Programming for Data Analysis - Project

Abstract

This Jupyter Python notebook explores a publicly-available dataset originating from the biennial Critical Assessment of protein Structure Prediction (CASP) competition, Series 5-9. After first performing exploratory data analysis on the dataset, different Python libraries are used to create a synthetic dataset. To this end, random numbers was generated to follow the distribution of each original attribute. Lastly, a visual comparison is made between the original and the synthetic dataset to evaluate the accurary of the entire data synthesis process. To provide reproducible results, the outputs of the synthesized values have been written as *comma-separated values* files.

Table of Contents

1. Introduction
2. Exploratory Data Analysis (EDA)
3. Methods for Data Synthesis / Simulation
4. Generation of a synthetic dataset
5. RMSD
6. Total Surface Area
7. Non-polar exposed area
8. Fractional area of exposed non polar residue
9. Fractional area of exposed non polar part of residue
10. Molecular mass weighted exposed area
11. Average deviation from standard exposed area of residue
12. Euclidian distance
13. Secondary structure penalty
14. Spatial distribution constraints
15. Discussion
16. Conclusion

Introduction

Proteins are complex molecules (macromolecules) responsible for performing a multitude of biological functions. These include:

Protein name	Function
Keratin	Component of hair and nail
Collagen	Component of skin
Insulin	Responsible for regulating sugar and fat level

An important aspect in the field of protein biochemistry is understanding how protein structures relate to their molecular function - which led to the emergence of the structural biology field. This active area of research applies experimental techniques such as X-ray (https://en.wikipedia.org/wiki/X-ray_crystallography) and neutron crystallography (https://en.wikipedia.org/wiki/Neutron_diffraction), and electron microscopy (https://en.wikipedia.org/wiki/Electron_microscope) to determine the three-dimensional structures of protein such that questions on their evolutionary relationships and exploitability for healthcare can be answered.

However, experimental techniques suffer from inherent drawbacks - they require highly-pure and stable physical samples that can be difficult to obtain in sufficient quantities, as well as human expertise to convert data to three-dimensional models that can be instinctively interpreted by researchers from other fields. Computational biology approaches are one way to overcome these drawbacks. By making certain assumptions, we can use computers to solve structures (https://en.wikipedia.org/wiki/Protein_structure_prediction). For instance, having solved one protein structure, we can predict that another similar protein will likely have the same overall structure, albeit with minor differences (homology modeling). Similar to other areas, machine learning techniques have been applied for protein structure prediction (1 (https://www.ncbi.nlm.nih.gov/pubmed/22274898) and 2 (https://www.nature.com/articles/srep18962)).

Alternatively, we can also use fundamental physics to note that a certain protein sequence will only fold into a particular structure as there will be attraction and repulsion forces among the different atoms (*de novo* prediction (https://en.wikipedia.org/wiki/De_novo_protein_structure_prediction)). The idea that there is no random sampling of all possible conformations during the process of protein folding is captured by Levinthal's paradox (https://en.wikipedia.org/wiki/Levinthal%27s_paradox), where it was postulated that random sampling leading to the correct conformation would take more time than the age of the universe.

In order to judge the performance of various algorithms, CASP (http://predictioncenter.org/) has been organized by the structural biology community as an open venue to learn from each other as often no one method is the best for *all* types of problems. Complementary techniques, including the use of lower resolution (less informative) tools such as circular dichroism, can also help to determine a model that is as close to the actual structure as possible.

In this notebook, we will look at the CASP dataset (obtained from the UCI Machine Learning Repository (http://archive.ics.uci.edu/ml/datasets/Physicochemical+Properties+of+Protein+Tertiary+Structure)) and understand the relationship among all the variables and their individual distribtions before using tools in Python to generate a synthetic dataset that mimicks the experimental values. For reference, there are two publications related to the use of this dataset for machine learning and simulation (1 (https://www.researchgate.net/publication/310441738_Tight_Clustering_for_Large_Datasets_with_an_Application_to_Microarray_Data) and 2 (https://dl.acm.org/citation.cfm?id=3029587)).

In order to not reinvent the wheel, we will use pre-existing Python libraries to achieve our goal.

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
# %matplotlib notebook
%matplotlib inline
import pandas as pd
import seaborn as sns

# to make interactive plots with plotly
import plotly.figure_factory as ff
import plotly.graph_objs as go
from plotly.offline import download_plotlyjs, init_notebook_mode, iplot
init_notebook_mode(connected=True)
from plotly import tools

# original fitter is here: https://pypi.org/project/fitter/
# but we use https://github.com/caiostringari/fitter/tree/master
# (to install: pip install --upgrade https://github.com/caiostringari/fitter/tarball/master)
from fitter import Fitter

from pylab import *
from scipy.optimize import curve_fit
import scipy.interpolate as interpolate

# import scipy distributions that are needed later
from scipy.stats import moyal, exponnorm, johnsonsu, norm, genlogistic, invgauss, burr, mielke, t, fisk

# to make plotly images - https://plot.ly/python/static-image-export/
import plotly.io as pio
from IPython.display import Image
# https://community.plot.ly/t/use-plotly-offline-to-save-chart-as-image-file/408/41
pio.orca.config.port = 80 # in case firewall blocks access to the default port of 39167
```

Exploratory Data Analysis (EDA)

We will first look at the data structure and make modifications as required.

In [11]:
```
df = pd.read_csv('CASP.csv')
df
```

Out[11]:

	RMSD	F1	F2	F3	F4	F5	F6	F7	F8	F9
0	17.284	13558.30	4305.35	0.31754	162.1730	1.872791e+06	215.3590	4287.87	102	27.0302
1	6.021	6191.96	1623.16	0.26213	53.3894	8.034467e+05	87.2024	3328.91	39	38.5468
2	9.275	7725.98	1726.28	0.22343	67.2887	1.075648e+06	81.7913	2981.04	29	38.8119
3	15.851	8424.58	2368.25	0.28111	67.8325	1.210472e+06	109.4390	3248.22	70	39.0651
4	7.962	7460.84	1736.94	0.23280	52.4123	1.021020e+06	94.5234	2814.42	41	39.9147
5	1.700	5117.30	1120.99	0.21905	51.6732	6.727227e+05	79.5911	3234.21	15	41.2382
6	9.314	5924.16	1625.27	0.27434	70.2103	8.285145e+05	76.8064	2821.40	70	39.4964
7	1.985	6882.15	1791.22	0.26027	77.2501	9.165165e+05	96.6785	3490.88	74	37.4203
8	1.915	12090.00	4190.74	0.34662	129.0020	1.687508e+06	186.3090	4262.78	39	30.3916
9	1.495	7400.24	1881.95	0.25430	82.9320	1.023846e+06	104.6970	3852.40	26	35.4140
10	12.118	6556.77	1612.77	0.24597	71.6315	8.915443e+05	93.5329	3161.33	76	38.0433
11	0.884	8828.21	2658.63	0.30115	90.8578	1.233384e+06	123.6860	3194.30	22	37.2413
12	7.913	5637.37	2665.83	0.47288	49.8566	7.716355e+05	95.7431	2177.61	7	41.5268
13	14.103	9021.10	3097.91	0.34340	98.1155	1.244160e+06	121.3480	3396.40	32	37.0667
14	6.581	17572.20	5226.42	0.29742	227.7690	2.434431e+06	296.8000	4876.00	122	26.8169
15	6.110	12330.60	4380.20	0.35523	195.9300	1.710155e+06	232.1540	4321.41	164	30.4491
16	1.339	8478.55	2206.94	0.26029	91.5054	1.199770e+06	128.1660	3996.13	164	33.5263
17	2.554	9147.09	2828.26	0.30919	104.0500	1.291447e+06	131.3580	3595.69	17	35.5938
18	3.805	3905.51	1102.49	0.28229	34.2034	5.643988e+05	48.1646	1834.36	13	44.5123
19	18.645	15620.90	5834.18	0.37348	153.4760	2.184645e+06	251.2250	5441.50	101	26.0101
20	8.114	18679.00	5966.18	0.31940	179.7570	2.650897e+06	268.6610	6149.89	213	23.8081
21	18.404	13605.50	4766.37	0.35032	139.7930	1.925985e+06	182.1950	4672.34	112	26.4540
22	19.379	8693.56	3265.99	0.37567	55.4973	1.258363e+06	126.9280	4890.54	15	36.3849
23	17.598	7727.36	1370.27	0.17732	77.5656	1.035614e+06	103.1770	3130.45	27	37.1829
24	16.102	8865.32	3317.32	0.37419	95.8075	1.223688e+06	122.7480	3509.39	38	36.7233
25	14.923	8259.29	2719.28	0.32923	76.1237	1.087133e+06	122.1430	3763.75	58	35.5364
26	1.655	7136.22	2645.28	0.37068	64.1673	1.032920e+06	93.2759	3213.99	28	34.6186
27	4.128	6922.15	2143.02	0.30958	63.6012	9.896310e+05	91.6472	3146.95	31	38.8427
28	1.477	6449.78	1439.30	0.22315	74.7722	8.497845e+05	90.5322	3393.06	80	38.2100
29	2.165	13274.30	3989.12	0.30051	169.7460	1.855448e+06	198.6440	5089.76	67	26.6638
...	...	...	...	...	...	...	...	...	...	...
45700	0.966	23403.50	5511.79	0.23551	319.1290	3.224178e+06	407.9570	6223.57	154	19.8365
45701	5.546	18403.60	4882.31	0.26529	245.5710	2.559338e+06	294.5570	5457.64	215	25.6038
45702	3.581	13696.00	4088.79	0.29853	145.3500	1.844331e+06	211.9420	4993.11	53	28.1334
45703	1.661	6405.36	2182.59	0.34074	72.0344	8.836039e+05	92.7146	3203.00	37	40.2693
45704	18.522	10840.30	5689.76	0.52487	85.2052	1.355167e+06	207.9200	2954.30	70	38.2021
45705	18.875	11215.80	3814.74	0.34012	102.4110	1.537030e+06	179.7740	4316.51	105	34.1568
45706	17.150	9779.31	3687.08	0.37702	82.2241	1.397476e+06	144.7080	4094.08	69	31.6338
45707	13.678	6233.61	846.29	0.13576	52.7953	8.656223e+05	71.4992	2733.96	45	34.9193
45708	4.758	6458.52	1380.97	0.21382	64.5865	8.980187e+05	72.4505	3220.17	34	38.0233
45709	1.788	6226.42	1551.74	0.24921	67.6087	8.793935e+05	74.1427	3131.57	14	38.9950
45710	3.509	13867.40	4320.27	0.31154	167.1120	1.923652e+06	216.2990	4917.21	72	28.5099
45711	1.572	8716.67	2369.97	0.27188	83.0696	1.222297e+06	116.8790	4092.53	49	33.9539
45712	8.221	9893.75	2426.80	0.24528	87.7843	1.396274e+06	124.5970	3957.71	153	33.4454
45713	12.684	7353.75	2471.99	0.33615	50.5474	1.047082e+06	107.5370	2965.32	68	39.5696
45714	17.180	9616.40	3410.33	0.35463	99.8179	1.353932e+06	150.5450	4188.42	60	35.1050
45715	0.984	9725.16	1990.64	0.20468	104.3870	1.372349e+06	138.4390	4324.37	50	30.6403
45716	3.619	10153.40	3390.35	0.33391	83.1698	1.410026e+06	154.4500	3871.40	33	34.1444
45717	13.697	6074.87	1322.94	0.21777	56.0178	8.186789e+05	70.0644	2471.63	38	41.4740
45718	11.283	3781.01	1149.39	0.30399	36.0219	5.128434e+05	47.6737	1391.45	32	44.4733
45719	4.540	5080.14	845.89	0.16650	54.8128	6.682106e+05	56.3293	2698.52	57	38.6640
45720	17.595	9583.84	2456.51	0.25631	57.4054	1.358302e+06	107.4720	4660.71	76	34.3923

From above, we can determine that we have 10 columns [RMSD (root mean square deviation) and F1-F9] of 45730 rows (data points) each. Except for the first column, the others are non-descriptive and should be renamed so avoid confusion later on.

For illustrative purposes, we do not really need all the 45730 rows, so it is possible to just use a random selection of about 50 % of the dataset using the `pandas.DataFrame.sample()` function. Because the Plotly plots are interactive and have the datapoints embedded within them, a smaller number of datapoints will allow for a smaller notebook size. A caveat of sampling is that it might result in an analysis not representative of the original dataset. Our strategy for handling a large dataset will be memory footprint optimization, [conversion of interactive plots to static images (https://plot.ly/python/static-image-export/)](https://plot.ly/python/static-image-export/) and large-enough random sampling.

To note, Plotly is capable of plotting hundreds of thousands of datapoints (big data) by using WebGL.

In [12]:
```
# rename columns based on information from the UCI webpage
# F1 - Total surface area.
# F2 - Non polar exposed area.
# F3 - Fractional area of exposed non polar residue.
# F4 - Fractional area of exposed non polar part of residue.
# F5 - Molecular mass weighted exposed area.
# F6 - Average deviation from standard exposed area of residue.
# F7 - Euclidian distance.
# F8 - Secondary structure penalty.
# F9 - Spacial Distribution constraints (N,K Value). (typo for Spatial?)
df.rename(columns={'F1': 'Total surface area',
                   'F2': 'Non-polar exposed area',
                   'F3': 'Fractional area of exposed non-polar residue',
                   'F4': 'Fractional area of exposed non-polar part of residue',
                   'F5': 'Molecular mass weighted exposed area',
                   'F6': 'Average deviation from standard exposed area of residue',
                   'F7': 'Euclidian distance',
                   'F8': 'Secondary structure penalty',
                   'F9': 'Spatial Distribution constraints (N,K Value)'
                  }, inplace = True)

# if required, we can use just a subset of the dataset for faster plotting, fitting etc.
# df = df.iloc[:10000] # takes first 10000 rows
df = df.sample(n=int(len(df)/2), random_state=1) # random sample size of 50 % of the original df, seed=
1
```

When dealing with large datasets, it is important to minimize the memory footprint. Several steps can be considered, but the most important one is to ensure the correct usage of datatypes, which can be examined using the `info()` function from Pandas.

In [101]:
```
df.info(memory_usage='deep')
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 22865 entries, 13171 to 15355
Data columns (total 10 columns):
RMSD                                                  22865 non-null float64
Total surface area                                    22865 non-null float64
Non-polar exposed area                                22865 non-null float64
Fractional area of exposed non-polar residue          22865 non-null float64
Fractional area of exposed non-polar part of residue  22865 non-null float64
Molecular mass weighted exposed area                  22865 non-null float64
Average deviation from standard exposed area of residue  22865 non-null float64
Euclidian distance                                    22865 non-null float64
Secondary structure penalty                           22865 non-null int64
Spatial Distribution constraints (N,K Value)          22865 non-null float64
dtypes: float64(9), int64(1)
memory usage: 1.9 MB
```

As we can see above, Pandas has imported the CASP dataset using float64 for almost all the variables. This is unnecessary as the output of `describe()` (not shown) indicate that we are dealing with smaller, positive values only. The `to_numeric()` function (https://pandas.pydata.org/pandas-docs/stable/generated/pandas.to_numeric.html) with the *downcast* parameter will automatically select the most appropriate one.

In [6]:
```
df['RMSD'] = pd.to_numeric(df['RMSD'], downcast='float')
df['Total surface area'] = pd.to_numeric(df['Total surface area'], downcast='unsigned')
df['Non-polar exposed area'] = pd.to_numeric(df['Non-polar exposed area'], downcast='float')

df['Fractional area of exposed non-polar residue'] = pd.to_numeric(df['Fractional area of exposed non-p
olar residue'],
                                                                   downcast='float')
df['Fractional area of exposed non-polar part of residue'] = pd.to_numeric(
                                          df['Fractional area of exposed non-polar part of residue'],
downcast='float')

df['Molecular mass weighted exposed area'] = pd.to_numeric(df['Molecular mass weighted exposed area'],
downcast='float')

df['Average deviation from standard exposed area of residue'] = pd.to_numeric(
    df['Average deviation from standard exposed area of residue'], downcast='float')

df['Euclidian distance'] = pd.to_numeric(df['Euclidian distance'], downcast='float')
df['Secondary structure penalty'] = pd.to_numeric(df['Secondary structure penalty'], downcast='unsigned
')

df['Spatial Distribution constraints (N,K Value)'] = pd.to_numeric(
    df['Spatial Distribution constraints (N,K Value)'], downcast='float')
```

In [7]:
```
df.info(memory_usage='deep')
```

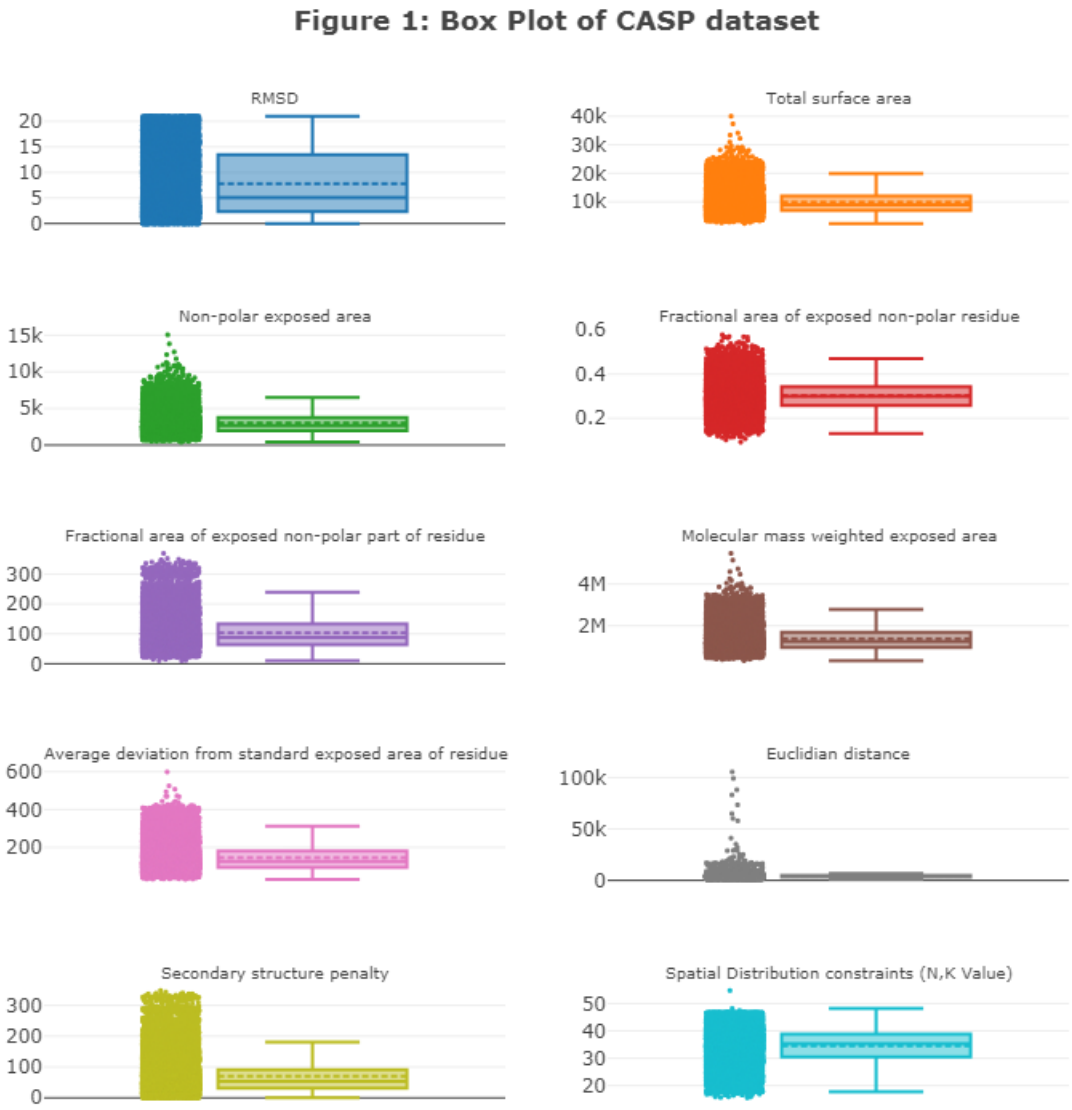
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 22865 entries, 36510 to 18037
Data columns (total 10 columns):
RMSD                                                  22865 non-null float32
Total surface area                                    22865 non-null float64
Non-polar exposed area                                22865 non-null float32
Fractional area of exposed non-polar residue          22865 non-null float32
Fractional area of exposed non-polar part of residue  22865 non-null float32
Molecular mass weighted exposed area                  22865 non-null float32
Average deviation from standard exposed area of residue  22865 non-null float32
Euclidian distance                                    22865 non-null float32
Secondary structure penalty                           22865 non-null uint16
Spatial Distribution constraints (N,K Value)          22865 non-null float32
dtypes: float32(8), float64(1), uint16(1)
memory usage: 1.1 MB
```

By optimizing the datatype, we have indeed halved the memory usage from 1.9 MB to 1.1 MB. Before simulating the experimental dataset, we will examine the distributions and identify any pathology before proceeding. For this, box plots will be used.

