

To see the first rows

```
df.head()
```

	Coaster name	Length	Speed	Location	Status	Opening date	Type	Manufacturer	Height restriction	Model	Height	Inversions	Duration	Capacity	G-force	Designer
0	Hyperion	762.0	95.0	Scandinavia Park, Sweden	Operating	June 30, 2001	Steel	Vekoma	No	Custom	76.0	0	00:01:40	1,000	3.0	Rolf Sachsse
1	Gwazi	1,676.0	80.0	Busch Gardens Tampa Bay	Operating	May 31, 2000	Wood	Gravity Group	Yes	Custom	50.0	0	00:02:00	1,000	3.0	Rolf Sachsse
2	Red Tail Ride	1,000.0	56.0	Pleasure Island	UNK	Unknown	Unknown	Intamin	Yes	Custom	14.0	0	00:01:00	1,000	2.0	Unknown
3	Cyclone	1,000.0	64.0	Coney Island, New York, United States	Operating	June 26, 1927	Steel	Vekoma	Yes	Custom	58.0	0	00:01:00	1,000	3.0	Unknown
4	Cyclone Mine Ride	1,000.0	64.0	Coney Island, New York, United States	Operating	June 26, 1927	Steel	Vekoma	Yes	Custom	58.0	0	00:01:00	1,000	3.0	Unknown

To see the columns

```
df.columns
```

```
Index(['Coaster name', 'Length', 'Speed', 'Location', 'Status',
       'Opening date', 'Type', 'Manufacturer', 'Height restriction',
       'Model', 'Height', 'Inversions', 'Duration', 'Capacity',
       'G-force', 'Designer'],
      dtype='object')
```

To check the datatypes of the columns

```
df.dtypes
```

```
Coaster name                   object
Length                         object
Speed                          object
Location                       object
Status                         object
Opening date                   object
Type                           object
Manufacturer                   object
Height restriction             object
Model                          object
Height                         object
Inversions                     object
Duration                       object
Capacity                       object
G-force                        object
Designer                       object
dtype: object
```

To describe the statistics about the Numeric data

```
df.describe()
```

	Inversions	Inversions	Inversions	Inversions	Inversions	Inversions	Inversions	Inversions
count	1,000.0	1,000.0	1,000.0	1,000.0	1,000.0	1,000.0	1,000.0	1,000.0
mean	1,070.0	1,000.0000000000000	61,000.0000000000000	61,000.0000000000000	1,000.0000000000000	1,000.0000000000000	1,000.0000000000000	1,000.0000000000000
std	1,110.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
min	1,000.0	1,000.0000000000000	61,000.0000000000000	61,000.0000000000000	1,000.0000000000000	1,000.0000000000000	1,000.0000000000000	1,000.0000000000000
25%	1,000.0000000000000	1,000.0000000000000	61,000.0000000000000	61,000.0000000000000	1,000.0000000000000	1,000.0000000000000	1,000.0000000000000	1,000.0000000000000
50%	1,000.0000000000000	1,000.0000000000000	61,000.0000000000000	61,000.0000000000000	1,000.0000000000000	1,000.0000000000000	1,000.0000000000000	1,000.0000000000000
75%	1,000.0000000000000	1,000.0000000000000	61,000.0000000000000	61,000.0000000000000	1,000.0000000000000	1,000.0000000000000	1,000.0000000000000	1,000.0000000000000
max	1,000.0000000000000	1,000.0000000000000	61,000.0000000000000	61,000.0000000000000	1,000.0000000000000	1,000.0000000000000	1,000.0000000000000	1,000.0000000000000

Data Preparation or Cleaning

Removing the unwanted columns by subsetting

```
df = df[['Coaster name',
         'Length',
         'Speed',
         'Location',
         'Status',
         'Opening date',
         'Type',
         'Manufacturer',
         'Inversions',
         'Duration',
         'Capacity',
         'G-force',
         'Height restriction',
         'Model',
         'Height',
         'Designer']]
df.head(1)
df.columns
df.shape
```

```
Index(['Coaster name', 'Length', 'Speed', 'Location', 'Status',
       'Opening date', 'Type', 'Manufacturer', 'Inversions', 'Duration',
       'Capacity', 'G-force', 'Height restriction', 'Model', 'Height',
       'Designer'],
      dtype='object')
(1000, 16)
```