In [9]:

```
trace0 = go.Box(y=df.iloc[:,0], marker = dict(size=3), boxpoints='all', boxmean = True)
trace1 = go.Box(y=df.iloc[:,1], marker = dict(size=3), boxpoints='all', boxmean = True)
trace2 = go.Box(y=df.iloc[:,2], marker = dict(size=3), boxpoints='all', boxmean = True)
trace3 = go.Box(y=df.iloc[:,3], marker = dict(size=3), boxpoints='all', boxmean = True)
trace4 = go.Box(y=df.iloc[:,4], marker = dict(size=3), boxpoints='all', boxmean = True)
trace5 = go.Box(y=df.iloc[:,5], marker = dict(size=3), boxpoints='all', boxmean = True)
trace6 = go.Box(y=df.iloc[:,6], marker = dict(size=3), boxpoints='all', boxmean = True)
trace7 = go.Box(y=df.iloc[:,7], marker = dict(size=3), boxpoints='all', boxmean = True)
trace8 = go.Box(y=df.iloc[:,8], marker = dict(size=3), boxpoints='all', boxmean = True)
trace9 = go.Box(y=df.iloc[:,9], marker = dict(size=3), boxpoints='all', boxmean = True)

# we have to make subplots because the scales are different for the variables
fig = tools.make_subplots(rows=5, cols=2, print_grid=False,
                          subplot_titles= (df.columns[0], df.columns[1], df.columns[2], df.columns[3],
                                           df.columns[4], df.columns[5], df.columns[6], df.columns[7],
                                           df.columns[8], df.columns[9]
                                                          ))

fig.append_trace(trace0, 1, 1)
fig.append_trace(trace1, 1, 2)
fig.append_trace(trace2, 2, 1)
fig.append_trace(trace3, 2, 2)
fig.append_trace(trace4, 3, 1)

fig.append_trace(trace5, 3, 2)
fig.append_trace(trace6, 4, 1)
fig.append_trace(trace7, 4, 2)
fig.append_trace(trace8, 5, 1)
fig.append_trace(trace9, 5, 2)

fig['layout'].update(height=800, width=800, title='<b>Figure 1: Box Plot of CASP dataset</b>')
fig['layout'].update(showlegend=False)

# https://github.com/plotly/plotly.py/issues/985
for i in fig['layout']['annotations']:
    i['font'] = dict(size=10)

for prop in fig.layout:
    if prop.startswith('xaxis'):
        # don't really need both xaxis label and subplot title, so turn xaxis label off
        fig.layout[prop].showticklabels = False

config = {
'include_plotlyjs': False
}

# iplot(fig)
img_bytes = pio.to_image(fig, format='png')
Image(img_bytes)
```

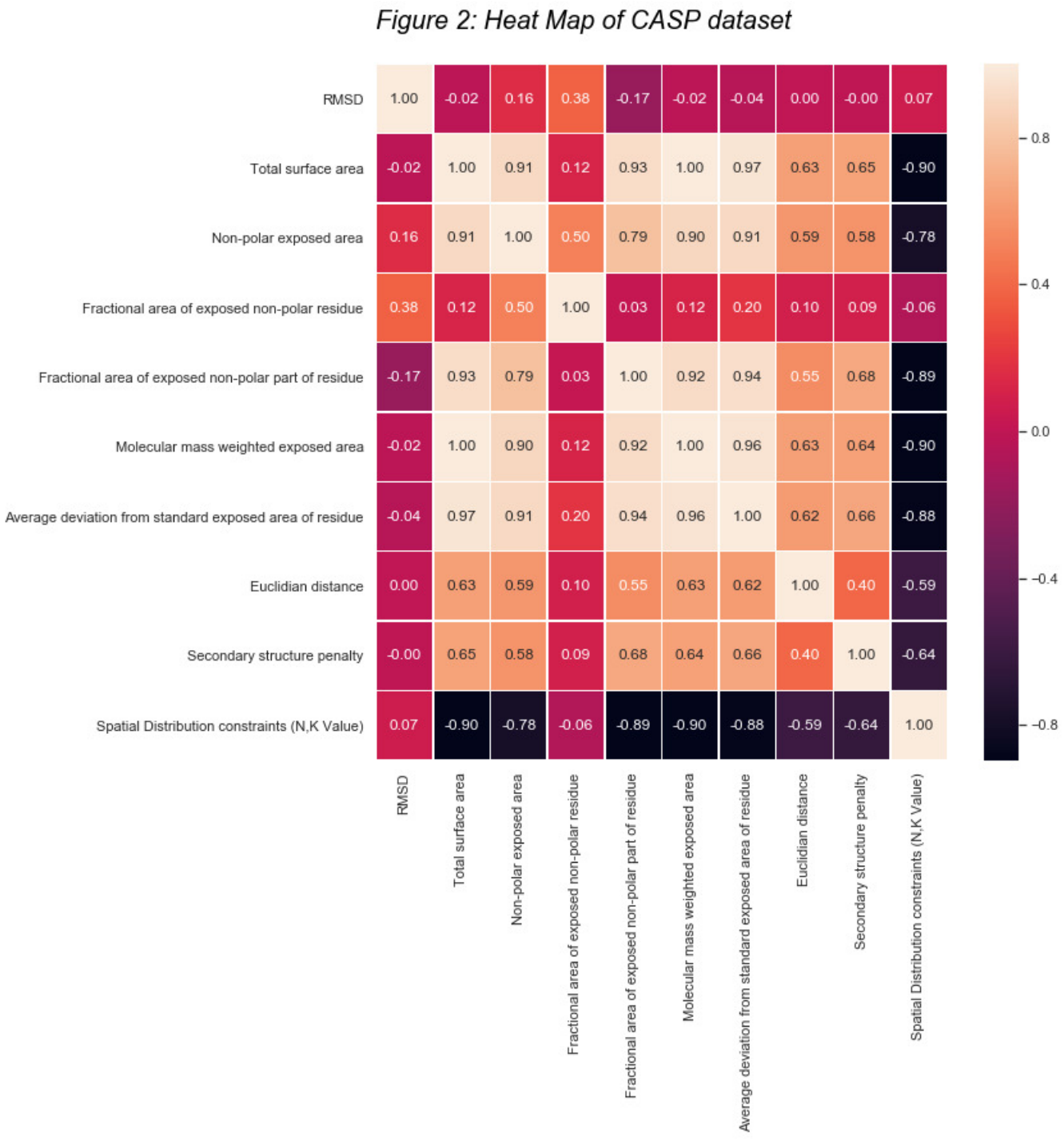
Out[9]:



The sampled dataset do not have many outliers, as shown by the box plots above. Most of these are present in the Euclidian distance variable. Moreover, the mean of almost all variables are very close to the median (middle 50 % of the dataset), including a normal-like distribution. An exception could be the RMSD variable.

Although a table containing the correlation coefficents between two variables for all the variables can be obtained by using `pandas.DataFrame.corr()` function, it is too 'numerical' for a good interpretation. A better option is to use Seaborn to construct a heat map based on the data from `pandas.DataFrame.corr()` .

In [106]:
```
sns.set(font_scale=1)
plt.figure(figsize=(10,10))
plt.text(0,-0.5, "Figure 2: Heat Map of CASP dataset", fontsize = 20, color='Black', fontstyle='italic
')
# plt.tight_layout()

# according to the docs https://seaborn.pydata.org/generated/seaborn.heatmap.html and
# http://alanpryorjr.com/visualizations/seaborn/heatmap/heatmap/
# we can use linewidths to create some spacing between the cells and
# fmt (format) to round 2 decimal points

# https://stackoverflow.com/questions/25790062/
sns.heatmap(df.corr(), annot=True, linewidths = 0.5, fmt='.2f');
```



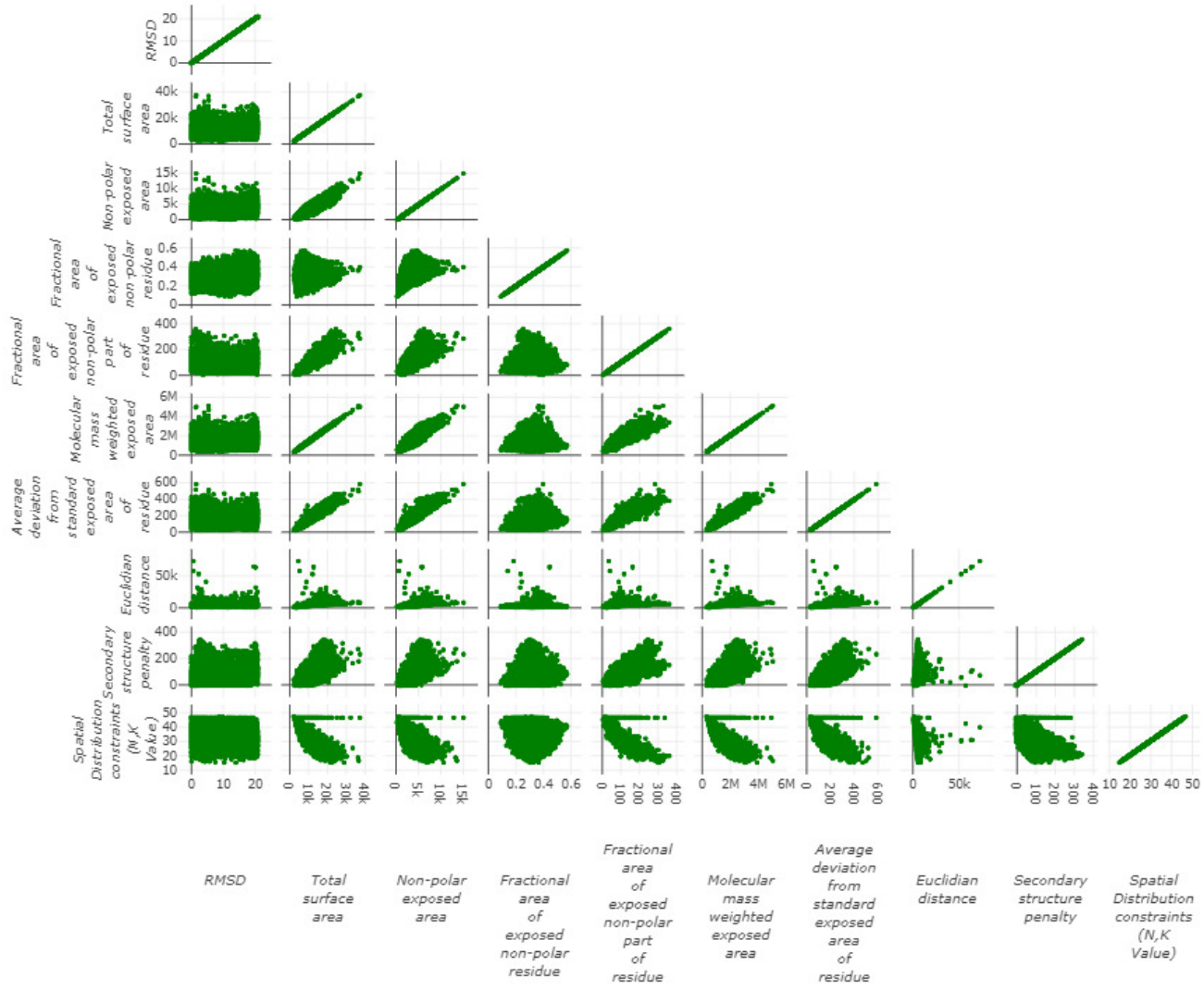
To complement the heat map, a scatterplot matrix can be employed for pairwise comparison.

In [107]:

```
# https://community.plot.ly/t/splom-scatter-matrix-changing-styles-of-all-axes-in-one-go/16636/2
# https://plot.ly/python/splom/

# initiate placeholders
dataPanda = []
label = ''

for column in list(df.columns):

    columnSplit = column.split(' ') # create a list of string from the column name

    for i in columnSplit:
        label = label + i + '<br>' # as plotly uses JavaScript, we can use the html <br> tag to split
line

    # inspired by: https://stackoverflow.com/questions/43229013/
    for i in range(3):
        label = '<br>' + label + '<br>'

    label = '<i>' + label + '</i>'
    trace = dict(label = label, values = df[column])

    dataPanda.append(trace)
    label = '' # reset the variable for next for iteration

trace1 = go.Splom(dimensions=dataPanda, showupperhalf=False, marker=dict(size=3, color='green'))
data = [trace1]

layout = go.Layout(title = '<b>Figure 3: Scatterplot Matrix of CASP dataset</b>',
                font=dict(size=8.5), # global font
                titlefont=dict(size=20), # but we want only title to have different size
                # layout control- left, right, bottom, top (t) and padding (pad )
                margin = dict(l = 150, r = 20, b = 150)
                )

fig = go.Figure(data=data, layout=layout)
fig['layout'].update(height=800, width=900)

# iplot(fig)
img_bytes = pio.to_image(fig, format='png')
Image(img_bytes)
```

Out[107]:

Figure 3: Scatterplot Matrix of CASP dataset



Based on the heatmap plot and the scatterplot matrix, we can make the following observations:

1. RMSD is poorly correlated with any of the other variables.
2. There is a strong positive correlation between Total surface area and non-polar exposed area, molecular mass weighted exposed area and average deviation from standard exposed area of residue.
3. The spatial distribution constraints appears to have a negative correlation with fractional area of exposed non-polar residue, molecular mass weighted exposed area and average deviation from standard exposed area of residue.

Based on the variable labels and the correlation coefficients above, it seems likely that some variables could be collinear, rather than truly correlated. For instance, the non-polar exposed area, fractional area of exposed non-polar residue and fractional area of exposed non-polar part of residue are probably calculations by related equations.

Methods for Data Synthesis / Simulation

In any case, we should look at the distribution of each variable. Scipy comes with built-in functions for probability distribution functions (PDF) for many well-known distributions. Each function takes usually two parameters - *scale* to control peak height and *loc* to control the mean (see the Jupyter notebook in the ProgrammingforDA-Assignment repository).

While we could manually fit each distribution for each variable, we can use the `fitter` module to automatically generate the curves for us. In essence, `fitter` uses Scipy curve_fit to perform non-linear least square fitting with initial parameters estimated from the data.

A synthetic dataset can be generated by the following methods:

1. For dataset related to a specific chemical or physical phenomenon, the relationship between the response and predictor variables can often be described as a mathematical equation (first principles and/or mechanistic model). Usually, the model takes the form of a differential equation. In this case, an error term referencing random numbers (noise) can be added to that model to yield a synthetic dataset.
2. For those that do not have a mathematical equation defined in the literature but follows a well-described and widely used distribution, we can use one of the 35+ `numpy.random` functions (https://docs.scipy.org/doc/numpy/reference/routines.random.html), for example `numpy.random.beta()`. Alternatively, a larger set of functions from `scipy.stats` (about 80 functions) (http://scipy.github.io/devdocs/stats.html) can be used, for example `scipy.stats.rvs()`.

   However, we need to know the parameters, such as μ and σ for the mean and standard deviation for a normal distribution, respectively, . This can be estimated by 'eye', trial-and-error or automated/manual curve fitting. In this project, the Fitter package (https://pythonhosted.org/fitter/) will be used. Fitter is essentially a wrapper module around `scipy.stats` as the probability distribution source and `scipy.optimize.curve_fit()` for non-linear least squares curve fitting, which can rapidly achieve convergence given initial parameter estimates that roughly match the actual parameter values for the corresponding probability distribution function (PDF).

   Note that the original GitHub repository for this package (https://github.com/cokelaer/fitter) was last updated about 2 years. Therefore, a different branch (https://github.com/caiostringari/fitter/tree/master) with more recent updates aligned to the latest Matplotlib library will be used. For installation, use the following command in shell: `pip install --upgrade https://github.com/caiostringari/fitter/tarball/master` (`--upgrade` can be omitted if installing for the first time).

   Simulating a dataset following a bimodal normal distribution is difficult with `numpy.random` but doable (https://stackoverflow.com/questions/4265988/generate-random-numbers-with-a-given-numerical-distribution).
3. For these difficult cases where there are no well-known distributions, inverse transform sampling (ITS) (http://www.nehalemlabs.net/prototype/blog/2013/12/16/how-to-do-inverse-transformation-sampling-in-scipy-and-numpy/) can be employed.
   A. Inverse transform sampling (https://en.wikipedia.org/wiki/Inverse_transform_sampling), in essence, is a method to rescale a uniform random variable (which is what numpy.random returns - see the Programming for Data Analysis Assignment) into a desired probability distribution.
   B. The cumulative distribution function (CDF) can be thought of as the integral of the data histogram [which itself is a (PDF)], giving us the percentage or the relative count plotted against the data (https://www.andata.at/en/software-blog-reader/why-we-love-the-cdf-and-do-not-like-histograms-that-much.html). CDF has a y axis of [0,1].
   C. Inverting the CDF allows the sampling of the uniform values because the y-axis is turned into an x-axis of [0,1], followed by a transformation into the target distribution.

Model selection

Oftentimes, replication of experimental distributions are not faithfully reproducible by standard probability distributions even with low error values (sum of the squared difference between the predicted and observed values). In other cases, there could be multiple suitable models with similar error values. From basic statistics, we know that fitting more complicated models come with the risk of fitting noise (for example, measurement errors in the experimental data).

Consequently, there is a need for a standardized method to compare models and choose the more parsimonious model. Parsimony in statistics (also Occam's razor (https://en.wikipedia.org/wiki/Occam%27s_razor)) refers to a model with fewer parameters that still describe the data adequately without missing the important subtleties of the data, as shown in the figure below.

In [16]:
```
# http://www.nehalemlabs.net/prototype/blog/2013/12/16/how-to-do-inverse-transformation-sampling-in-scipy-and-numpy/

# increase n_bins and samples to get a better sampling
def inverse_transform_sampling(data, n_bins = 200, n_samples = len(df)):

    # calculate a histogram over the data. density = True calc's a pdf
    hist, bin_edges = np.histogram(data, bins = n_bins, density=True)

    cum_values = np.zeros(bin_edges.shape) # return an array filled with zeros with the dimension of the bin_edges above
    cum_values[1:] = np.cumsum(hist*np.diff(bin_edges)) # cumulative sum of histogram x discrete difference

    inv_cdf = interpolate.interp1d(cum_values, bin_edges) # interpolate over the cumulative sum
    r = np.random.rand(n_samples) # generate a bunch of random numbers

    return inv_cdf(r) # return the random numbers shaped from the interpolate result above
```

Generation of a synthetic dataset

For clarity, each column will be synthesized using one or more standard Scipy distributions and ITS followed by a graphical distribution comparison to the original (experimental) data. In the end, these synthetic columns will be assembled as a Pandas dataframe and corresponding box plots and scatter matrices generated.

RMSD

We will fit the RMSD variable first using the `Fitter` library. Note that the output from the `Fitter.fit()` function is hidden for a better readability of this notebook.

In [17]:
```
RMSD_values = df['RMSD'] # get all RMSD values

f_RMSD = Fitter(RMSD_values)
f_RMSD.fit()
```

```
E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_distn_infrastructure.py:2305: RuntimeWarning:

invalid value encountered in double_scalars


Fitted alpha distribution with error=0.11631537349566168, AIC=787.6541508807237, BIC=811.7662387890002)
Fitted anglit distribution with error=0.19276851456375033, AIC=643.8353809189516, BIC=659.9101061911359)
Fitted arcsine distribution with error=0.24309575085678953, AIC=644.9283391864687, BIC=661.0030644586531)
Fitted argus distribution with error=0.20891487068510717, AIC=632.8929130702172, BIC=657.0050009784936)
Fitted beta distribution with error=0.13526581212944883, AIC=630.8470278456356, BIC=662.9964783900042)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_continuous_distns.py:589: RuntimeWarning:

divide by zero encountered in true_divide

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_continuous_distns.py:593: RuntimeWarning:

divide by zero encountered in true_divide

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_distn_infrastructure.py:1037: RuntimeWarning:

invalid value encountered in subtract


Fitted betaprime distribution with error=0.10488235852903133, AIC=679.0511385637383, BIC=711.2005891081069)
Fitted bradford distribution with error=0.13715540058853243, AIC=617.9689058856807, BIC=642.0809937939571)
Fitted burr distribution with error=0.13883726987919984, AIC=669.451731895358, BIC=701.6011824397266)
Fitted burr12 distribution with error=0.11060255689983448, AIC=674.2171494329822, BIC=706.3665999773508)
Fitted cauchy distribution with error=0.15504248117711472, AIC=780.9505544870893, BIC=797.0252797592736)
Fitted chi distribution with error=0.12922977015357104, AIC=664.5358968695555, BIC=688.6479847778319)
Fitted chi2 distribution with error=0.7348728447419687, AIC=950.8555753912547, BIC=974.9676632995311)
Fitted cosine distribution with error=0.19527444399582342, AIC=654.8639856923106, BIC=670.9387109644949)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_continuous_distns.py:6428: RuntimeWarning:

overflow encountered in power

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_continuous_distns.py:6428: RuntimeWarning:

invalid value encountered in multiply


Fitted crystalball distribution with error=0.19751263949979758, AIC=672.4560178492272, BIC=704.6054683935959)
Fitted dgamma distribution with error=0.1533304119009997, AIC=695.6076715530654, BIC=719.7197594613419)
Fitted dweibull distribution with error=0.13524938685194957, AIC=689.3572860112448, BIC=713.4693739195213)
Fitted erlang distribution with error=0.10476519617346725, AIC=676.9565215168249, BIC=701.0686094251014)
Fitted expon distribution with error=0.11843827707173928, AIC=684.5054757657238, BIC=700.5802010379081)
```

```
E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 12.600000000000001.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 12.400000000000002.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 11.666666666666668.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 12.366666666666667.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 11.844444444444441.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 12.140740740740739.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 12.035185185185185.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 12.42345679012346.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 12.278703703703705.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 12.395061728395063.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 12.575000000000003.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 12.229115226337449.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 12.266306584362141.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 12.074245541838135.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 11.92803497942387.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 12.04656492912666.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:
```

```
Fitted exponnorm distribution with error=0.07122365108652987, AIC=689.2286832028907, BIC=713.3407711111672)
Fitted exponpow distribution with error=0.12947347373429055, AIC=656.5327693360075, BIC=680.6448572442839)
Fitted exponweib distribution with error=0.12886462606685542, AIC=665.1187242840754, BIC=697.2681748284444)
Fitted f distribution with error=0.24476060874774452, AIC=762.8578701751305, BIC=795.0073207194991)
Fitted fatiguelife distribution with error=0.07643887009228663, AIC=694.3230309776778, BIC=718.4351188859544)
Fitted fisk distribution with error=0.09085124940788687, AIC=709.4556969060047, BIC=733.5677848142811)
Fitted foldcauchy distribution with error=0.10943837452673223, AIC=724.0689909834591, BIC=748.1810788917355)
Fitted foldnorm distribution with error=0.1283367823691229, AIC=657.3241661557171, BIC=681.4362540639936)
Fitted frechet_l distribution with error=1.3671080245441556, AIC=884.9102279562394, BIC=909.0223158645158)
Fitted frechet_r distribution with error=0.29422151278557873, AIC=739.7548422894129, BIC=763.8669301976893)
Fitted gamma distribution with error=0.10476611120994918, AIC=676.9557579278209, BIC=701.0678458360974)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_continuous_distns.py:3080: RuntimeWarning:

divide by zero encountered in power


Fitted gausshyper distribution with error=0.1074942851012003, AIC=639.8777118959185, BIC=688.1018877124715)
Fitted genexpon distribution with error=0.09215180664121385, AIC=687.3486961113448, BIC=727.5355092918055)
Fitted genextreme distribution with error=0.08340683326568019, AIC=717.5351699218693, BIC=741.6472578301457)
Fitted gengamma distribution with error=0.13539405741454874, AIC=663.8688932154845, BIC=696.0183437598531)
Fitted genhalflogistic distribution with error=0.14096146671270463, AIC=650.2852355632026, BIC=674.397323471479)
Fitted genlogistic distribution with error=0.16431992828800812, AIC=680.7415034176859, BIC=704.8535913259623)
Fitted gennorm distribution with error=0.16380449140776707, AIC=620.6195627683715, BIC=644.7316506766479)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_continuous_distns.py:2159: RuntimeWarning:

divide by zero encountered in true_divide
```

```
Fitted genpareto distribution with error=0.12842941256937032, AIC=636.1648989885651, BIC=660.2769868968417)
Fitted gilbrat distribution with error=0.07489316014120513, AIC=702.2304558818292, BIC=718.3051811540136)
Fitted gompertz distribution with error=0.12377347528640892, AIC=657.9417572123297, BIC=682.0538451206062)
Fitted gumbel_l distribution with error=0.2178808630977001, AIC=671.9953864065061, BIC=688.0701116786904)
Fitted gumbel_r distribution with error=0.16447236028544923, AIC=678.5909748390841, BIC=694.6657001112684)
Fitted halfcauchy distribution with error=0.11127295650547575, AIC=723.8211505723982, BIC=739.8958758445825)
Fitted halfgennorm distribution with error=0.20614663419840476, AIC=722.9690106610864, BIC=747.0810985693629)
Fitted halflogistic distribution with error=0.12085631174468997, AIC=668.8761668419472, BIC=684.9508921141315)
Fitted halfnorm distribution with error=0.12640352862863327, AIC=657.9503486367262, BIC=674.0250739089105)
Fitted hypsecant distribution with error=0.1975552084031603, AIC=693.8915531370253, BIC=709.9662784092096)
Fitted invgamma distribution with error=0.08651706329126407, AIC=711.96727639554, BIC=736.0793643038164)
Fitted invgauss distribution with error=0.0788865886766928, AIC=699.8991230806445, BIC=724.0112109889021)
Fitted invweibull distribution with error=0.08340856516115405, AIC=717.5345141816433, BIC=741.6466020899197)
Fitted johnsonsb distribution with error=0.09806556884453975, AIC=638.8007071943298, BIC=670.9501577386984)
Fitted johnsonsu distribution with error=0.08036517993727903, AIC=705.6578344420298, BIC=737.8072849863984)
Fitted kappa3 distribution with error=0.15702356585450464, AIC=630.376443463549, BIC=654.4885313718255)
Fitted kappa4 distribution with error=0.1325985199341014, AIC=628.5774750166912, BIC=660.7269255610598)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\integrate\quadpack.py:385: IntegrationWarning:

The maximum number of subdivisions (50) has been achieved.
  If increasing the limit yields no improvement it is advised to analyze
  the integrand in order to determine the difficulties.  If the position of a
  local difficulty can be determined (singularity, discontinuity) one will
  probably gain from splitting up the interval and calling the integrator
  on the subranges.  Perhaps a special-purpose integrator should be used.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_continuous_distns.py:44: RuntimeWarning:

floating point number truncated to an integer


SKIPPED ksone distribution(taking more than 30 seconds)
Fitted kstwobign distribution with error=0.16084051513998554, AIC=670.4056968220672, BIC=686.4804220942515)
Fitted laplace distribution with error=0.18149050053051569, AIC=728.3828229440606, BIC=744.4575482162545)
Fitted levy distribution with error=0.10308224311224408, AIC=769.0559489182906, BIC=785.1306741904749)
Fitted levy_l distribution with error=0.2828879928300592, AIC=771.9203034011894, BIC=787.9950286733737)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_distn_infrastructure.py:1615: RuntimeWarning:

divide by zero encountered in log


SKIPPED levy_stable distribution(taking more than 30 seconds)
Fitted loggamma distribution with error=0.19721477008822844, AIC=670.0638024978842, BIC=694.1758903961608)
Fitted logistic distribution with error=0.19850477430701408, AIC=682.3604849807748, BIC=698.4352102529591)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_continuous_distns.py:3785: RuntimeWarning:

divide by zero encountered in true_divide
```

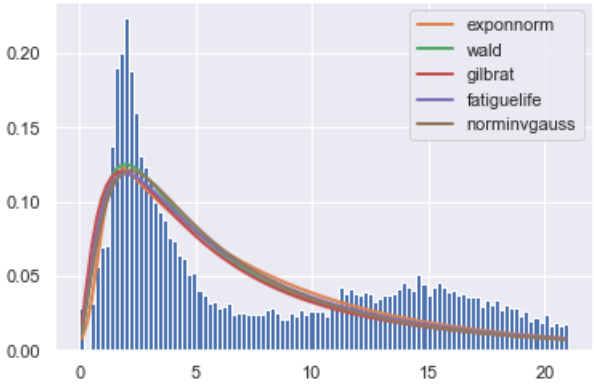
```
Fitted loglaplace distribution with error=0.42169694280895387, AIC=839.722204120245, BIC=863.83429202
85215)
Fitted lognorm distribution with error=0.08426409356382847, AIC=700.717172843455, BIC=724.82926075173
15)
Fitted lomax distribution with error=0.12104643825640862, AIC=695.1634614479029, BIC=719.275549356179
5)
Fitted maxwell distribution with error=0.1837377031355419, AIC=664.5754741282772, BIC=680.65019940046
15)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\integrate\q
uadpack.py:385: IntegrationWarning:

Extremely bad integrand behavior occurs at some points of the
  integration interval.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\integrate\q
uadpack.py:385: IntegrationWarning:

The integral is probably divergent, or slowly convergent.


Fitted mielke distribution with error=0.07924327190770933, AIC=722.5434347166578, BIC=754.69288526102
64)
Fitted moyal distribution with error=0.1444351955215317, AIC=684.2258462720207, BIC=700.300571544205)
Fitted nakagami distribution with error=0.1285384145482213, AIC=669.7254487472038, BIC=693.8375366554
803)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_dist
n_infrastructure.py:1615: RuntimeWarning:

invalid value encountered in log


SKIPPED ncf distribution(taking more than 30 seconds)
SKIPPED nct distribution(taking more than 30 seconds)
SKIPPED ncx2 distribution(taking more than 30 seconds)
Fitted norm distribution with error=0.19751263810815886, AIC=668.4560182539798, BIC=684.5307435261641
)
Fitted norminvgauss distribution with error=0.07855845296741956, AIC=702.1644832958824, BIC=734.31393
3840251)
Fitted pareto distribution with error=0.11884779262588767, AIC=687.0826854618331, BIC=711.19477337010
95)
Fitted pearson3 distribution with error=0.10476603188404154, AIC=676.9558252078366, BIC=701.067913116
113)
Fitted powerlaw distribution with error=0.16862929831891738, AIC=635.0187376179598, BIC=659.130825526
2363)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5166: RuntimeWarning:

divide by zero encountered in power

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5166: RuntimeWarning:

overflow encountered in power


Fitted powerlognorm distribution with error=0.09098882422164196, AIC=698.6353639584796, BIC=730.78481
45028482)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5203: RuntimeWarning:

divide by zero encountered in power

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5203: RuntimeWarning:

invalid value encountered in multiply

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5250: RuntimeWarning:

divide by zero encountered in power
```

```
Fitted powernorm distribution with error=nan, AIC=662.8402427262754, BIC=686.9523306345518)
Fitted rayleigh distribution with error=0.1772366951139294, AIC=660.9433356342151, BIC=677.0180609063
995)
Fitted rdist distribution with error=0.16816154605384767, AIC=615.5872975933057, BIC=639.699385501582
1)
Fitted recipinvgauss distribution with error=0.08315797843359732, AIC=688.8040647364312, BIC=712.9161
526447076)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5358: RuntimeWarning:

invalid value encountered in log


Fitted reciprocal distribution with error=0.39625631916423487, AIC=inf, BIC=inf)
Fitted rice distribution with error=0.1772371013030185, AIC=662.943674757014, BIC=687.0557626652906)
SKIPPED rv_continuous distribution(taking more than 30 seconds)
SKIPPED rv_histogram distribution(taking more than 30 seconds)
Fitted semicircular distribution with error=0.18774125934358676, AIC=626.1890695837377, BIC=642.26379
4855922)
Fitted skewnorm distribution with error=0.11576624685361314, AIC=661.9884854397284, BIC=686.100573348
0049)
Fitted t distribution with error=0.19751985704084707, AIC=670.4536792204831, BIC=694.5657671287595)
Fitted trapz distribution with error=0.2940204561467141, AIC=681.7819349140374, BIC=713.931385458406)
Fitted triang distribution with error=0.10748607459570461, AIC=647.1993933585185, BIC=671.31148126679
5)
Fitted truncexpon distribution with error=0.1336305411250901, AIC=619.4155106413361, BIC=643.52759854
96125)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5823: RuntimeWarning:

divide by zero encountered in log

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5823: RuntimeWarning:

invalid value encountered in log


Fitted truncnorm distribution with error=0.19751264044557398, AIC=672.4560166371116, BIC=704.60546718
14802)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5883: RuntimeWarning:

divide by zero encountered in power


SKIPPED tukeylambda distribution(taking more than 30 seconds)
Fitted uniform distribution with error=0.16923987526987816, AIC=612.88543901855, BIC=628.960164290734
3)
Fitted vonmises distribution with error=4.003227524934667e+55, AIC=-12249.465587300925, BIC=-12225.35
3499392648)
Fitted vonmises_line distribution with error=0.16923988125482564, AIC=614.9177445892464, BIC=639.0298
324975229)
Fitted wald distribution with error=0.07186774326386595, AIC=700.9441738258258, BIC=717.0188990980101
)
Fitted weibull_max distribution with error=1.3671080245441556, AIC=884.9102279562394, BIC=909.0223158
645158)
Fitted weibull_min distribution with error=0.29422151278557873, AIC=739.7548422894129, BIC=763.866930
1976893)
Fitted wrapcauchy distribution with error=nan, AIC=nan, BIC=nan)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\optimize\op
timize.py:563: RuntimeWarning:

invalid value encountered in subtract

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\numpy\core\fromnu
meric.py:83: RuntimeWarning:

invalid value encountered in reduce
```

In [18]:
```
f_RMSD.summary()
```

Out[18]:

	sumsquare_error	AIC	BIC
exponnorm	0.071224	689.228683	713.340771
wald	0.071868	700.944174	717.018899
gilbrat	0.074893	702.230456	718.305181
fatiguelife	0.076439	694.323031	718.435119
norminvgauss	0.078558	702.164483	734.313934



Based on the plot above, it is clear that none of the distributions in `scipy.stats` module fits the RMSD data with the default bin size. We can try increasing the bins size.

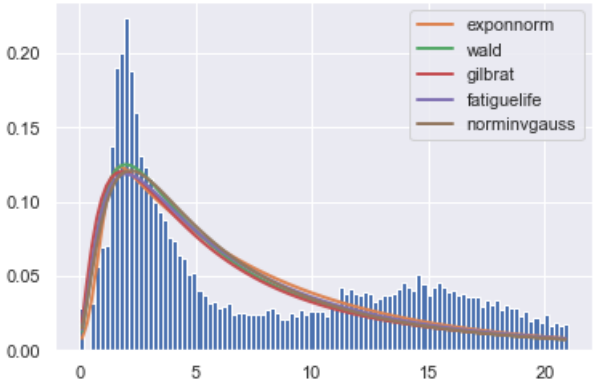
In [19]:
```
f_RMSD = Fitter(RMSD_values, bins = 100) # can pass xmin = 7.5 if needed
f_RMSD.fit()
```

Fitted alpha distribution with error=0.11631537349566168, AIC=787.6541508807237, BIC=811.766238789000
2)
Fitted anglit distribution with error=0.19276851456375033, AIC=643.8353809189516, BIC=659.91010619113
59)
Fitted arcsine distribution with error=0.24309575085678953, AIC=644.9283391864687, BIC=661.0030644586
531)
Fitted argus distribution with error=0.20891487068510717, AIC=632.8929130702172, BIC=657.005000978493
6)
Fitted beta distribution with error=0.13526581212944883, AIC=630.8470278456356, BIC=662.9964783900042
)
Fitted betaprime distribution with error=0.10488235852903133, AIC=679.0511385637383, BIC=711.20058910
81069)
Fitted bradford distribution with error=0.13715540058853243, AIC=617.9689058856807, BIC=642.080993793
9571)
Fitted burr distribution with error=0.13883726987919984, AIC=669.451731895358, BIC=701.6011824397266)
Fitted burr12 distribution with error=0.11060255689983448, AIC=674.2171494329822, BIC=706.36659997735
08)
Fitted cauchy distribution with error=0.15504248117711472, AIC=780.9505544870893, BIC=797.02527975927
36)
Fitted chi distribution with error=0.12922977015357104, AIC=664.5358968695555, BIC=688.6479847778319)
Fitted chi2 distribution with error=0.7348728447419687, AIC=950.8555753912547, BIC=974.9676632995311)
Fitted cosine distribution with error=0.19527444399582342, AIC=654.8639856923106, BIC=670.93871096449
49)
Fitted crystalball distribution with error=0.19751263949979758, AIC=672.4560178492272, BIC=704.605468
3935959)
Fitted dgamma distribution with error=0.1533304119009997, AIC=695.6076715530654, BIC=719.719759461341
9)
Fitted dweibull distribution with error=0.13524938685194957, AIC=689.3572860112448, BIC=713.469373919
5213)
Fitted erlang distribution with error=0.10476519617346725, AIC=676.9565215168249, BIC=701.06860942510
14)
Fitted expon distribution with error=0.11843827707173928, AIC=684.5054757657238, BIC=700.580201037908
1)
Fitted exponnorm distribution with error=0.07122365108652987, AIC=689.2286832028907, BIC=713.34077111
11672)
Fitted exponpow distribution with error=0.12947347373429055, AIC=656.5327693360075, BIC=680.644857244
2839)
Fitted exponweib distribution with error=0.12886462606685542, AIC=665.1187242840754, BIC=697.26817482
8444)
Fitted f distribution with error=0.24476060874774452, AIC=762.8578701751305, BIC=795.0073207194991)
Fitted fatiguelife distribution with error=0.07643887009228663, AIC=694.3230309776778, BIC=718.435118
8859544)
Fitted fisk distribution with error=0.09085124940788687, AIC=709.4556969060047, BIC=733.5677848142811
)
Fitted foldcauchy distribution with error=0.10943837452673223, AIC=724.0689909834591, BIC=748.1810788
917355)
Fitted foldnorm distribution with error=0.1283367823691229, AIC=657.3241661557171, BIC=681.4362540639
936)
Fitted frechet_l distribution with error=1.3671080245441556, AIC=884.9102279562394, BIC=909.022315864
5158)
Fitted frechet_r distribution with error=0.29422151278557873, AIC=739.7548422894129, BIC=763.86693019
76893)
Fitted gamma distribution with error=0.10476611120994918, AIC=676.9557579278209, BIC=701.067845836097
4)
Fitted gausshyper distribution with error=0.1074942851012003, AIC=639.8777118959185, BIC=688.10188771
24715)
Fitted genexpon distribution with error=0.09215180664121385, AIC=687.3486961113448, BIC=727.535509291
8055)
Fitted genextreme distribution with error=0.08340683326568019, AIC=717.5351699218693, BIC=741.6472578
301457)
Fitted gengamma distribution with error=0.13539405741454874, AIC=663.8688932154845, BIC=696.018343759
8531)
Fitted genhalflogistic distribution with error=0.14096146671270463, AIC=650.2852355632026, BIC=674.39
7323471479)
Fitted genlogistic distribution with error=0.16431992828800812, AIC=680.7415034176859, BIC=704.853591
3259623)
Fitted gennorm distribution with error=0.16380449140776707, AIC=620.6195627683715, BIC=644.7316506766
479)
Fitted genpareto distribution with error=0.12842941256937032, AIC=636.1648989885651, BIC=660.27698689
68417)
Fitted gilbrat distribution with error=0.07489316014120513, AIC=702.2304558818292, BIC=718.3051811540
136)
Fitted gompertz distribution with error=0.12377347528640892, AIC=657.9417572123297, BIC=682.053845120
6062)
Fitted gumbel_l distribution with error=0.2178808630977001, AIC=671.9953864065061, BIC=688.0701116786
904)
Fitted gumbel_r distribution with error=0.16447236028544923, AIC=678.5909748390841, BIC=694.665700111
2684)
Fitted halfcauchy distribution with error=0.11127295650547575, AIC=723.8211505723982, BIC=739.8958758
445825)
Fitted halfgennorm distribution with error=0.20614663419840476, AIC=722.9690106610864, BIC=747.081098
5693629)

In [20]:
```
f_RMSD.summary()
```

Out[20]:

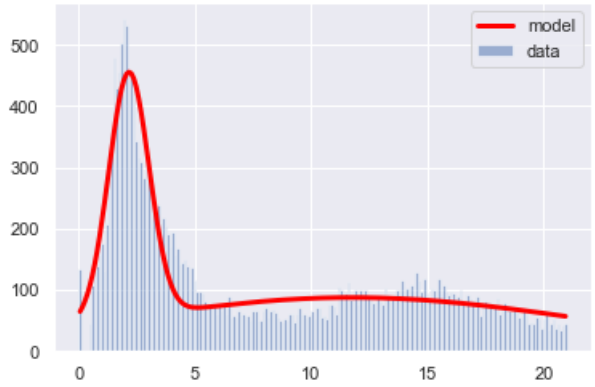
	sumsquare_error	AIC	BIC
exponnorm	0.071224	689.228683	713.340771
wald	0.071868	700.944174	717.018899
gilbrat	0.074893	702.230456	718.305181
fatiguelife	0.076439	694.323031	718.435119
norminvgauss	0.078558	702.164483	734.313934



Unfortunately, automatic fitting has failed on the RMSD data. More likely, this is a bimodal normal distribution with a mean of about 3 and 15. In this instance, we will have to manually define two normal distributions following by curve fitting.