Example of dropping single columns

```
df.drop('Opening date', axis=1, inplace=True)
```

```
KeyAttributeerror: 'DataFrame' object has no attribute 'dtypes'
```

```
df.dtypes
```

```
Coaster name                   object
Length                         object
Speed                          object
Location                       object
Status                         object
Type                           object
Manufacturer                   object
Inversions                     object
Duration                       object
Capacity                       object
G-force                        object
Height restriction             object
Model                          object
Height                         object
Designer                       object
dtype: object
```

converting dataype of 'opening_date_clean' from 'obj' to 'datetime64'

```
df['Opening date clean'].astype('datetime64[M]')
```

```
0    2001-06-01
1    2000-05-01
2    2000-01-01
3    1927-06-01
4    1927-06-01
5    1927-06-01
6    1927-06-01
7    1927-06-01
8    1927-06-01
9    1927-06-01
dtype: datetime64[ns]
```

To Rename the columns

```
df.columns
df.rename(columns={'Coaster name': 'Coaster_Name',
                   'Length': 'Length_in_meters',
                   'Speed': 'Speed_in_mph',
                   'Location': 'Location',
                   'Status': 'Status',
                   'Type': 'Type',
                   'Manufacturer': 'Manufacturer',
                   'Inversions': 'Inversions',
                   'Duration': 'Duration',
                   'Capacity': 'Capacity',
                   'G-force': 'G_force',
                   'Height restriction': 'Height_restriction',
                   'Model': 'Model',
                   'Height': 'Height',
                   'Designer': 'Designer',
                   'Opening date clean': 'Opening_date_clean'},
          inplace=True)
```

```
df.head()
```

	Coaster_Name	Length_in_meters	Speed_in_mph	Location	Status	Opening_date_clean	Type	Manufacturer	Inversions	Duration	Capacity	G_force	Height_restriction	Model	Height	Designer
0	Hyperion	762.0	95.0	Scandinavia Park, Sweden	Operating	2001-06-01	Steel	Vekoma	0	00:01:40	1,000	3.0	No	Custom	76.0	Rolf Sachsse
1	Gwazi	1,676.0	80.0	Busch Gardens Tampa Bay	Operating	2000-05-01	Wood	Gravity Group	0	00:02:00	1,000	3.0	Yes	Custom	50.0	Rolf Sachsse
2	Red Tail Ride	1,000.0	56.0	Pleasure Island	UNK	2000-01-01	Unknown	Intamin	0	00:01:00	1,000	2.0	Yes	Custom	14.0	Unknown
3	Cyclone	1,000.0	64.0	Coney Island, New York, United States	Operating	1927-06-01	Steel	Vekoma	0	00:01:00	1,000	3.0	Yes	Custom	58.0	Unknown
4	Cyclone Mine Ride	1,000.0	64.0	Coney Island, New York, United States	Operating	1927-06-01	Steel	Vekoma	0	00:01:00	1,000	3.0	Yes	Custom	58.0	Unknown

To identify the Missing values/ or 'Null values'

```
df.isnull() # to check which single cell contains the 'Null' values
```

	Coaster_Name	Length_in_meters	Speed_in_mph	Location	Status	Opening_date_clean	Type	Manufacturer	Inversions	Duration	Capacity	G_force	Height_restriction	Model	Height	Designer
0	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
995	False	False	False	True	False	False	False	False	False	False	False	False	False	False	False	False
996	False	False	False	True	False	False	False	False	False	False	False	False	False	False	False	False
997	False	False	False	True	False	False	False	False	False	False	False	False	False	False	False	False
998	False	False	False	True	False	False	False	False	False	False	False	False	False	False	False	False
999	False	False	False	True	False	False	False	False	False	False	False	False	False	False	False	False

1000 rows × 16 columns

To see the number of 'Null' values per column

```
df.isnull().sum()
```

```
Coaster_Name                   0
Length_in_meters               0
Speed_in_mph                   0
Location                       0
Status                         0
Opening_date_clean             0
Type                           0
Manufacturer                   0
Inversions                     0
Duration                       0
Capacity                       0
G_force                        0
Height_restriction             0
Model                          0
Height                         0
Designer                       0
dtype: int64
```

To identify the duplicate values

```
df.duplicated()
```

```
0    False
1    False
2    False
3    False
4    False
...  ...
995  False
996  False
997  False
998  False
999  False
dtype: bool
```

```
df.isnull.duplicated()
```

```
Coaster_Name  Length_in_meters  Speed_in_mph  Location  Status  Opening_date_clean  Type  Manufacturer  Inversions  Duration  Capacity  G_force  Height_restriction  Model  Height  Designer
False         False             False         False     False   False               False False         False       False     False     False    False               False  False   False
dtype: bool, dtype: bool
```