In [21]:
```
# https://stackoverflow.com/questions/37559470/what-do-all-the-distributions-available-in-scipy-stats-look-like
# https://stackoverflow.com/questions/35990467/fit-two-gaussians-to-a-histogram-from-one-set-of-data-python

# https://matplotlib.org/api/_as_gen/matplotlib.pyplot.hist.html
# hist() returns a tuple, with the last element being a silent list of patches,
# so we assign this to special variable '_'
# https://hackernoon.com/understanding-the-underscore-of-python-309d1a029edc
# https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.curve_fit.html
# https://docs.scipy.org/doc/numpy/reference/generated/numpy.diag.html

y, x, _ = hist(df['RMSD'], 200, alpha = 0.5, label = 'data')

x = (x[1:] + x[:-1]) / 2 # for len(x)==len(y)

def gauss(x, mu, sigma, A):
    return A*exp(-(x - mu) **2 / 2 / sigma**2)

def bimodal(x, mu1, sigma1, A1, mu2, sigma2, A2):
    return gauss(x, mu1, sigma1, A1) + gauss(x, mu2, sigma2, A2)

# initial estimates for non-linear least square
# we define the parameters from estimating by eye:
# Normal Distribution 1 - mean = 3, standard deviation = 0.5 and scale of 800
# Normal Distribution 2 - mean = 15, standard deviation = 9 and scale of 150
expected = (3, 0.5, 800, 15, 9, 150)

# curve_fit returns popt (array of optimal params) and pcov (array of est covariance of popt)
params, cov = curve_fit(bimodal, x, y, expected)
sigma = sqrt(diag(cov))

plot(x, bimodal(x, *params), color='red', lw=3, label='model')
legend()

print(params, '\n', sigma)
```

```
[  2.15343859   0.86642361 401.06184843  11.75858425  -9.89387758
  87.6740905 ]
 [ 0.0282778   0.03797205 13.47084028  0.82197174  1.7028006   4.22117304]
```



From the output above, we obtain two normal distributions that is relatively a good approximation of the RMSD data.

HTML code is used for rendering the table below because of some bugs in Markdown/Jupyter for center alignment.

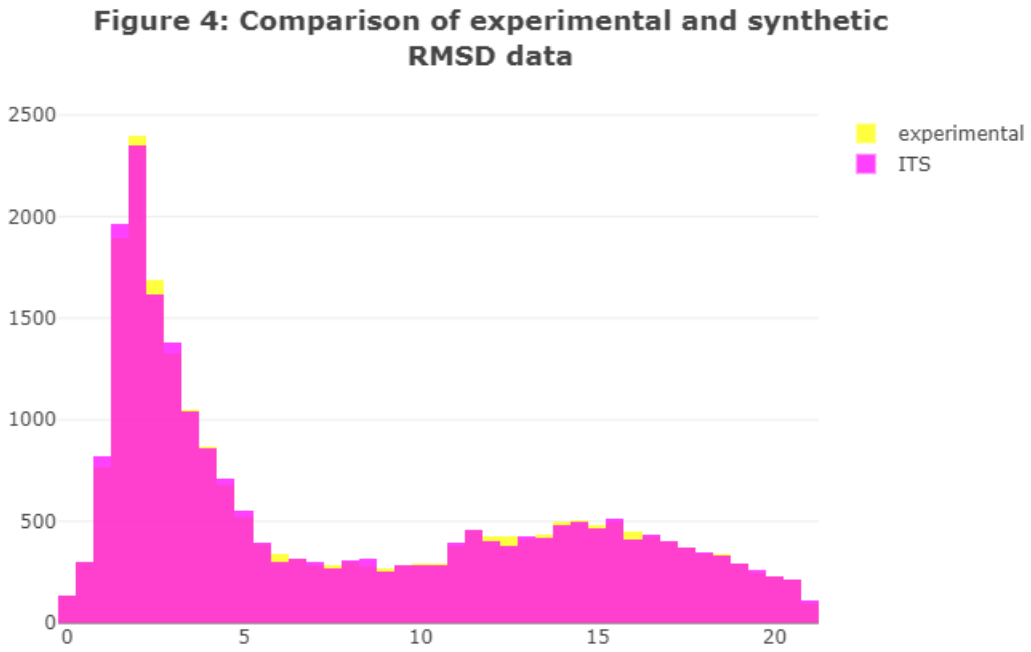
Distribution	μ	σ	**A (scale)**
Normal1	2.15	0.87	401.06
Normal2	11.76	9.89	87.67

However, while we can describe the data as an additive function of two normal distributions, this whole process of providing initial estimates and performing function addition is quite cumbersome and not really amenable for automation. As such, ITS is a better option to generate random values for RMSD. To note, Monte Carlo rejection sampling could also be used (https://stackoverflow.com/questions/41470070/python-random-sampling-from-self-defined-probability-function).

In [22]:
```
RMSD_ITS = inverse_transform_sampling(df['RMSD'])

# reduce the variable memory footprint
# https://docs.scipy.org/doc/numpy-1.9.0/reference/generated/numpy.ndarray.astype.html
RMSD_ITS = RMSD_ITS.astype(dtype = 'float32', casting = 'same_kind')
```

In [108]:
```
# https://plot.ly/python/histograms/

trace_experimental = go.Histogram(x = df['RMSD'], opacity= 0.75, name = 'experimental', marker = dict(
color = 'yellow'))
trace_ITS = go.Histogram(x = RMSD_ITS, opacity= 0.75, name = 'ITS', marker = dict(color = 'magenta'))

data = [trace_experimental, trace_ITS]

layout = go.Layout(barmode= 'overlay', title = '<b>Figure 4: Comparison of experimental and synthetic<
br>RMSD data</b>')

fig = go.Figure(data=data, layout=layout)
# iplot(fig)

img_bytes = pio.to_image(fig, format='png')
Image(img_bytes)
```

Out[108]:



By visual comparison, we can be confident that ITS provides a good approximation of the experimental dataset and that the random values hence generated follows the original distribution.

Total Surface Area

Let us now look at the Total Surface Area data.

In [24]:
```
data = df['Total surface area']

f_TotalSurfaceArea = Fitter(data, bins=200)
f_TotalSurfaceArea.fit()
```

```
Fitted alpha distribution with error=1.0478016182660564e-08, AIC=4836.504044685137, BIC=4860.616132593414)
Fitted anglit distribution with error=2.691256504249789e-07, AIC=4304.669976582023, BIC=4320.744701854207)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_continuous_distns.py:312: RuntimeWarning:

divide by zero encountered in true_divide


Fitted arcsine distribution with error=4.2362253955991655e-07, AIC=4252.917175124052, BIC=4268.991900396236)
Fitted argus distribution with error=4.018623983461105e-07, AIC=4230.436159188017, BIC=4254.548247096293)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_continuous_distns.py:437: RuntimeWarning:

invalid value encountered in sqrt


Fitted beta distribution with error=1.19253663649438e-08, AIC=5051.151715004125, BIC=5083.301165548494)
Fitted betaprime distribution with error=8.658083517404978e-09, AIC=4822.089627119727, BIC=4854.239077664096)
Fitted bradford distribution with error=2.2256915312590553e-07, AIC=4273.480238085705, BIC=4297.592325993982)
Fitted burr distribution with error=1.8963417825992672e-08, AIC=4701.385610530255, BIC=4733.535061074624)
Fitted burr12 distribution with error=1.333440434533281e-08, AIC=4785.561818312838, BIC=4817.711268857207)
Fitted cauchy distribution with error=4.298806659538157e-08, AIC=4756.638025225648, BIC=4772.712750497832)
Fitted chi distribution with error=6.055610091630982e-08, AIC=6181.624798840752, BIC=6205.736886749029)
Fitted chi2 distribution with error=1.6302001995893303e-08, AIC=5243.453402858791, BIC=5267.565490767068)
Fitted cosine distribution with error=2.2441164497822283e-07, AIC=4452.9672319367255, BIC=4469.0419572089095)
Fitted crystalball distribution with error=4.7136028022213187e-07, AIC=inf, BIC=inf)
Fitted dgamma distribution with error=4.437036971198926e-08, AIC=5227.7631465581735, BIC=5251.875234466645)
Fitted dweibull distribution with error=4.789694082233307e-08, AIC=5345.893110169589, BIC=5370.005198077865)
```

```
E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.1500000000000004.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.1000000000000005.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.166666666666667.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.25.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.083333333333334.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.0500000000000007.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.0999999999999996.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.2666666666666666.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.4000000000000004.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.116666666666667.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.233333333333335.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.3000000000000025.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.450000000000002.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.650000000000002.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.26666666666667.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.2000000000000046.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:
```

```
Fitted erlang distribution with error=3.987697022531147e-08, AIC=5720.289974311106, BIC=5744.402062219382)
Fitted expon distribution with error=2.0993258875442315e-07, AIC=4534.776381387949, BIC=4550.851106660133)
Fitted exponnorm distribution with error=8.945773381055425e-09, AIC=4833.083546131788, BIC=4857.195634040065)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_continuous_distns.py:1428: RuntimeWarning:

overflow encountered in exp
```

Fitted exponpow distribution with error=4.7136028022213187e-07, AIC=inf, BIC=inf)
Fitted exponweib distribution with error=1.3607192760211508e-06, AIC=4745.922271114468, BIC=4778.071721658836)
Fitted f distribution with error=8.734732202856323e-09, AIC=4819.227720154706, BIC=4851.377170699075)
Fitted fatiguelife distribution with error=4.185778932789757e-07, AIC=4820.214086203025, BIC=4844.326174111301)
Fitted fisk distribution with error=1.0193670235703998e-08, AIC=4750.936663698276, BIC=4775.048751606552)
Fitted foldcauchy distribution with error=3.1893349220114126e-08, AIC=4658.062810084446, BIC=4682.174897992722)
Fitted foldnorm distribution with error=4.7136028022213187e-07, AIC=inf, BIC=inf)
Fitted frechet_l distribution with error=5.894522836066464e-07, AIC=5337.674277749529, BIC=5361.786365657806)
Fitted frechet_r distribution with error=6.739427756602809e-07, AIC=5269.09409662267, BIC=5293.206184530946)
Fitted gamma distribution with error=1.1830052026068336e-08, AIC=5066.944682503529, BIC=5091.056770411806)
Fitted gausshyper distribution with error=1.3281320443951388e-08, AIC=5148.946102088576, BIC=5197.170277905128)
Fitted genexpon distribution with error=2.0120954846252852e-07, AIC=4512.654150563086, BIC=4552.840963743546)
Fitted genextreme distribution with error=6.691550481326913e-07, AIC=5280.422809987989, BIC=5304.534897896266)
Fitted gengamma distribution with error=2.1146085399478523e-07, AIC=4542.206045840832, BIC=4574.355496385201)
Fitted genhalflogistic distribution with error=1.0109438032966297e-07, AIC=5240.680694181436, BIC=5264.792782089712)
Fitted genlogistic distribution with error=1.4660463381917285e-08, AIC=5044.451760814245, BIC=5068.5638487225215)
Fitted gennorm distribution with error=4.666999732814913e-08, AIC=5479.381662980345, BIC=5503.493750888621)
Fitted genpareto distribution with error=1.6474337636532276e-07, AIC=4804.067223149799, BIC=4828.179311058076)
Fitted gilbrat distribution with error=9.985704396462411e-08, AIC=4467.788811829849, BIC=4483.863537102033)
Fitted gompertz distribution with error=4.7136028022213187e-07, AIC=inf, BIC=inf)
Fitted gumbel_l distribution with error=1.2854187803700586e-07, AIC=11417.282582332107, BIC=11433.357307604292)
Fitted gumbel_r distribution with error=1.4679537428064888e-08, AIC=5042.061939466233, BIC=5058.136664738417)
Fitted halfcauchy distribution with error=1.8614555786070317e-07, AIC=4477.92358085556, BIC=4493.998306127744)
Fitted halfgennorm distribution with error=7.217529180751973e-07, AIC=4558.749397503452, BIC=4582.861485411729)
Fitted halflogistic distribution with error=1.3379835464195368e-07, AIC=4650.0670816587935, BIC=4666.1418069309775)
Fitted halfnorm distribution with error=1.1917362903107074e-07, AIC=4909.8638662031135, BIC=4925.938591475297 5)
Fitted hypsecant distribution with error=4.441776869839163e-08, AIC=5222.220613040461, BIC=5238.295338312645)
Fitted invgamma distribution with error=9.122187617092113e-09, AIC=4853.796518492744, BIC=4877.908606401021)
Fitted invgauss distribution with error=1.614055974669724e-07, AIC=4495.311830676759, BIC=4519.423918585036)
Fitted invweibull distribution with error=9.576457967578482e-09, AIC=4855.520589588704, BIC=4879.632677749698)
Fitted johnsonsb distribution with error=9.378065974032111e-09, AIC=5122.12073803149, BIC=5154.270188575859)
Fitted johnsonsu distribution with error=8.761409773351198e-09, AIC=4883.145990877799, BIC=4915.295441422168)
Fitted kappa3 distribution with error=9.16045497944607e-08, AIC=4908.479806692071, BIC=4932.591894600348)
Fitted kappa4 distribution with error=2.6116884635483285e-07, AIC=4530.519818719042, BIC=4562.669269263411)
SKIPPED ksone distribution(taking more than 30 seconds)
Fitted kstwobign distribution with error=1.4960762826650748e-08, AIC=5340.838393490987, BIC=5356.913118763171)
Fitted laplace distribution with error=4.388822408298884e-08, AIC=5118.402427055535, BIC=5134.477152327719)
Fitted levy distribution with error=1.7927175303566675e-07, AIC=4544.964301616801, BIC=4561.039026888985)
Fitted levy_l distribution with error=4.002835560583469e-07, AIC=4881.791539607611, BIC=4897.866264879795)
SKIPPED levy_stable distribution(taking more than 30 seconds)
Fitted loggamma distribution with error=6.30076991316392e-08, AIC=6264.182476785519, BIC=6288.294564693795)
Fitted logistic distribution with error=4.9446511350778173e-08, AIC=5352.657897807494, BIC=5368.732623079678)
Fitted loglaplace distribution with error=3.488187801864823e-08, AIC=4758.073970252466, BIC=4782.186058581607425)
Fitted lognorm distribution with error=6.868437074791697e-07, AIC=5240.713452690998, BIC=5264.825540599274)

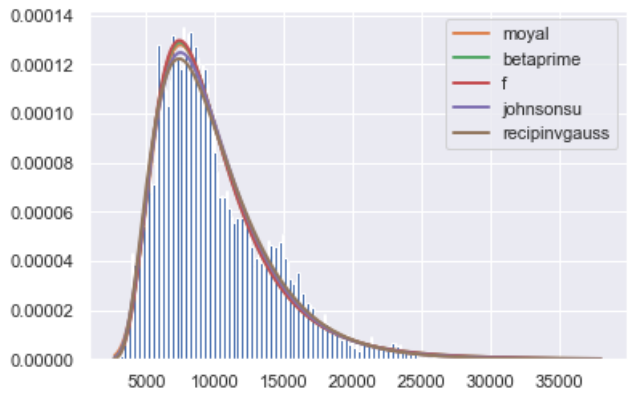
```
E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:4064: RuntimeWarning:

overflow encountered in power

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:4064: RuntimeWarning:

overflow encountered in multiply

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:4064: RuntimeWarning:

invalid value encountered in true_divide


Fitted mielke distribution with error=4.776466468662414e-08, AIC=4482.127646599927, BIC=4514.27709714
4296)
Fitted moyal distribution with error=8.402316212981134e-09, AIC=4888.113879261367, BIC=4904.188604533
551)
Fitted nakagami distribution with error=2.7248298095713998e-08, AIC=5548.319332859219, BIC=5572.43142
0767495)
SKIPPED ncf distribution(taking more than 30 seconds)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:4768: RuntimeWarning:

invalid value encountered in multiply

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:4764: RuntimeWarning:

overflow encountered in multiply
```

```
SKIPPED nct distribution(taking more than 30 seconds)
Fitted ncx2 distribution with error=4.7136025488315914e-07, AIC=inf, BIC=inf)
Fitted norm distribution with error=6.211602584679603e-08, AIC=6278.2790207863, BIC=6294.353746058484
)
Fitted norminvgauss distribution with error=4.6514707321660875e-07, AIC=6209.4855867614715, BIC=6241.
63503730584)
Fitted pareto distribution with error=5.403497841738131e-07, AIC=4592.775550236206, BIC=4616.88763814
44825)
Fitted pearson3 distribution with error=1.1830051294687458e-08, AIC=5066.944665407929, BIC=5091.05675
3316206)
Fitted powerlaw distribution with error=3.1141373449229147e-07, AIC=4250.9477979209605, BIC=4275.0598
85829237)
Fitted powerlognorm distribution with error=1.4599547168183611e-06, AIC=4837.172284973488, BIC=4869.3
21735517857)
Fitted powernorm distribution with error=nan, AIC=5662.561830571382, BIC=5686.673918479659)
Fitted rayleigh distribution with error=2.7429776068037014e-08, AIC=5566.181890396848, BIC=5582.25661
5669032)
Fitted rdist distribution with error=6.218772281977415e-08, AIC=6279.899154172457, BIC=6304.011242080
734)
Fitted recipinvgauss distribution with error=8.881908290953462e-09, AIC=4942.794368181852, BIC=4966.9
06456090129)
Fitted reciprocal distribution with error=4.7136028022213187e-07, AIC=inf, BIC=inf)
Fitted rice distribution with error=2.7429776279020352e-08, AIC=5568.181885161797, BIC=5592.293973070
074)
SKIPPED rv_continuous distribution(taking more than 30 seconds)
SKIPPED rv_histogram distribution(taking more than 30 seconds)
Fitted semicircular distribution with error=3.034638736136199e-07, AIC=4230.850318230932, BIC=4246.92
5043503116)
Fitted skewnorm distribution with error=1.2892602026702598e-08, AIC=5304.417158387654, BIC=5328.52924
62959305)
Fitted t distribution with error=4.925943774076605e-08, AIC=5212.428808418835, BIC=5236.540896327111)
Fitted trapz distribution with error=4.723509762998312e-07, AIC=4331.4796380799635, BIC=4363.62908862
4332)
Fitted triang distribution with error=1.5033407118833356e-07, AIC=4317.343333289156, BIC=4341.4554211
97433)
Fitted truncexpon distribution with error=2.3403037074539588e-07, AIC=4280.432231427376, BIC=4304.544
319335652)
Fitted truncnorm distribution with error=4.7136028022213187e-07, AIC=inf, BIC=inf)
SKIPPED tukeylambda distribution(taking more than 30 seconds)
Fitted uniform distribution with error=3.1208506220062946e-07, AIC=4194.189913331181, BIC=4210.264638
603365)
Fitted vonmises distribution with error=2.990508306991258e+49, AIC=-21303.557172104767, BIC=-21279.44
508419649)
Fitted vonmises_line distribution with error=5.771257380245715e-08, AIC=5508.72855256155, BIC=5532.84
0640469826)
Fitted wald distribution with error=6.581843650381245e-08, AIC=4548.973183450117, BIC=4565.0479087223
01)
Fitted weibull_max distribution with error=5.894522836066464e-07, AIC=5337.674277749529, BIC=5361.786
365657806)
Fitted weibull_min distribution with error=6.739427756602809e-07, AIC=5269.09409662267, BIC=5293.2061
84530946)
Fitted wrapcauchy distribution with error=nan, AIC=nan, BIC=nan)
```

In [25]:
```
f_TotalSurfaceArea.summary()
```

Out[25]:

	sumsquare_error	AIC	BIC
moyal	8.402316e-09	4888.113879	4904.188605
betaprime	8.658084e-09	4822.089627	4854.239078
f	8.734732e-09	4819.227720	4851.377171
johnsonsu	8.761410e-09	4883.145991	4915.295441
recipinvgauss	8.881908e-09	4942.794368	4966.906456



In contrast to the RMSD data, it seems the Total surface area can be easily approximated by these 5 distributions. However, looking at the (`numpy.random` documentation)[https://www.numpy.org/devdocs/reference/routines.random.html (https://www.numpy.org/devdocs/reference/routines.random.html)], the only standard method to generate random numbers is through the `numpy.random.f` (F distribution). This is one of the main problems of fitting a curve first and then trying to determine a method to generate random numbers that follows a particular distribution - ITS is much easier.

In any case, for illustrative purposes, random numbers following the distributions below will be generated from parameters given by the `fitted_param` object:

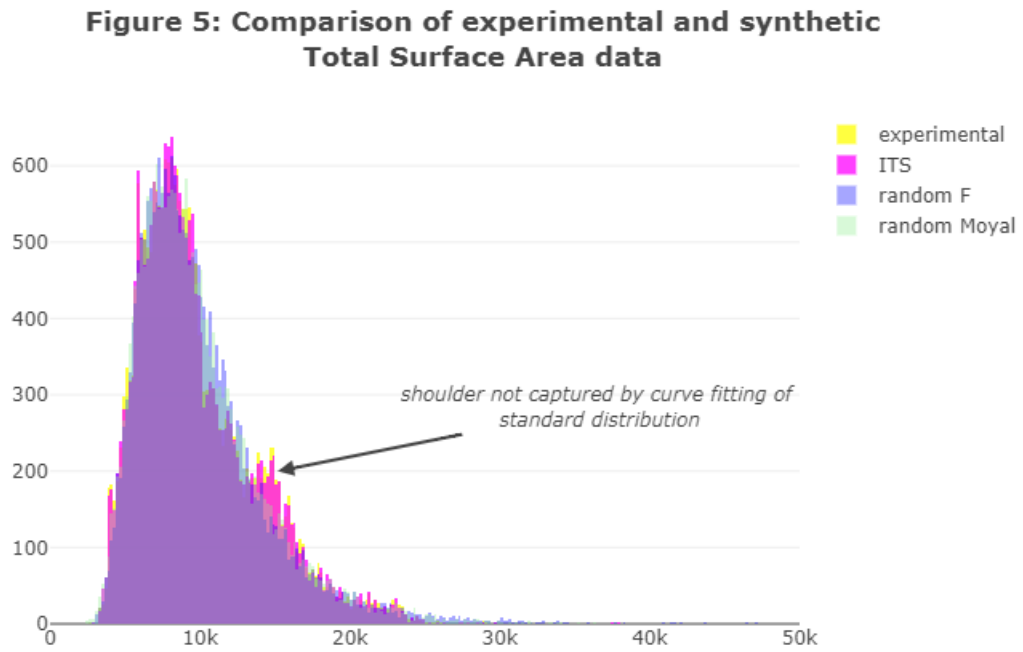
1. Moyal distribution - this is not available from `numpy.random`, so `scipy.stats.moyal.rvs()` (https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.moyal.html) will be used
2. F distribution - although the Moyal distribution has the lowest error, the F distribution seems to be similar.

These two distributions will also be compared visually to the one from ITS method.

In [95]:
```
# for numpy.random, we have to match the scipy parameters from fitted_param into numpy.random arguments
TotalArea_random_F = (np.random.f(f_TotalSurfaceArea.fitted_param['f'][0],
                      f_TotalSurfaceArea.fitted_param['f'][1], len(df)) *
                      f_TotalSurfaceArea.fitted_param['f'][3] ) - f_TotalSurfaceArea.fitted_param['f'][
2]

TotalArea_random_moyal = moyal.rvs(f_TotalSurfaceArea.fitted_param['moyal'][0],
                                   f_TotalSurfaceArea.fitted_param['moyal'][1], len(df))

TotalArea_ITS = inverse_transform_sampling(df['Total surface area'], n_samples=len(df))

TotalArea_random_F = TotalArea_random_F.astype(dtype = 'float32', casting = 'same_kind')
TotalArea_random_moyal = TotalArea_random_moyal.astype(dtype = 'float32', casting = 'same_kind')
TotalArea_ITS = TotalArea_ITS.astype(dtype = 'float32', casting = 'same_kind')
```

In [109]:
```
trace_experimental = go.Histogram(x = df['Total surface area'], opacity= 0.75, name = 'experimental',
                                  marker = dict(color = 'yellow'))

trace_ITS = go.Histogram(x = TotalArea_ITS, opacity= 0.75, name = 'ITS',
                                  marker = dict(color = 'magenta'))

trace_random_F = go.Histogram(x = TotalArea_random_F, opacity= 0.35, name = 'random F',
                                  marker = dict(color = 'blue'))

trace_random_moyal = go.Histogram(x = TotalArea_random_moyal, opacity= 0.35, name = 'random Moyal',
                                  marker = dict(color = 'lightgreen'))

data = [trace_experimental, trace_ITS, trace_random_F, trace_random_moyal]

# https://plot.ly/python/text-and-annotations/
# https://plot.ly/python/axes/

layout = go.Layout(barmode = 'overlay',
                   title = '<b>Figure 5: Comparison of experimental and synthetic<br>Total Surface Are
a data</b>',
                   xaxis=dict(range=[0, 50000]),
    annotations=[
        dict(
            x=15200,
            y=200,
            xref='x',
            yref='y',
            text='<i>shoulder not captured by curve fitting of <br>standard distribution</i>',
            showarrow=True,
            arrowhead=2,
            arrowwidth = 2,
            arrowsize = 1,
            ax=200,
            ay=-40,
            align = 'center'
        )
    ]
)

fig = go.Figure(data=data, layout=layout)
# iplot(fig)
img_bytes = pio.to_image(fig, format='png')
Image(img_bytes)
```

Out[109]:



While our random numbers closely mimic the experimental data, it does not fully capture the subtleties present, for instance, the shoulder around (x = 15000, y = 200) (annotated above).

Non-polar Exposed Area

In this section, we will generate synthetic Non-polar exposed area data.

In [28]:
```
f_NPEA = Fitter(df['Non-polar exposed area'], bins=200)
f_NPEA.fit()
```

```
Fitted alpha distribution with error=3.678925586942494e-08, AIC=4663.1062066840295, BIC=4687.218294592306)
Fitted anglit distribution with error=1.9843725656404864e-06, AIC=3970.333989305675, BIC=3986.4087145778594)
Fitted arcsine distribution with error=2.7604523721356452e-06, AIC=3906.463651779715, BIC=3922.5383770518993)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_continuous_distns.py:6532: RuntimeWarning:

divide by zero encountered in double_scalars


Fitted argus distribution with error=2.634455052356412e-06, AIC=3858.2608000287955, BIC=3882.372887937072)
Fitted beta distribution with error=5.31345675441063e-08, AIC=5058.101711592413, BIC=5090.2511621367832)
Fitted betaprime distribution with error=3.408534650989799e-08, AIC=4633.050190756943, BIC=4665.199641301312)
Fitted bradford distribution with error=1.5432867953848278e-06, AIC=3879.8391690466824, BIC=3903.951256954959)
Fitted burr distribution with error=2.0561322984934345e-07, AIC=4429.375750093599, BIC=4461.525200637968)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_continuous_distns.py:768: RuntimeWarning:

overflow encountered in power


Fitted burr12 distribution with error=2.9464960678852713e-06, AIC=4413.338506251447, BIC=4445.487956795816)
Fitted cauchy distribution with error=2.56810227340771e-07, AIC=4488.833254284032, BIC=4504.907979556216)
Fitted chi distribution with error=1.5653972357363993e-07, AIC=5944.272528483451, BIC=5968.384616391728)
Fitted chi2 distribution with error=5.608856540037681e-08, AIC=5071.180729594946, BIC=5095.292817503222)
Fitted cosine distribution with error=1.7142908180493399e-06, AIC=4125.345562482416, BIC=4141.420287754546)
Fitted crystalball distribution with error=3.121253587917722e-06, AIC=inf, BIC=inf)
Fitted dgamma distribution with error=2.810032740456121e-07, AIC=5297.0572046977895, BIC=5321.169292606066)
Fitted dweibull distribution with error=2.6822687951882724e-07, AIC=5486.157401381136, BIC=5510.269489289412)
```

```
E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.1.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.0666666666666664.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.1111111111111107.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.166666666666666.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.055555555555555.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.0333333333333323.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.0666666666666655.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.1777777777777763.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.266666666666664.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.077777777777775.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.0333333333333297.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.1555555555555515.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.199999999999994.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.2999999999999927.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.433333333333323.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.1777777777777674.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:
```

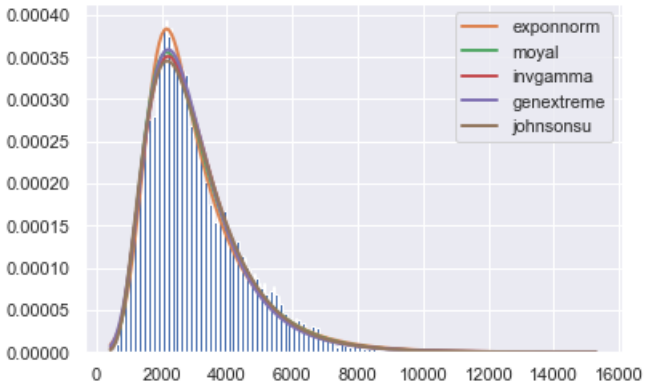
```
Fitted erlang distribution with error=2.8535054523074254e-07, AIC=6314.7486648828335, BIC=6338.860752
79111)
Fitted expon distribution with error=1.3955479385563948e-06, AIC=4291.21020194811, BIC=4307.284927220
294)
Fitted exponnorm distribution with error=2.6311072973365234e-08, AIC=4713.408131274537, BIC=4737.5202
19182813)
Fitted exponpow distribution with error=3.121253587917722e-06, AIC=inf, BIC=inf)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:1380: RuntimeWarning:

invalid value encountered in add


Fitted exponweib distribution with error=8.466028297712115e-06, AIC=4451.729644255354, BIC=4483.87909
4799723)
Fitted f distribution with error=3.3862634637220106e-08, AIC=4758.385460395295, BIC=4790.534910939664
)
Fitted fatiguelife distribution with error=3.169084769041547e-07, AIC=4328.72081066516, BIC=4352.8328
98573436)
Fitted fisk distribution with error=3.5763027667451976e-08, AIC=4542.712655659695, BIC=4566.824743567
971)
Fitted foldcauchy distribution with error=1.9014027621564046e-07, AIC=4358.25064099478, BIC=4382.3627
28903056)
Fitted foldnorm distribution with error=3.121253587917722e-06, AIC=inf, BIC=inf)
Fitted frechet_l distribution with error=3.956320364494568e-06, AIC=4952.14625897031, BIC=4976.258346
878586)
Fitted frechet_r distribution with error=4.196921792588656e-06, AIC=4930.153832742106, BIC=4954.26592
0650382)
Fitted gamma distribution with error=5.323134905432527e-08, AIC=5068.497255020249, BIC=5092.609342928
526)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:3080: RuntimeWarning:

invalid value encountered in power
```

Fitted gausshyper distribution with error=1.1902338309742346e-07, AIC=5987.618311218046, BIC=6035.842
487034599)
Fitted genexpon distribution with error=1.39618643417541e-06, AIC=4297.331601462547, BIC=4337.5184146
43007)
Fitted genextreme distribution with error=3.1804763257712944e-08, AIC=4740.224071931876, BIC=4764.336
1598401525)
Fitted gengamma distribution with error=3.2683441522402615e-06, AIC=4306.68191209313, BIC=4338.831362
637499)
Fitted genhalflogistic distribution with error=6.717089066790304e-07, AIC=5032.125681105221, BIC=5056
.237769013497)
Fitted genlogistic distribution with error=6.144107512271015e-08, AIC=5014.460324152948, BIC=5038.572
412061224)
Fitted gennorm distribution with error=2.612302824877916e-07, AIC=5545.994763933025, BIC=5570.1068518
41301)
Fitted genpareto distribution with error=1.1426098180233255e-06, AIC=4648.360753886609, BIC=4672.4728
41794885)
Fitted gilbrat distribution with error=6.741743545933931e-07, AIC=4172.866008217473, BIC=4188.9407334
89657)
Fitted gompertz distribution with error=3.121253587917722e-06, AIC=inf, BIC=inf)
Fitted gumbel_l distribution with error=8.797332849253318e-07, AIC=18560.70208043368, BIC=18576.77680
5705864)
Fitted gumbel_r distribution with error=6.158697646315506e-08, AIC=5008.3982365904485, BIC=5024.47296
18626325)
Fitted halfcauchy distribution with error=1.2336059298641582e-06, AIC=4163.951710809243, BIC=4180.026
436081427)
Fitted halfgennorm distribution with error=4.460584001185633e-06, AIC=4220.631193163519, BIC=4244.743
2810717955)
Fitted halflogistic distribution with error=8.872528211930473e-07, AIC=4471.191719773138, BIC=4487.26
6445045322)
Fitted halfnorm distribution with error=7.869721858237774e-07, AIC=4942.067199367857, BIC=4958.141924
640041)
Fitted hypsecant distribution with error=2.4842148820359246e-07, AIC=5253.590530154161, BIC=5269.6652
55426345)
Fitted invgamma distribution with error=3.1711912676862656e-08, AIC=4708.791311433008, BIC=4732.90339
9341284)
Fitted invgauss distribution with error=1.0534023978771831e-06, AIC=4364.658031750312, BIC=4388.77011
9658589)
Fitted invweibull distribution with error=3.2423857872532554e-08, AIC=4708.372033945147, BIC=4732.484
121853424)
Fitted johnsonsb distribution with error=3.667746688679689e-08, AIC=4985.824600621503, BIC=5017.97405
1165872)
Fitted johnsonsu distribution with error=3.2205670832465375e-08, AIC=4763.138900217261, BIC=4795.2883
5076163)
Fitted kappa3 distribution with error=5.941349765512793e-07, AIC=4765.687268337013, BIC=4789.79935624
5289)
Fitted kappa4 distribution with error=1.990973935021834e-06, AIC=4332.099604996703, BIC=4364.24905554
1072)
SKIPPED ksone distribution(taking more than 30 seconds)
Fitted kstwobign distribution with error=7.029039972655365e-08, AIC=5568.792192488694, BIC=5584.86691
7760878)
Fitted laplace distribution with error=2.579134426876058e-07, AIC=5090.327905358217, BIC=5106.4026306
30401)
Fitted levy distribution with error=1.210289758684869e-06, AIC=4201.2625525646945, BIC=4217.337277836
8785)
Fitted levy_l distribution with error=2.7331445884335217e-06, AIC=4563.865107721951, BIC=4579.9398329
94135)
SKIPPED levy_stable distribution(taking more than 30 seconds)
Fitted loggamma distribution with error=3.9219984419215424e-07, AIC=7070.413353542515, BIC=7094.52544
1450792)
Fitted logistic distribution with error=2.804304454107058e-07, AIC=5455.922249326968, BIC=5471.996974
599152)
Fitted loglaplace distribution with error=1.565797450534864e-07, AIC=4452.832438356378, BIC=4476.9445
26264654)
Fitted lognorm distribution with error=6.278113972983729e-06, AIC=4893.7270437179395, BIC=4917.839131
626216)
Fitted lomax distribution with error=1.3961790851885162e-06, AIC=4289.722941953179, BIC=4313.83502986
1455)
Fitted maxwell distribution with error=2.1442218403576233e-07, AIC=6322.742442174762, BIC=6338.817167
446946)
Fitted mielke distribution with error=3.218565690102756e-07, AIC=4230.252079514741, BIC=4262.40153005
911)
Fitted moyal distribution with error=2.7808754568983866e-08, AIC=4777.123854706727, BIC=4793.19857997
8911)
Fitted nakagami distribution with error=1.5659266690091577e-07, AIC=5944.036167300233, BIC=5968.14825
5208509)
SKIPPED ncf distribution(taking more than 30 seconds)
SKIPPED nct distribution(taking more than 30 seconds)
Fitted ncx2 distribution with error=3.1212493667984903e-06, AIC=inf, BIC=inf)
Fitted norm distribution with error=3.8082659237783485e-07, AIC=7174.242885008039, BIC=7190.317610280
223)
Fitted norminvgauss distribution with error=3.4673909536951494e-08, AIC=4801.439676228628, BIC=4833.5
89126772997)

In [29]:
```
f_NPEA.summary()
```

Out[29]:

	sumsquare_error	AIC	BIC
exponnorm	2.631107e-08	4713.408131	4737.520219
moyal	2.780875e-08	4777.123855	4793.198580
invgamma	3.171191e-08	4708.791311	4732.903399
genextreme	3.180476e-08	4740.224072	4764.336160
johnsonsu	3.220567e-08	4763.138900	4795.288351

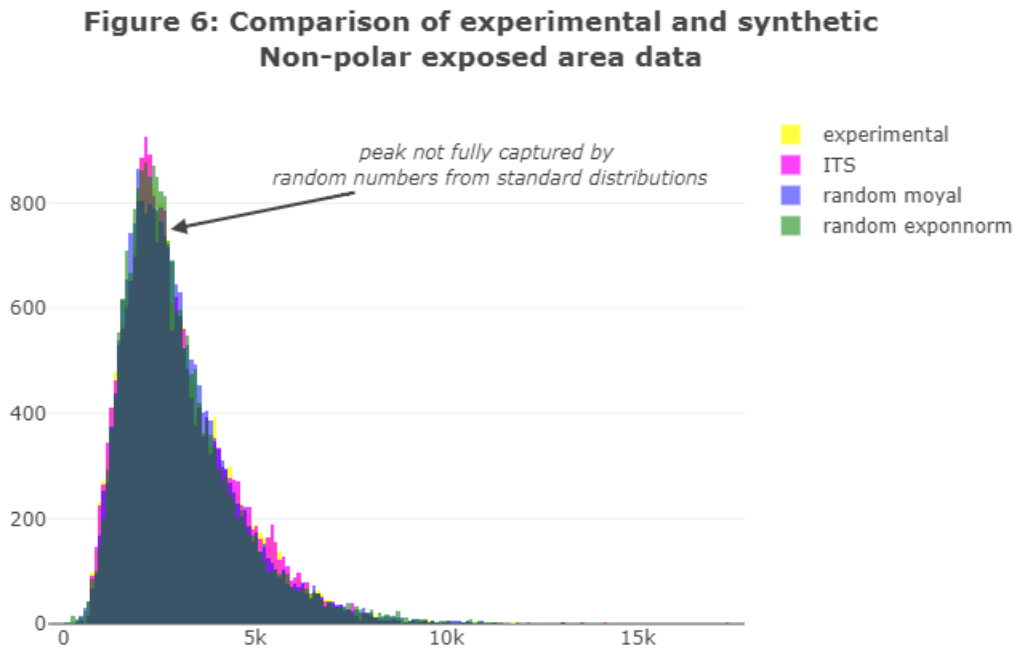


Similar to the Total Surface Area above, there are more than one distribution function that can adequately describe the Non-polar Exposed Area data. Hence, only the Moyal and exponentially modified Normal (exponnorm) distributions will be used to generate a synthetic data alongside ITS.