To drop the duplicates in the overall table

```
df.drop_duplicates()
```

To run duplicated on the subset of columns

```
df.duplicated(subset=['Coaster_Name'])
```

```
0    False
1    False
2    False
3    False
4    False
...  ...
995  False
996  False
997  False
998  False
999  False
dtype: bool
```

To identify the duplicates in a particular column (i.e) 'Coaster_Name'

```
df.duplicated(subset=['Coaster_Name'])
df[df.duplicated(subset='Coaster_Name')]
```

	Coaster_Name	Length_in_meters	Speed_in_mph	Location	Status	Opening_date_clean	Type	Manufacturer	Inversions	Duration	Capacity	G_force	Height_restriction	Model	Height	Designer
30	Steel Vengeance	1,158.0	75.0	Cedar Point, Sandusky, Ohio, United States	Operating	2018-05-26	Hybrid	Rocky Mountain Construction	0	00:03:00	2,400	4.0	Yes	Custom	62.0	Unknown
31	Maverick	1,000.0	70.0	Cedar Point, Sandusky, Ohio, United States	Operating	2007-05-26	Steel	Intamin	0	00:02:00	1,000	4.0	Yes	Custom	65.0	Unknown
32	Blue Streak (Cedaron park)	1,000.0	50.0	Cedaron park	Closed	May 31, 1964	Wood	Dinn Corporation	0	00:01:00	1,000	2.0	No	Custom	40.0	Unknown
33	Blue Streak (Cedaron park)	1,000.0	50.0	Cedaron park	Closed	May 31, 1964	Wood	Dinn Corporation	0	00:01:00	1,000	2.0	No	Custom	40.0	Unknown
34	Pirate Ride (Cedaron park)	1,000.0	50.0	Cedaron park	Operating	May 31, 1964	Wood	Dinn Corporation (California)	0	00:01:00	1,000	2.0	No	Custom	40.0	Unknown
35	Pirate Ride (Kennedy Park)	1,000.0	50.0	Kennedy Park, Lakewood, Ohio, United States	Operating	May 31, 1964	Wood	Dinn Corporation	0	00:01:00	1,000	2.0	No	Custom	40.0	Unknown
36	Lil Devil Coaster	1,000.0	50.0	Fun Forge Kinder Kingdom	Operating	UNK	Steel	Pinfari Ferrari Ferrone Family	0	00:01:00	1,000	2.0	Yes	Custom	40.0	Unknown
37	Little Dipper (Kennedy Park)	1,000.0	50.0	Kennedy Park	Operating	1950s	Wood	Dinn Corporation	0	00:01:00	1,000	2.0	Yes	Custom	40.0	Unknown
38	Hot Shoppe	1,000.0	50.0	Kennedy's Hickory Park, Lakewood, Ohio, United States	Operating	1950s	Wood	Dinn Corporation	0	00:01:00	1,000	2.0	Yes	Custom	40.0	Unknown
39	Ghost Train	1,000.0	50.0	Lake Compounce, Bristol, Connecticut, United States	Operating	1950s	Wood	Dinn Corporation	0	00:01:00	1,000	2.0	Yes	Custom	40.0	Unknown
40	Ghost Train	1,000.0	50.0	Ohio	Unknown	June 30, 1949	Wood	Dinn Corporation	0	00:01:00	1,000	2.0	Yes	Custom	40.0	Unknown

40 rows × 16 columns

Checking an example duplicate

To check how many times a particular duplicate is repeated in 'Coaster_Name' column

```
df.groupby('Coaster_Name').filtered.count()
```

Coaster_Name	Length	Speed	Location	Status	Opening_date_clean	Type	Manufacturer	Inversions	Duration	Capacity	G_force	Height_restriction	Model	Height	Designer
0 Crystal Beach Cyclone	Crystal Beach Park, Fort Erie, Ontario, Canada	Removed	1925	Steel Engineering	1927, 1948-1951	1,158.0	Steel	2000-01-01	0	00:01:00	1,000	3.0	Yes	0	0.0
1 Crystal Beach Cyclone	Crystal Beach Park, Fort Erie, Ontario, Canada	Removed	1925	Steel Engineering	1927, 1948-1951	1,158.0	Steel	2000-01-01	0	00:01:00	1,000	3.0	Yes	0	0.0

```
df.columns
```

```
Index(['Coaster_Name', 'Length', 'Speed', 'Opening_date', 'Manufacturer',
       'Inversions', 'Duration', 'Capacity', 'G-force', 'Height restriction',
       'Model', 'Height', 'Designer', 'Opening_date_clean', 'Status',
       'Location', 'Type'],
      dtype='object')
```