In [30]:
```
# create synthetic data
NonpolarExposedArea_random_moyal = moyal.rvs(f_NPEA.fitted_param['moyal'][0],
                                             f_NPEA.fitted_param['moyal'][1], len(df))

NonpolarExposedArea_random_exponnorm = exponnorm.rvs(f_NPEA.fitted_param['exponnorm'][0],
                                                     f_NPEA.fitted_param['exponnorm'][1],
                                                     f_NPEA.fitted_param['exponnorm'][2], len(df))

NonpolarExposedArea_ITS = inverse_transform_sampling(df['Non-polar exposed area'])

NonpolarExposedArea_random_moyal = NonpolarExposedArea_random_moyal.astype(dtype = 'float32', casting =
'same_kind')
NonpolarExposedArea_random_exponnorm = NonpolarExposedArea_random_exponnorm.astype(
                                    dtype = 'float32', casting = 'same_kind')
NonpolarExposedArea_ITS = NonpolarExposedArea_ITS.astype(dtype = 'float32', casting = 'same_kind')
```

In [110]:
```
# plot original and synthetic data

trace_experimental = go.Histogram(x = df['Non-polar exposed area'], opacity= 0.75, name = 'experimenta
l',
                                  marker = dict(color = 'yellow'))
trace_ITS = go.Histogram(x = NonpolarExposedArea_ITS, opacity= 0.75, name = 'ITS',
                                  marker = dict(color = 'magenta'))

trace_random1 = go.Histogram(x = NonpolarExposedArea_random_moyal, opacity= 0.5, name = 'random moyal'
,
                                  marker = dict(color = 'blue'))

trace_random2 = go.Histogram(x = NonpolarExposedArea_random_exponnorm, opacity= 0.55, name = 'random e
xponnorm',
                                  marker = dict(color = 'green'))

data = [trace_experimental, trace_ITS, trace_random1, trace_random2]

layout = go.Layout(barmode = 'overlay',
                   title = '<b>Figure 6: Comparison of experimental and synthetic<br>Non-polar exposed
area data</b>',
    annotations=[
        dict(
            x=2800,
            y=750,
            xref='x',
            yref='y',
            text='<i>peak not fully captured by <br>random numbers from standard distributions</i>',
            showarrow=True,
            arrowhead=2,
            arrowwidth = 2,
            arrowsize = 1,
            ax=200,
            ay=-40,
            align = 'center'
        )
    ]
)

fig = go.Figure(data=data, layout=layout)
# iplot(fig)

img_bytes = pio.to_image(fig, format='png')
Image(img_bytes)
```

Out[110]:



From Figure 6, we can conclude that the ITS and standard distributions result in relatively similar curves, although there are very minor differences from the experimental data.

Fractional area of exposed non polar residue

For this data, we will continue to employ the same strategy as for the previous variables.

In [32]:
```
f_FAENPR = Fitter(df['Fractional area of exposed non-polar residue'], bins=200)
f_FAENPR.fit()
```

```
Fitted alpha distribution with error=8.093314934614217, AIC=245.41622324479033, BIC=269.5283111530668
)
Fitted anglit distribution with error=518.0423459221691, AIC=-190.37863224910183, BIC=-174.3039069769
1752)
Fitted arcsine distribution with error=1975.9698967266681, AIC=-226.6751043467027, BIC=-210.600379074
5184)
Fitted argus distribution with error=1095.160141472109, AIC=-383.549129428025, BIC=-359.4370415197485
6)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\optimize\mi
npack.py:163: RuntimeWarning:

The iteration is not making good progress, as measured by the
  improvement from the last ten iterations.


Fitted beta distribution with error=7.87954626435779, AIC=261.25410040428494, BIC=293.40355094865356)
Fitted betaprime distribution with error=7.8345450444057345, AIC=255.19609634554755, BIC=287.34554688
991614)
Fitted bradford distribution with error=1006.1846060654832, AIC=-279.5469282556468, BIC=-255.43484034
73703)
Fitted burr distribution with error=19.549709652691494, AIC=128.14823360580164, BIC=160.2976841501702
6)
Fitted burr12 distribution with error=8.838581078355519, AIC=241.9881807931738, BIC=274.1376313375424
)
Fitted cauchy distribution with error=156.78430643259895, AIC=30.47356960090096, BIC=46.5482948730852
7)
Fitted chi distribution with error=8.213448269388154, AIC=282.1750499037281, BIC=306.2871378120046)
Fitted chi2 distribution with error=8.28066660780731, AIC=254.12117474437855, BIC=278.233262652655)
Fitted cosine distribution with error=1869.2415739533196, AIC=inf, BIC=inf)
Fitted crystalball distribution with error=13.667848040499118, AIC=309.3404679641393, BIC=341.4899185
085079)
Fitted dgamma distribution with error=45.653890336504105, AIC=132.82441121321963, BIC=156.93649912149
61)
Fitted dweibull distribution with error=29.295642705624573, AIC=184.64048633230476, BIC=208.752574240
58123)
Fitted erlang distribution with error=7.879308771682144, AIC=259.26540819816637, BIC=283.377496106442
83)
Fitted expon distribution with error=1331.5214125085556, AIC=-158.14564335487492, BIC=-142.0709180826
906)
```

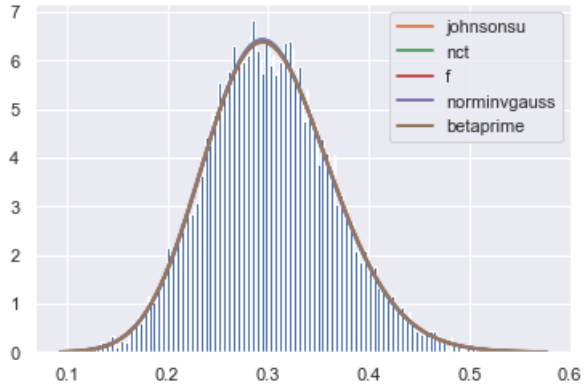
```
E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 59.5.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 62.66666666666667.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 61.25.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 60.5.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 60.125.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 61.41666666666667.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 59.777777777777786.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 60.88194444444444.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 59.25462962962963.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 59.79513888888889.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 60.87615740740741.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 60.06539351851852.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 60.5065586419753.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 60.22038966049382.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 59.30857767489712.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 59.701919367283956.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:
```

Fitted exponnorm distribution with error=8.426150721796114, AIC=210.10915515438938, BIC=234.221243062
66585)
Fitted exponpow distribution with error=130.1946581854532, AIC=793.038538752462, BIC=817.150626660738
5)
Fitted exponweib distribution with error=7.85984052055882, AIC=256.41482317019234, BIC=288.5642737145
6095)
Fitted f distribution with error=7.832769338429533, AIC=254.90030142340368, BIC=287.0497519677723)
Fitted fatiguelife distribution with error=7.855780275991185, AIC=256.358650243396, BIC=280.470738151
67255)
Fitted fisk distribution with error=18.883003887562676, AIC=125.25054419966453, BIC=149.362632107941)
Fitted foldcauchy distribution with error=142.04042443297675, AIC=-22.891453335302316, BIC=1.22063457
29741486)
Fitted foldnorm distribution with error=1020.8261931414561, AIC=-181.73499652468, BIC=-157.6229086164
0355)
Fitted frechet_l distribution with error=11.102124859334861, AIC=331.99680155245414, BIC=356.10888946
07306)
Fitted frechet_r distribution with error=23.294218519079777, AIC=400.85060741953316, BIC=424.96269532
78096)
Fitted gamma distribution with error=7.879309851364153, AIC=259.2652981052588, BIC=283.3773860135353)
Fitted gausshyper distribution with error=10.83956176377853, AIC=336.14550570617774, BIC=384.36968152
27307)
Fitted genexpon distribution with error=450.91941830652866, AIC=-150.94088521082128, BIC=-110.7540720
303605)
Fitted genextreme distribution with error=11.104377324355454, AIC=331.90977250760506, BIC=356.0218604
158815)
Fitted gengamma distribution with error=8.065536797848166, AIC=275.8222268328155, BIC=307.97167737718
41)
Fitted genhalflogistic distribution with error=681.4517224694791, AIC=-184.78823943765138, BIC=-160.6
7615152937492)
Fitted genlogistic distribution with error=15.380307936771537, AIC=138.98632506050535, BIC=163.098412
96878182)
Fitted gennorm distribution with error=13.674290607895106, AIC=283.87914014538114, BIC=307.9912280536
576)
Fitted genpareto distribution with error=954.1879982241264, AIC=-220.2666702748827, BIC=-196.15458236
660623)
Fitted gilbrat distribution with error=972.2003469923948, AIC=-119.75888892101902, BIC=-103.684163648
83471)
Fitted gompertz distribution with error=708.2430202574253, AIC=-95.94831127555486, BIC=-71.8362233672
784)
Fitted gumbel_l distribution with error=162.04163544186784, AIC=963.6374534493667, BIC=979.7121787215
51)
Fitted gumbel_r distribution with error=63.45009154244226, AIC=300.0635656475641, BIC=316.13829091974
84)
Fitted halfcauchy distribution with error=1202.7507400611337, AIC=-98.34636397563604, BIC=-82.2716387
0345173)
Fitted halfgennorm distribution with error=694.1092262825813, AIC=267.58061808269656, BIC=291.6927059
90973)
Fitted halflogistic distribution with error=1085.4200985526259, AIC=-181.63030990134638, BIC=-165.555
58462916207)
Fitted halfnorm distribution with error=1036.8651929642292, AIC=-186.15062281198806, BIC=-170.0758975
3980375)
Fitted hypsecant distribution with error=40.44407222188805, AIC=106.66897288580877, BIC=122.743698157
99308)
Fitted invgamma distribution with error=7.870136000724316, AIC=250.51583579335397, BIC=274.6279237016
3044)
Fitted invgauss distribution with error=10.408517342381424, AIC=241.37916544923812, BIC=265.491253357
51455)
Fitted invweibull distribution with error=63.432248509224486, AIC=302.13673675537376, BIC=326.2488246
636502)
Fitted johnsonsb distribution with error=7.877314963933088, AIC=262.48664345169766, BIC=294.636093996
0663)
Fitted johnsonsu distribution with error=7.806415273804277, AIC=248.29035123202434, BIC=280.439801776
393)
Fitted kappa3 distribution with error=675.0133619787822, AIC=55.759450410185764, BIC=79.8715383184622
3)
Fitted kappa4 distribution with error=1078.3599606922337, AIC=-239.58230223021835, BIC=-207.432851685
84973)
SKIPPED ksone distribution(taking more than 30 seconds)
Fitted kstwobign distribution with error=58.44479159067671, AIC=nan, BIC=nan)
Fitted laplace distribution with error=136.12157388952693, AIC=71.7361076759341, BIC=87.8108329481184
)
Fitted levy distribution with error=1190.1999891847463, AIC=145.10761896020264, BIC=161.1823442323869
5)
Fitted levy_l distribution with error=1351.3736021460131, AIC=236.2126400352878, BIC=252.287365307472
1)
SKIPPED levy_stable distribution(taking more than 30 seconds)
Fitted loggamma distribution with error=14.945044289799458, AIC=303.37230693380627, BIC=327.484394842
08274)
Fitted logistic distribution with error=22.07443711445923, AIC=148.1385164961098, BIC=164.21324176829
41)
Fitted loglaplace distribution with error=133.93030486374082, AIC=55.932742860405526, BIC=80.04483076
868199)

In [33]:
```
f_FAENPR.summary()
```

Out[33]:

	sumsquare_error	AIC	BIC
johnsonsu	7.806415	248.290351	280.439802
nct	7.813336	241.540548	273.689998
f	7.832769	254.900301	287.049752
norminvgauss	7.834372	234.088729	266.238180
betaprime	7.834545	255.196096	287.345547



Given the shape and error, the Johnson SU (unbounded support) (https://www.rdocumentation.org/packages/ExtDist/versions/0.6-3/topics/JohnsonSU) distribution is the best fit. In addition, there are no atypical 'features' such as a peak shoulder that needs to be included. Interestingly, the shape of the histogram looks like a normal distribution, but somehow this is not showing up in the fit summary, perhaps due a slightly larger error than a pure Johnson SU. For completeness, we will include a normal distribution as well in the comparison plot.

In [34]:
```
# create synthetic data

FractionalAreaExposedNonPolarResidue_random_johnsonsu = johnsonsu.rvs(f_FAENPR.fitted_param['johnsonsu'
][0],
                                                                      f_FAENPR.fitted_param['johnsonsu'
][1],
                                                                      f_FAENPR.fitted_param['johnsonsu'
][2],
                                                                      f_FAENPR.fitted_param['johnsonsu'
][3], len(df))

FractionalAreaExposedNonPolarResiduea_random_norm = norm.rvs(f_FAENPR.fitted_param['norm'][0],
                                                             f_FAENPR.fitted_param['norm'][1], len(df))

FractionalAreaExposedNonPolarResidue_ITS = inverse_transform_sampling(df['Fractional area of exposed no
n-polar residue'])

FractionalAreaExposedNonPolarResidue_random_johnsonsu = FractionalAreaExposedNonPolarResidue_random_joh
nsonsu.astype(
                                    dtype = 'float32', casting = 'same_kind')
FractionalAreaExposedNonPolarResiduea_random_norm = FractionalAreaExposedNonPolarResiduea_random_norm.a
stype(
                                    dtype = 'float32', casting = 'same_kind')
FractionalAreaExposedNonPolarResidue_ITS = FractionalAreaExposedNonPolarResidue_ITS.astype(
                                    dtype = 'float32', casting = 'same_kind')
```

In [111]:
```
# plot original and synthetic data

trace_experimental = go.Histogram(x = df['Fractional area of exposed non-polar residue'], opacity= 0.7
5,
                                  name = 'experimental',
                                  marker = dict(color = 'yellow'))

trace_ITS = go.Histogram(x = FractionalAreaExposedNonPolarResidue_ITS, opacity= 0.75, name = 'ITS',
                                  marker = dict(color = 'magenta'))

trace_random1 = go.Histogram(x = FractionalAreaExposedNonPolarResidue_random_johnsonsu, opacity= 0.5,
                                  name = 'random Johnson SU',
                                  marker = dict(color = 'blue'))

trace_random2 = go.Histogram(x = FractionalAreaExposedNonPolarResiduea_random_norm, opacity= 0.55,
                                  name = 'random normal',
                                  marker = dict(color = 'green'))

data = [trace_experimental, trace_ITS, trace_random1, trace_random2]

layout = go.Layout(barmode = 'overlay',
                   title = '<b>Figure 7: Comparison of experimental and synthetic <br>' +
                   'Fractional area of exposed non-polar residue data</b>')

fig = go.Figure(data=data, layout=layout)
# iplot(fig)
img_bytes = pio.to_image(fig, format='png')
Image(img_bytes)
```

Out[111]:

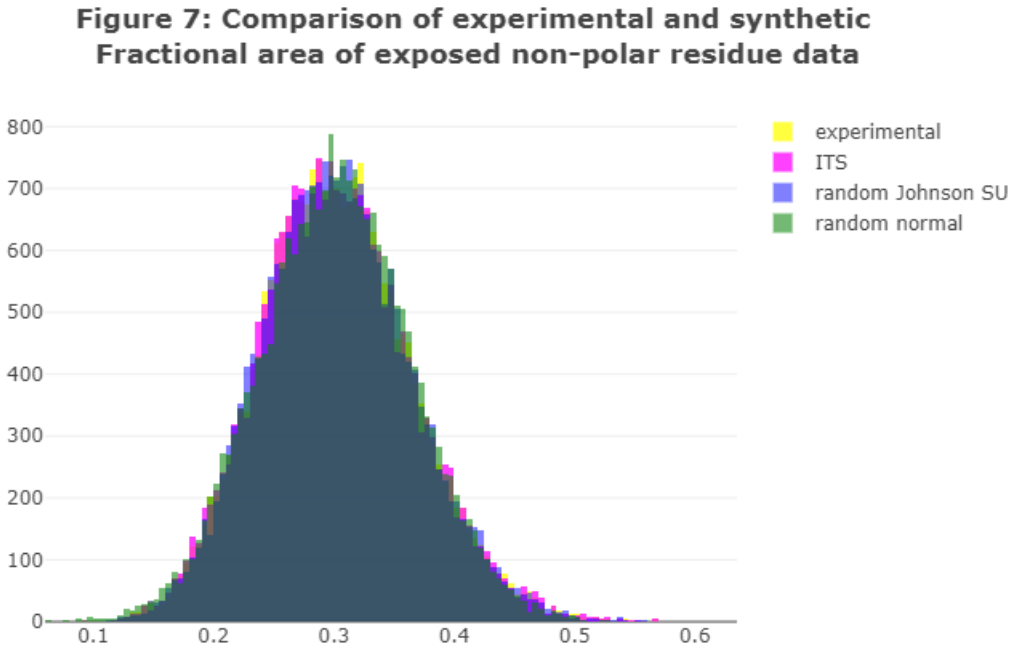


Figure 7 above shows that indeed, a normal distribution is just as good as Johnson SU. As normality is assumed in many parametric statistical tests, and as non-parametric ones are less powerful, the ability to fit a normal distribution can be considered crucial so long as no information is lost.

Fractional area of exposed non polar part of residue

In this section, we will look at Fractional area of exposed non polar part of residue, which is related but is not the same as Fractional area of exposed non-polar residue data.

In [36]:
```
f_FAENPRR = Fitter(df['Fractional area of exposed non-polar part of residue'], bins=200)
f_FAENPRR.fit()
```

```
Fitted alpha distribution with error=0.00025304709427461525, AIC=2769.7804912365746, BIC=2793.8925791
44851)
Fitted anglit distribution with error=0.002008470866384027, AIC=2450.153239309435, BIC=2466.227964581
6193)
Fitted arcsine distribution with error=0.0036945793045973993, AIC=2415.5146774161085, BIC=2431.589402
688293)
Fitted argus distribution with error=0.004009157620899597, AIC=2177.3743636857175, BIC=2201.486451593
994)
Fitted beta distribution with error=0.00037589089417974065, AIC=2825.1463732782167, BIC=2857.29582382
25856)
Fitted betaprime distribution with error=0.00023328097937338823, AIC=2780.186384945789, BIC=2812.3358
354901575)
Fitted bradford distribution with error=0.0017685140792916654, AIC=2397.6985620142464, BIC=2421.81064
99225228)
Fitted burr distribution with error=0.00021963268896040106, AIC=2755.1067513968637, BIC=2787.25620194
12326)
Fitted burr12 distribution with error=0.0002565498968821978, AIC=2750.393588362407, BIC=2782.54303890
6776)
Fitted cauchy distribution with error=0.000544972746097654, AIC=2833.771168759765, BIC=2849.845894031
9492)
Fitted chi distribution with error=0.0005877201191122133, AIC=2933.3625024706307, BIC=2957.4745903789
07)
Fitted chi2 distribution with error=0.0003746972358533645, AIC=2831.969500930234, BIC=2856.0815888385
105)
Fitted cosine distribution with error=0.0016661462555207845, AIC=2588.033630623551, BIC=2604.10835589
57353)
Fitted crystalball distribution with error=0.0009748540026820009, AIC=3160.7713699484448, BIC=3192.92
08204928136)
Fitted dgamma distribution with error=0.0006266197558217596, AIC=2949.409234665914, BIC=2973.52132257
41904)
Fitted dweibull distribution with error=0.0006668778380328486, AIC=2952.296143267161, BIC=2976.408231
1754373)
```

```
E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.0716049382715758.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.367901234567883.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.6786008230452385.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.51687242798352.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.4120713305898347.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.5863511659807723.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.4423868312757016.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.608436213991748.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.580658436213969.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.4181755829903793.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.337631458619098.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.5199016918152513.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.5275796372503985.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.5701760402377474.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.4551834070009475.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.503722120611675.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:
```

Fitted erlang distribution with error=0.048191683821360014, AIC=19560.59638654911, BIC=19584.708474457388)
Fitted expon distribution with error=0.0018625354555424612, AIC=2588.5294422270817, BIC=2604.604167499266)
Fitted exponnorm distribution with error=0.00019385308936752045, AIC=2752.6870907551815, BIC=2776.799178663458)
Fitted exponpow distribution with error=0.0081690121447998, AIC=3444.0577387422377, BIC=3468.169826650514)
Fitted exponweib distribution with error=0.0002336878419602801, AIC=2799.9724702692447, BIC=2832.1219208136135)
Fitted f distribution with error=0.0002419308099177826, AIC=2795.158197791571, BIC=2827.30764833594)
Fitted fatiguelife distribution with error=0.00024549449529689304, AIC=2832.8773405447682, BIC=2856.9894284530446)
Fitted fisk distribution with error=0.00026139036899501517, AIC=2760.093514307253, BIC=2784.2056022155293)
Fitted foldcauchy distribution with error=0.0005025291737661687, AIC=2720.1886551700686, BIC=2744.300743078345)
Fitted foldnorm distribution with error=0.00744641240274314, AIC=17689.837540684843, BIC=17713.94962859312)
Fitted frechet_l distribution with error=0.009043141049108269, AIC=3223.855594619624, BIC=3247.9676825279003)
Fitted frechet_r distribution with error=0.01576114229280942, AIC=3104.7942781679203, BIC=3128.9063660761967)
Fitted gamma distribution with error=0.0005159902388595797, AIC=2922.603197417442, BIC=2946.7152853257185)
Fitted gausshyper distribution with error=0.000404808771144463, AIC=2862.074652338537, BIC=2910.2988281550897)
Fitted genexpon distribution with error=0.0017953567984042187, AIC=2595.509743268086, BIC=2635.6965564485467)
Fitted genextreme distribution with error=0.0014865696269272886, AIC=3041.634132001052, BIC=3065.746219909328 4)
Fitted gengamma distribution with error=0.00035099459148717053, AIC=2804.411532544723, BIC=2836.5609830890917)
Fitted genhalflogistic distribution with error=0.0010714346770421452, AIC=2854.8864912910312, BIC=2878.9985791993076)
Fitted genlogistic distribution with error=0.00042508675043972965, AIC=2864.6132577334747, BIC=2888.725345641751)
Fitted gennorm distribution with error=0.000694684580206797, AIC=2955.6489129363026, BIC=2979.761000844579)
Fitted genpareto distribution with error=0.0014174968037223609, AIC=2667.4254028710598, BIC=2691.537490779336)
Fitted gilbrat distribution with error=0.0009599513236783397, AIC=2574.493095000348, BIC=2590.567820 2725326)
Fitted gompertz distribution with error=0.029933082889680506, AIC=inf, BIC=inf)
Fitted gumbel_l distribution with error=0.00149572422859969, AIC=3642.9700882846187, BIC=3659.044813556803)
Fitted gumbel_r distribution with error=0.00042567239923392623, AIC=2860.7753273164253, BIC=2876.850052 5886097)
Fitted halfcauchy distribution with error=0.0016811757868373399, AIC=2601.291511051738, BIC=2617.366236 3239223)
Fitted halfgennorm distribution with error=0.004449667388395659, AIC=2648.725382520812, BIC=2672.837470 4290886)
Fitted halflogistic distribution with error=0.0012833584621181034, AIC=2628.6436637861707, BIC=2644.718389058355)
Fitted halfnorm distribution with error=0.0011929415221878115, AIC=2699.578366828604, BIC=2715.653092 1007884)
Fitted hypsecant distribution with error=0.0007331099893430096, AIC=2949.326887316906, BIC=2965.401612 5890903)
Fitted invgamma distribution with error=0.0002298003164489052, AIC=2776.919997444582, BIC=2801.032085 3528583)
Fitted invgauss distribution with error=0.0009410413446148441, AIC=2764.675102818478, BIC=2788.787190 7267545)
Fitted invweibull distribution with error=0.0002314155203849677, AIC=2766.7720411597174, BIC=2790.884 129067994)
Fitted johnsonsb distribution with error=0.00026010598896382913, AIC=2823.921773078988, BIC=2856.0712 236233567)
Fitted johnsonsu distribution with error=0.00022883866195651698, AIC=2793.751838432681, BIC=2825.9012 889770493)
Fitted kappa3 distribution with error=0.0010617240342108943, AIC=2727.6200879496364, BIC=2751.7321758 57913)
Fitted kappa4 distribution with error=0.0022340078862224133, AIC=2561.993066279346, BIC=2594.1425168237147)
SKIPPED ksone distribution(taking more than 30 seconds)
Fitted kstwobign distribution with error=0.0004491948880393574, AIC=2923.928510736281, BIC=2940.0032360084656)
Fitted laplace distribution with error=0.0006741607636008092, AIC=2914.3930114557525, BIC=2930.467736 727937)
Fitted levy distribution with error=0.0015984500837467445, AIC=2708.2813290426475, BIC=2724.356054314832)
Fitted levy_l distribution with error=0.0032176151974735285, AIC=2982.537527632552, BIC=2998.6122529047366)
SKIPPED levy_stable distribution(taking more than 30 seconds)

```
E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5166: RuntimeWarning:

invalid value encountered in multiply


Fitted powerlognorm distribution with error=0.0002692757120448146, AIC=3020.402908736808, BIC=3052.55
2359281177)
Fitted powernorm distribution with error=nan, AIC=3010.337874785678, BIC=3034.4499626939546)
Fitted rayleigh distribution with error=0.0006287192204175114, AIC=2982.8969714358586, BIC=2998.97169
6708043)
Fitted rdist distribution with error=0.0009762871216819727, AIC=3160.2779684195366, BIC=3184.39005632
7813)
Fitted recipinvgauss distribution with error=0.0002554511103345437, AIC=2838.5934445075272, BIC=2862.
7055324158036)
Fitted reciprocal distribution with error=0.003952407706447001, AIC=inf, BIC=inf)
Fitted rice distribution with error=0.0006287193068990637, AIC=2984.89672138149, BIC=3009.00880928976
63)
SKIPPED rv_continuous distribution(taking more than 30 seconds)
SKIPPED rv_histogram distribution(taking more than 30 seconds)
Fitted semicircular distribution with error=0.0022889108220409005, AIC=2384.0466657196084, BIC=2400.1
21390991793)
Fitted skewnorm distribution with error=0.00034952256682956273, AIC=2891.056912966944, BIC=2915.16900
08752203)
Fitted t distribution with error=0.0007784163565085792, AIC=2930.9933411984607, BIC=2955.105429106737
)
Fitted trapz distribution with error=0.003728455854650649, AIC=2492.124724120262, BIC=2524.2741746646
307)
Fitted triang distribution with error=0.0010328252719446812, AIC=2480.046823022032, BIC=2504.15891093
03086)
Fitted truncexpon distribution with error=0.0018248441449199501, AIC=2436.0099824168974, BIC=2460.122
070325174)
Fitted truncnorm distribution with error=0.003952407706447001, AIC=inf, BIC=inf)
SKIPPED tukeylambda distribution(taking more than 30 seconds)
Fitted uniform distribution with error=0.002397366687371503, AIC=2356.91414428404, BIC=2372.988869556
2245)
Fitted vonmises distribution with error=2.3994500660263047e+54, AIC=-23485.52782799603, BIC=-23461.41
5740087752)
Fitted vonmises_line distribution with error=0.0008875185813483508, AIC=2941.4276950595076, BIC=2965.
539782967784)
Fitted wald distribution with error=0.0006318771465404505, AIC=2665.6342177562037, BIC=2681.708943028
388)
Fitted weibull_max distribution with error=0.009043141049108269, AIC=3223.855594619624, BIC=3247.9676
825279003)
Fitted weibull_min distribution with error=0.01576114229280942, AIC=3104.7942781679203, BIC=3128.9063
660761967)
Fitted wrapcauchy distribution with error=nan, AIC=nan, BIC=nan)
```

In [37]:
```
f_FAENPRR.summary()
```

Out[37]:

	sumsquare_error	AIC	BIC
exponnorm	0.000194	2752.687091	2776.799179
burr	0.000220	2755.106751	2787.256202
mielke	0.000220	2755.112937	2787.262388
johnsonsu	0.000229	2793.751838	2825.901289
invgamma	0.000230	2776.919997	2801.032085

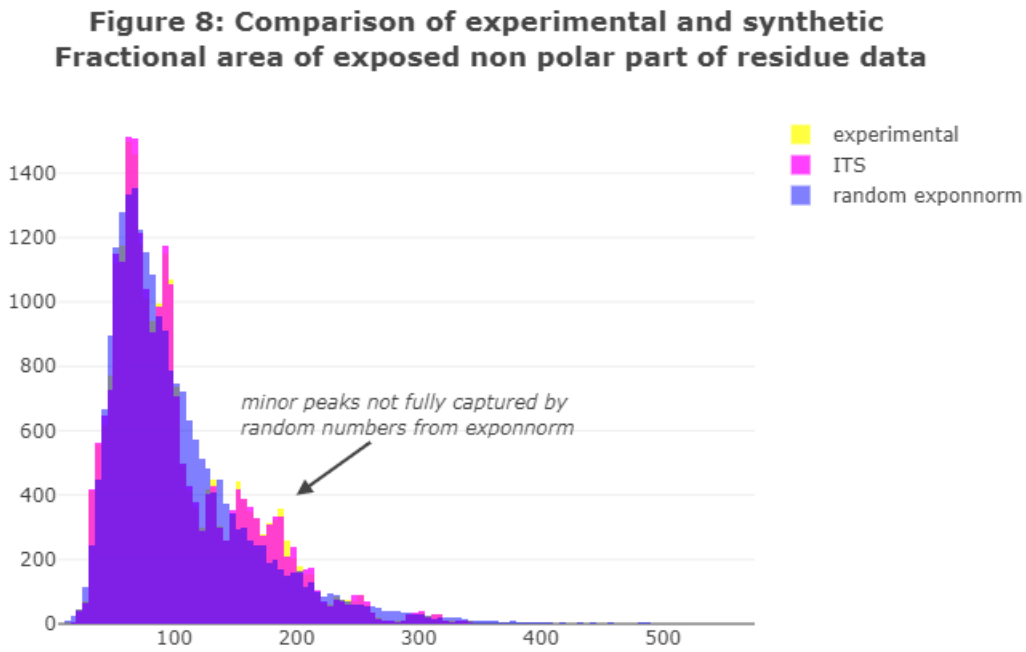
This data is a bit complicated to model adequately with the presence of several minor peaks. While this is another example where ITS will be quite useful, it is also possible that these are outliers (see Figure 1).

As the exponentially-modified normal distribution (exponnorm) has the lowest error, this will be used to generate the synthetic dataset along with ITS.

In [38]:
```
# create synthetic data

FractionalAreaExposedNonPolarPartOfResidue_random_exponnorm = exponnorm.rvs(f_FAENPRR.fitted_param['exponnorm'][0],
                                                                            f_FAENPRR.fitted_param['exponnorm'][1],
                                                                            f_FAENPRR.fitted_param['exponnorm'][2], len(df))

FractionalAreaExposedNonPolarPartOfResidue_ITS = inverse_transform_sampling(
                                                    df['Fractional area of exposed non-polar part of residue'])

FractionalAreaExposedNonPolarPartOfResidue_random_exponnorm = FractionalAreaExposedNonPolarPartOfResidue_random_exponnorm.astype(
                                    dtype = 'float32', casting = 'same_kind')

FractionalAreaExposedNonPolarPartOfResidue_ITS = FractionalAreaExposedNonPolarPartOfResidue_ITS.astype(
                                    dtype = 'float32', casting = 'same_kind')
```

In [112]:
```
# plot original and synthetic data

trace_experimental = go.Histogram(x = df['Fractional area of exposed non-polar part of residue'], opac
ity= 0.75,
                                  name = 'experimental',
                                  marker = dict(color = 'yellow'))

trace_ITS = go.Histogram(x = FractionalAreaExposedNonPolarPartOfResidue_ITS, opacity= 0.75,
                                  name = 'ITS',
                                  marker = dict(color = 'magenta'))

trace_random1 = go.Histogram(x = FractionalAreaExposedNonPolarPartOfResidue_random_exponnorm, opacity=
0.5,
                                  name = 'random exponnorm',
                                  marker = dict(color = 'blue'))

data = [trace_experimental, trace_ITS, trace_random1]

layout = go.Layout(barmode = 'overlay', title = '<b>Figure 8: Comparison of experimental and synthetic
<br>' +
                   'Fractional area of exposed non polar part of residue data</b>',
    annotations=[
        dict(
            x=200,
            y=400,
            xref='x',
            yref='y',
            text='<i>minor peaks not fully captured by <br>random numbers from exponnorm</i>',
            showarrow=True,
            arrowhead=2,
            arrowwidth = 2,
            arrowsize = 1,
            ax=70,
            ay=-50,
            align = 'center'
        )
    ]
)

fig = go.Figure(data=data, layout=layout)
# iplot(fig)
img_bytes = pio.to_image(fig, format='png')
Image(img_bytes)
```

Out[112]:



Again, we see that the random numbers from exponnorm distribution does not fully capture the subtle nuances of the data, especially the minor peaks at around x = 200 and the split main peak at around x = 60 and 90. While it could be that some of these are outliers, we cannot be entirely certain of this.

Molecular mass weighted exposed area

Let's now look at the molecular mass weighted exposed area. As before, we will try automatic fitting with Fitter first.

In [40]:
```
f_MMWEA = Fitter(df['Molecular mass weighted exposed area'], bins=200)
f_MMWEA.fit()
```

Fitted alpha distribution with error=5.419569990177882e-13, AIC=6772.014922402528, BIC=6796.127010310
8045)
Fitted anglit distribution with error=1.3724614465804362e-11, AIC=6265.185900090153, BIC=6281.2606253
62337)
Fitted arcsine distribution with error=1.8922602974912172e-11, AIC=6243.237683469111, BIC=6259.312408
741295)
Fitted argus distribution with error=2.126735022676452e-11, AIC=6193.219355600796, BIC=6217.331443509
072)
Fitted beta distribution with error=5.284863494351797e-13, AIC=6939.403954124196, BIC=6971.5534046685
65)
Fitted betaprime distribution with error=4.4584584731712474e-13, AIC=6742.265995471169, BIC=6774.4154
46015538)
Fitted bradford distribution with error=1.1871269959307063e-11, AIC=6244.520207553312, BIC=6268.63229
5461588)
Fitted burr distribution with error=9.930645347108423e-13, AIC=6628.408900674633, BIC=6660.5583512190
01)
Fitted burr12 distribution with error=7.278548797280517e-13, AIC=6730.539164033831, BIC=6762.68861457
82)
Fitted cauchy distribution with error=2.3536534629202915e-12, AIC=6701.430489703529, BIC=6717.5052149
75713)
Fitted chi distribution with error=2.4846830041211905e-11, AIC=inf, BIC=inf)
Fitted chi2 distribution with error=2.4666407869831797e-11, AIC=7986.445712864861, BIC=8010.557800773
137)
Fitted cosine distribution with error=1.1304141950606558e-11, AIC=6412.7980808271495, BIC=6428.872806
0993335)
Fitted crystalball distribution with error=2.4846830041211905e-11, AIC=inf, BIC=inf)
Fitted dgamma distribution with error=2.406363926380831e-12, AIC=7145.283280178464, BIC=7169.39536808
6741)
Fitted dweibull distribution with error=2.3160962051700703e-12, AIC=7260.88967958674, BIC=7285.001767
495017)

```
E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 5.783333333333362.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 6.650000000000038.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 7.100000000000044.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 8.50000000000006.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 9.050000000000056.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 9.000000000000064.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 8.300000000000054.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 9.650000000000063.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 7.400000000000044.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 7.400000000000043.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 8.07500000000005.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 6.0000000000000355.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 8.287500000000051.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 8.741666666666719.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 7.510416666666712.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 7.39027777777782.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:
```