To check the sum of the 'Coaster_Name' 'Location' 'Opening date'

```
df.duplicated(subset=['Coaster_Name', 'Location', 'Opening date'].sum()
```

```
1
```

```
df.duplicated(subset=['Coaster_Name', 'Location', 'Opening date'])
```

```
0    False
1    False
2    False
3    False
4    False
...  ...
995  False
996  False
997  False
998  False
999  False
dtype: bool
```

To take the non-duplicated from 'Coaster_Name','Location','Opening date' columns using '~' symbol

```
# df.drop(df.duplicated(subset=['Coaster_Name', 'Location', 'Opening date']), inplace=True # use this to make permanent change
# df.drop(df.duplicated(subset=['Coaster_Name', 'Location', 'Opening date']), inplace=True # use this to make permanent change
df.iloc[~df.duplicated(subset=['Coaster_Name', 'Location', 'Opening date'])].reset_index(drop=True).shape
```

```
(1000, 16)
```

Feature Understanding

```
df['Top_continent'] = pd.Series(country) # since country list is equivalent value and count the main unique values value
```

```
0    Europe
1    Europe
2    Europe
3    Europe
4    Europe
...  ...
995  Europe
996  Europe
997  Europe
998  Europe
999  Europe
Name: Top_continent, dtype: object
```

```
df.to_csv("roller_coaster.csv", index=False)
df.groupby('Top_continent').Coaster_Name.count().plot(kind='bar',title='Top Years Coasters Introduced')
```

```
<ipynb_checkpoint_0>:1: FutureWarning: The default value of regex will change from True to False in a future version.
<ipynb_checkpoint_0>:2: FutureWarning: Series.__getitem__ treating keys as positions is deprecated. In a future version, integer keys will be treated as labels.
```

```
<Axes: Top_continent>
```

To see the Coaster speed using Histogram

```
df.dropna(inplace=True)
df.groupby('Speed_in_mph').Coaster_Name.count.plot(kind='hist',xlabel='Coaster Speed (mph)')
```

```
<ipynb_checkpoint_0>:1: FutureWarning: The default value of regex will change from True to False in a future version.
<ipynb_checkpoint_0>:2: FutureWarning: Series.__getitem__ treating keys as positions is deprecated. In a future version, integer keys will be treated as labels.
```

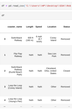
```
<Axes: Coaster Speed (mph)>
```


To see the Coaster speed using Kernel density plot

```
df.dropna(inplace=True)
df.groupby('Speed_in_mph').Coaster_Name.count.plot(kind='kde',xlabel='Coaster Speed (mph)')
```

```
<ipynb_checkpoint_0>:1: FutureWarning: The default value of regex will change from True to False in a future version.
<ipynb_checkpoint_0>:2: FutureWarning: Series.__getitem__ treating keys as positions is deprecated. In a future version, integer keys will be treated as labels.
```

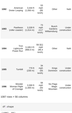
```
<Axes: Coaster Speed (mph)>
```



Feature Relationships

Plotting 'speed_mph' and 'Height_ft' in scatter plot

```
df.columns
df.plot(x='Speed_in_mph', y='Height_in_meters')
plt.xlabel('Speed')
plt.ylabel('Height')
```


Coaster Speed vs. Height
Speed
Height
-200
-150
-100
-50
0
50
100
150
200
250
300
350
400
0.000
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0.008
0.009
0.010
0.011
0.012
0.013
0.014
0.015
0.016
0.017
0.018
0.019
0.020
0.021
0.022
0.023
0.024
0.025
0.026
0.027
0.028
0.029
0.030
0.031
0.032
0.033
0.034
0.035
0.036
0.037
0.038
0.039
0.040
0.041
0.042
0.043
0.044
0.045
0.046
0.047
0.048
0.049
0.050
0.051
0.052
0.053
0.054
0.055
0.056
0.057
0.058
0.059
0.060
0.061
0.062
0.063
0.064
0.065
0.066
0.067
0.068
0.069
0.070
0.071
0.072
0.073
0.074
0.075
0.076
0.077
0.078
0.079
0.080
0.081
0.082
0.083
0.084
0.085
0.086
0.087
0.088
0.089
0.090
0.091
0.092
0.093
0.094
0.095
0.096
0.097
0.098
0.099
0.100
0.101
0.102
0.103
0.104
0.105
0.106
0.107
0.108
0.109
0.110
0.111
0.112
0.113
0.114
0.115
0.116
0.117
0.118
0.119
0.120
0.121
0.122
0.123
0.124
0.125
0.126
0.127
0.128
0.129
0.130
0.131
0.132
0.133
0.134
0.135
0.136
0.137
0.138
0.139
0.140
0.141
0.142
0.143
0.144
0.145
0.146
0.147
0.148
0.149
0.150
0.151
0.152
0.153
0.154
0.155
0.156
0.157
0.158
0.159
0.160
0.161
0.162
0.163
0.164
0.165
0.166
0.167
0.168
0.169
0.170
0.171
0.172
0.173
0.174
0.175
0.176
0.177
0.178
0.179
0.180
0.181
0.182
0.183
0.184
0.185
0.186
0.187
0.188
0.189
0.190
0.191
0.192
0.193
0.194
0.195
0.196
0.197
0.198
0.199
0.200
0.201
0.202
0.203
0.204
0.205
0.206
0.207
0.208
0.209
0.210
0.211
0.212
0.213
0.214
0.215
0.216
0.217
0.218
0.219
0.220
0.221
0.222
0.223
0.224
0.225
0.226
0.227
0.228
0.229
0.230
0.231
0.232
0.233
0.234
0.235
0.236
0.237
0.238
0.239
0.240
0.241
0.242
0.243
0.244
0.245
0.246
0.247
0.248
0.249
0.250
0.251
0.252
0.253
0.254
0.255
0.256
0.257
0.258
0.259
0.260
0.261
0.262
0.263
0.264
0.265
0.266
0.267
0.268
0.269
0.270
0.271
0.272
0.273
0.274
0.275
0.276
0.277
0.278
0.279
0.280
0.281
0.282
0.283
0.284
0.285
0.286
0.287
0.288
0.289
0.290
0.291
0.292
0.293
0.294
0.295
0.296
0.297
0.298
0.299
0.300
0.301
0.302
0.303
0.304
0.305
0.306
0.307
0.308
0.309
0.310
0.311
0.312
0.313
0.314
0.315
0.316
0.317
0.318
0.319
0.320
0.321
0.322
0.323
0.324
0.325
0.326
0.327
0.328
0.329
0.330
0.331
0.332
0.333
0.334
0.335
0.336
0.337
0.338
0.339
0.340
0.341
0.342
0.343
0.344
0.345
0.346
0.347
0.348
0.349
0.350
0.351
0.352
0.353
0.354
0.355
0.356
0.357
0.358
0.359
0.360
0.361
0.362
0.363
0.364
0.365
0.366
0.367
0.368
0.369
0.370
0.371
0.372
0.373
0.374
0.375
0.376
0.377
0.378
0.379
0.380
0.381
0.382
0.383
0.384
0.385
0.386
0.387
0.388
0.389
0.390
0.391
0.392
0.393
0.394
0.395
0.396
0.397
0.398
0.399
0.400
0.401
0.402
0.403
0.404
0.405
0.406
0.407
0.408
0.409
0.410
0.411
0.412
0.413
0.414
0.415
0.416
0.417
0.418
0.419
0.420
0.421
0.422
0.423
0.424
0.425
0.426
0.427
0.428
0.429
0.430
0.431
0.432
0.433
0.434
0.435
0.436
0.437
0.438
0.439
0.440
0.441
0.442
0.443
0.444
0.445
0.446
0.447
0.448
0.449
0.450
0.451
0.452
0.453
0.454
0.455
0.456
0.457
0.458
0.459
0.460
0.461
0.462
0.463
0.464
0.465
0.466
0.467
0.468
0.469
0.470
0.471
0.472
0.473
0.474
0.475
0.476
0.477
0.478
0.479
0.480
0.481
0.482
0.483
0.484
0.485
0.486
0.487
0.488
0.489
0.490
0.491
0.492
0.493
0.494
0.495
0.496
0.497
0.498
0.499
0.500
0.501
0.502
0.503
0.504
0.505
0.506
0.507
0.508
0.509
0.510
0.511
0.512
0.513
0.514
0.515
0.516
0.517
0.518
0.519
0.520
0.521
0.522
0.523
0.524
0.525
0.526
0.527
0.528
0.529
0.530
0.531
0.532
0.533
0.534
0.535
0.536
0.537
0.538
0.539
0.540
0.541
0.542
0.543
0.544
0.545
0.546
0.547
0.548
0.549
0.550
0.551
0.552
0.553
0.554
0.555
0.556
0.557
0.558
0.559
0.560
0.561
0.562
0.563
0.564
0.565
0.