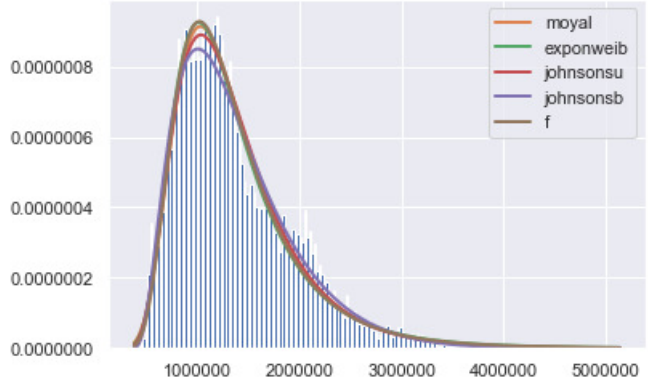
Fitted erlang distribution with error=2.4846830041211905e-11, AIC=inf, BIC=inf)
Fitted expon distribution with error=1.0563275600050806e-11, AIC=6486.774278373069, BIC=6502.849003645253)
Fitted exponnorm distribution with error=4.919226803654785e-13, AIC=6757.447249965391, BIC=6781.559337873668)
Fitted exponpow distribution with error=2.4846830041211905e-11, AIC=inf, BIC=inf)
Fitted exponweib distribution with error=4.2637656475890215e-13, AIC=6729.508942474073, BIC=6761.6583930184415)
Fitted f distribution with error=4.425238893562557e-13, AIC=6746.007321779453, BIC=6778.156772323822)
Fitted fatiguelife distribution with error=2.0543819774889403e-11, AIC=6768.359108773927, BIC=6792.471196682203)
Fitted fisk distribution with error=7.735702901977203e-13, AIC=6746.823349708494, BIC=6770.935437616767)
Fitted foldcauchy distribution with error=1.9670111178029725e-12, AIC=6625.7568869145025, BIC=6649.868974822779)
Fitted foldnorm distribution with error=2.4846830041211905e-11, AIC=inf, BIC=inf)
Fitted frechet_l distribution with error=2.7193148630727724e-11, AIC=7470.48206580324, BIC=7494.594153711517)
Fitted frechet_r distribution with error=2.6896500421199422e-11, AIC=7419.057764262285, BIC=7443.169852170561)
Fitted gamma distribution with error=5.249188442647711e-13, AIC=6958.820905485263, BIC=6982.93299339354)
Fitted gausshyper distribution with error=6.069185853789683e-13, AIC=7063.945004165989, BIC=7112.169179982542)
Fitted genexpon distribution with error=1.009858263387315e-11, AIC=6469.794928411387, BIC=6509.981741591848)
Fitted genextreme distribution with error=4.926597518173774e-13, AIC=6788.59412279935, BIC=6812.706210707626)
Fitted gengamma distribution with error=3.5942572031894675e-11, AIC=7154.286843390939, BIC=7186.436293935308)
Fitted genhalflogistic distribution with error=5.197038486942045e-12, AIC=6827.087601478628, BIC=6851.1996893869045)
Fitted genlogistic distribution with error=7.301418254636812e-13, AIC=6953.446254722001, BIC=6977.558342630277)
Fitted gennorm distribution with error=2.45939847410215e-12, AIC=7422.440190793311, BIC=7446.552278701587)
Fitted genpareto distribution with error=9.640059898243597e-12, AIC=6568.318731752317, BIC=6592.430819660593)
Fitted gilbrat distribution with error=4.8159158177416294e-12, AIC=6426.500612614939, BIC=6442.575337887123)
Fitted gompertz distribution with error=2.4846830041211905e-11, AIC=inf, BIC=inf)
Fitted gumbel_l distribution with error=6.488258346857035e-12, AIC=12545.26985522863, BIC=12561.344580500814)
Fitted gumbel_r distribution with error=7.294769748733967e-13, AIC=6950.075772157928, BIC=6966.150497430112)
Fitted halfcauchy distribution with error=9.294130293451756e-12, AIC=6435.7398938928345, BIC=6451.8146191650185)
Fitted halfgennorm distribution with error=4.3942632817860393e-11, AIC=6556.779949784276, BIC=6580.892037692553)
Fitted halflogistic distribution with error=6.5639408621156e-12, AIC=6594.153462102483, BIC=6610.228187374667)
Fitted halfnorm distribution with error=2.4846830041211905e-11, AIC=inf, BIC=inf)
Fitted hypsecant distribution with error=2.278723859637027e-12, AIC=7122.904350045845, BIC=7138.979075318029)
Fitted invgamma distribution with error=4.4392378104061203e-13, AIC=6743.4283466101915, BIC=6767.540434518468)
Fitted invgauss distribution with error=9.046414976090571e-12, AIC=6508.29689430114, BIC=6532.408982209 4165)
Fitted invweibull distribution with error=4.875070243647544e-13, AIC=6785.794291843342, BIC=6809.906379751618)
Fitted johnsonsb distribution with error=4.3544335584081457e-13, AIC=7063.120755861222, BIC=7095.270206405591)
Fitted johnsonsu distribution with error=4.277905650358051e-13, AIC=6807.157170349413, BIC=6839.306620893782)
Fitted kappa3 distribution with error=5.6295288513847014e-12, AIC=6788.362960730701, BIC=6812.475048638978)
Fitted kappa4 distribution with error=1.372855083529926e-11, AIC=6429.8521235617945, BIC=6462.001574106163)
SKIPPED ksone distribution(taking more than 30 seconds)
Fitted kstwobign distribution with error=7.102703969442474e-13, AIC=7213.402973993067, BIC=7229.477699265 2505)
Fitted laplace distribution with error=2.301319668461389e-12, AIC=7028.7894949075335, BIC=7044.864220179717)
Fitted levy distribution with error=9.046413008415787e-12, AIC=6506.29700516089, BIC=6522.371730433074)
Fitted levy_l distribution with error=2.0917150643072574e-11, AIC=6840.2044530525445, BIC=6856.279178 3247285)
SKIPPED levy_stable distribution(taking more than 30 seconds)
Fitted loggamma distribution with error=3.184091530963608e-12, AIC=8017.8988126206095, BIC=8042.010900528886)
Fitted logistic distribution with error=2.500316249675612e-12, AIC=7244.095094452263, BIC=7260.169819724447)

```
E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5477: RuntimeWarning:

divide by zero encountered in true_divide

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5477: RuntimeWarning:

divide by zero encountered in log

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5477: RuntimeWarning:

invalid value encountered in subtract


Fitted recipinvgauss distribution with error=2.4846830040303392e-11, AIC=inf, BIC=inf)
Fitted reciprocal distribution with error=2.4846830041211905e-11, AIC=inf, BIC=inf)
Fitted rice distribution with error=1.3033260030331103e-12, AIC=7450.859090023493, BIC=7474.971177931
7695)
SKIPPED rv_continuous distribution(taking more than 30 seconds)
SKIPPED rv_histogram distribution(taking more than 30 seconds)
Fitted semicircular distribution with error=1.5592891170199325e-11, AIC=6191.952862220573, BIC=6208.0
275874927565)
Fitted skewnorm distribution with error=5.751820724771507e-13, AIC=7174.885761591744, BIC=7198.997849
5000205)
Fitted t distribution with error=1.383443673862618e-10, AIC=7011.6271814467345, BIC=7035.739269355011
)
Fitted trapz distribution with error=2.481734801161656e-11, AIC=6293.185714735391, BIC=6325.335165279
76)
Fitted triang distribution with error=7.382745453938108e-12, AIC=6278.825933629583, BIC=6302.93802153
786)
Fitted truncexpon distribution with error=1.2027280362192033e-11, AIC=6241.654280053761, BIC=6265.766
367962037)
Fitted truncnorm distribution with error=2.4846830041211905e-11, AIC=inf, BIC=inf)
SKIPPED tukeylambda distribution(taking more than 30 seconds)
Fitted uniform distribution with error=1.6080614482200282e-11, AIC=6155.686658086381, BIC=6171.761383
358565)
Fitted vonmises distribution with error=9.85636505935224e+45, AIC=-19748.573995135404, BIC=-19724.461
907227127)
Fitted vonmises_line distribution with error=2.8670750750539246e-12, AIC=7369.290364050141, BIC=7393.
402451958417)
Fitted wald distribution with error=3.1814682010126693e-12, AIC=6507.499362532382, BIC=6523.574087804
566)
Fitted weibull_max distribution with error=2.7193148630727724e-11, AIC=7470.48206580324, BIC=7494.594
153711517)
Fitted weibull_min distribution with error=2.6896500421199422e-11, AIC=7419.057764262285, BIC=7443.16
9852170561)
Fitted wrapcauchy distribution with error=nan, AIC=nan, BIC=nan)
```

In [41]:
```
f_MMWEA.summary()
```

Out[41]:

	sumsquare_error	AIC	BIC
moyal	4.188684e-13	6806.255648	6822.330374
exponweib	4.263766e-13	6729.508942	6761.658393
johnsonsu	4.277906e-13	6807.157170	6839.306621
johnsonsb	4.354434e-13	7063.120756	7095.270206
f	4.425239e-13	6746.007322	6778.156772

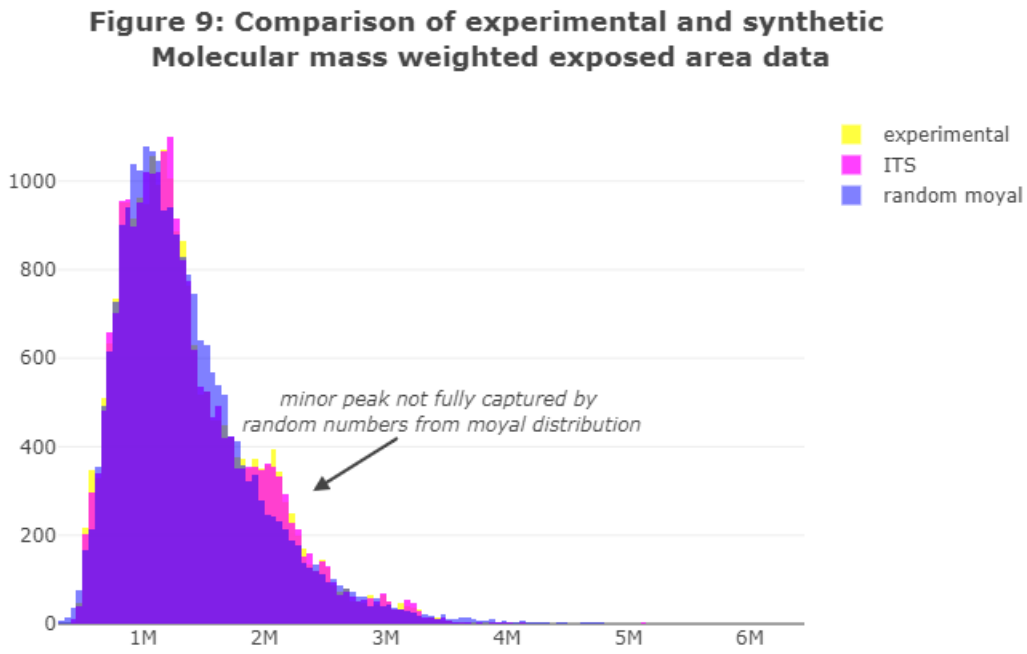


As the Moyal distribution results in a very good fit, it will be used to generate a set of random numbers. For comparison, ITS will also be used.

In [42]:
```
MolecularMassWeightedExposedArea_random_moyal = moyal.rvs(f_MMWEA.fitted_param['moyal'][0],
                                                          f_MMWEA.fitted_param['moyal'][1], len(df))

MolecularMassWeightedExposedArea_ITS = inverse_transform_sampling(
                                                    df['Molecular mass weighted exposed area'])

MolecularMassWeightedExposedArea_random_moyal = MolecularMassWeightedExposedArea_random_moyal.astype(
                                    dtype = 'float32', casting = 'same_kind')

MolecularMassWeightedExposedArea_ITS = MolecularMassWeightedExposedArea_ITS.astype(
                                    dtype = 'float32', casting = 'same_kind')
```

In [113]:
```
# plot original and synthetic data

trace_experimental = go.Histogram(x = df['Molecular mass weighted exposed area'], opacity= 0.75,
                                  name = 'experimental',
                                  marker = dict(color = 'yellow'))

trace_synthetic = go.Histogram(x = MolecularMassWeightedExposedArea_ITS, opacity= 0.75,
                                  name = 'ITS',
                                  marker = dict(color = 'magenta'))

trace_random1 = go.Histogram(x = MolecularMassWeightedExposedArea_random_moyal, opacity= 0.5,
                                  name = 'random moyal',
                                  marker = dict(color = 'blue'))

data = [trace_experimental, trace_synthetic, trace_random1]

# https://plot.ly/python/text-and-annotations/

layout = go.Layout(barmode = 'overlay',
                   title = '<b>Figure 9: Comparison of experimental and synthetic <br>' +
                   'Molecular mass weighted exposed area data</b>',
    annotations=[
        dict(
            x=2.4e+6,
            y=300,
            xref='x',
            yref='y',
            text='<i>minor peak not fully captured by <br>random numbers from moyal distribution</i>',
            showarrow=True,
            arrowhead=2,
            arrowwidth = 2,
            arrowsize = 1,
            ax=80,
            ay=-50,
            align = 'center'
        )
    ]
)

fig = go.Figure(data=data, layout=layout)
# iplot(fig)

img_bytes = pio.to_image(fig, format='png')
Image(img_bytes)
```

Out[113]:



Again, we can see that the Moyal distribution slightly 'simplifies' the actual distribution of the data unlike ITS.

Average deviation from standard exposed area of residue

In this section, we will look at the Average deviation from standard exposed area of residue.

In [44]:
```
f_ADSEAR = Fitter(df['Average deviation from standard exposed area of residue'], bins=200)
f_ADSEAR.fit()
```

Fitted alpha distribution with error=4.9258864528964864e-05, AIC=3103.1961254874823, BIC=3127.3082133957587)
Fitted anglit distribution with error=0.0010195392796914062, AIC=2650.806552024893, BIC=2666.8812772970773)
Fitted arcsine distribution with error=0.0015780889595901663, AIC=2594.7932194152822, BIC=2610.8679446874667)
Fitted argus distribution with error=0.0020017738921001755, AIC=2380.9058514641756, BIC=2405.017939372452)
Fitted beta distribution with error=6.0210069388078306e-05, AIC=3282.177674517639, BIC=3314.3271250620073)
Fitted betaprime distribution with error=4.1690764942362445e-05, AIC=3117.2069855140967, BIC=3149.356436058465)
Fitted bradford distribution with error=0.0007138559058521107, AIC=2598.5700711579175, BIC=2622.682159066194)
Fitted burr distribution with error=4.016414242729099e-05, AIC=3048.149255891062, BIC=3080.29870643543)
Fitted burr12 distribution with error=5.716151946237583e-05, AIC=3069.778943428775, BIC=3101.928393973144)
Fitted cauchy distribution with error=0.00018606591136171994, AIC=3103.5820872932995, BIC=3119.656812565484)
Fitted chi distribution with error=0.00013938512729042826, AIC=3644.6107470316133, BIC=3668.7228349398897)
Fitted chi2 distribution with error=0.023602172249471105, AIC=12479.077787573375, BIC=12503.189875481652)
Fitted cosine distribution with error=0.0008497016759741201, AIC=2800.8943529136595, BIC=2816.969078185844)
Fitted crystalball distribution with error=0.0008868259884936988, AIC=3365.6117616176616, BIC=3397.7612121620305)
Fitted dgamma distribution with error=0.00021053747689434564, AIC=3503.3166832513843, BIC=3527.4287711596608)
Fitted dweibull distribution with error=0.0002331474942842281, AIC=3609.8132674845006, BIC=3633.925355392777)
Fitted erlang distribution with error=6.052885195420037e-05, AIC=3268.9207044730274, BIC=3293.032792381304)
Fitted expon distribution with error=0.0006733516425302364, AIC=2892.2462332718633, BIC=2908.3209585440477)

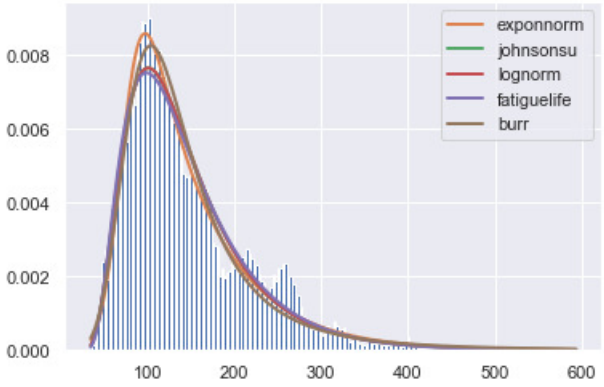
```
E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 2.916666666666667.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.091666666666667.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.1277777777777773.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.1916666666666664.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.060185185185185.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.0444444444444443.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.1919753086419753.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.1532921810699586.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.1799382716049376.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.053292181069957.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.067832647462275.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.0378600823045243.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.120541838134427.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.1054526748971165.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.162208504801095.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 3.0805212620027413.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:
```

Fitted exponnorm distribution with error=2.8134942600201863e-05, AIC=3105.0113841975085, BIC=3129.123472105785)
Fitted exponpow distribution with error=0.0035097219180125192, AIC=3662.4467185788403, BIC=3686.5588064871167)
Fitted exponweib distribution with error=4.221398612573632e-05, AIC=3143.785531952365, BIC=3175.934982496734)
Fitted f distribution with error=4.9556907983629105e-05, AIC=3193.861223173328, BIC=3226.0106737176966)
Fitted fatiguelife distribution with error=4.0074186613674576e-05, AIC=3168.9042711972775, BIC=3193.016359105554)
Fitted fisk distribution with error=4.183489187951495e-05, AIC=3043.316435632145, BIC=3067.4285235404213)
Fitted foldcauchy distribution with error=0.00013330773179325193, AIC=2971.212947621099, BIC=2995.3250355293753)
Fitted foldnorm distribution with error=0.0018159730961242748, AIC=inf, BIC=inf)
Fitted frechet_l distribution with error=0.003502103779061294, AIC=3441.564348923199, BIC=3465.6764368314753)
Fitted frechet_r distribution with error=0.006419815759847138, AIC=3387.051468099568, BIC=3411.1635560078444)
Fitted gamma distribution with error=6.052882414582273e-05, AIC=3268.9205939556336, BIC=3293.032681863916391)
Fitted gausshyper distribution with error=0.00021768073007794907, AIC=3433.257405492758, BIC=3481.481581309311)
Fitted genexpon distribution with error=0.0006585427133090918, AIC=2899.9469456204706, BIC=2940.1337588009314)
Fitted genextreme distribution with error=4.233498748275997e-05, AIC=3105.227000015495, BIC=3129.3390879237713)
Fitted gengamma distribution with error=0.0007967258945672015, AIC=2899.52942793246, BIC=2931.6788784768287)
Fitted genhalflogistic distribution with error=0.0003462696381377131, AIC=3279.6939552409417, BIC=3303.806043149218)
Fitted genlogistic distribution with error=9.27293641859304e-05, AIC=3331.5574783317734, BIC=3355.66956624005)
Fitted gennorm distribution with error=0.0002367883157581558, AIC=3662.6349286511577, BIC=3686.747016559434)
Fitted genpareto distribution with error=0.0006224128997901426, AIC=2868.7727478537663, BIC=2892.8848357620427)
Fitted gilbrat distribution with error=0.0002794678587915201, AIC=2830.964517025739, BIC=2847.0392422979235)
Fitted gompertz distribution with error=0.011338926985595648, AIC=inf, BIC=inf)
Fitted gumbel_l distribution with error=0.000561949295038357, AIC=8124.418749617855, BIC=8140.493474890039)
Fitted gumbel_r distribution with error=9.2884040771587e-05, AIC=3330.305320366911, BIC=3346.3800456390954)
Fitted halfcauchy distribution with error=0.0006006865982148521, AIC=2831.8796056182655, BIC=2847.95433089045)
Fitted halfgennorm distribution with error=0.0019072348583817368, AIC=2872.174099447637, BIC=2896.2861873559136)
Fitted halflogistic distribution with error=0.0004047409568256089, AIC=3005.888014197651, BIC=3021.9627394698355)
Fitted halfnorm distribution with error=0.0003620419174980041, AIC=3247.7650152026968, BIC=3263.839740474881)
Fitted hypsecant distribution with error=0.0002222372725915371, AIC=3484.4554527988275, BIC=3500.530178071012)
Fitted invgamma distribution with error=4.02547111795557e-05, AIC=3107.6574468419085, BIC=3131.769534750185)
Fitted invgauss distribution with error=0.0001009378515811587, AIC=3124.994965110766, BIC=3149.1070530190423)
Fitted invweibull distribution with error=4.233513781678851e-05, AIC=3105.2283660828816, BIC=3129.340453991158)
Fitted johnsonsb distribution with error=4.655044840894839e-05, AIC=3387.6283828327873, BIC=3419.7778333771557)
Fitted johnsonsu distribution with error=3.921614746910063e-05, AIC=3127.787759921315, BIC=3159.9372104650683)
Fitted kappa3 distribution with error=0.0003088850078580583 4, AIC=3142.3776936627582, BIC=3166.4897815710347)
Fitted kappa4 distribution with error=0.0009552865015055711, AIC=3030.2272729724973, BIC=3062.376723516866)
SKIPPED ksone distribution(taking more than 30 seconds)
Fitted kstwobign distribution with error=9.799759366817982e-05, AIC=3570.946157374042, BIC=3587.0208826462263)
Fitted laplace distribution with error=0.00020939856347908798, AIC=3399.769944729435, BIC=3415.8446700016193)
Fitted levy distribution with error=0.0006000602183183067, AIC=2888.9105394332432, BIC=2904.9852647054277)
Fitted levy_l distribution with error=0.001533557545652654, AIC=3227.2480690227308, BIC=3243.3227942949915)
SKIPPED levy_stable distribution(taking more than 30 seconds)
Fitted loggamma distribution with error=0.0003183895979502029, AIC=4359.297515275496, BIC=4383.409603183773)
Fitted logistic distribution with error=0.00025172262923500954, AIC=3588.2040908525055, BIC=3604.278816121 2469)

In [45]:
```
f_ADSEAR.summary()
```

Out[45]:

	sumsquare_error	AIC	BIC
exponnorm	0.000028	3105.011384	3129.123472
johnsonsu	0.000039	3127.787760	3159.937210
lognorm	0.000039	3127.029450	3151.141537
fatiguelife	0.000040	3168.904271	3193.016359
burr	0.000040	3048.149256	3080.298706

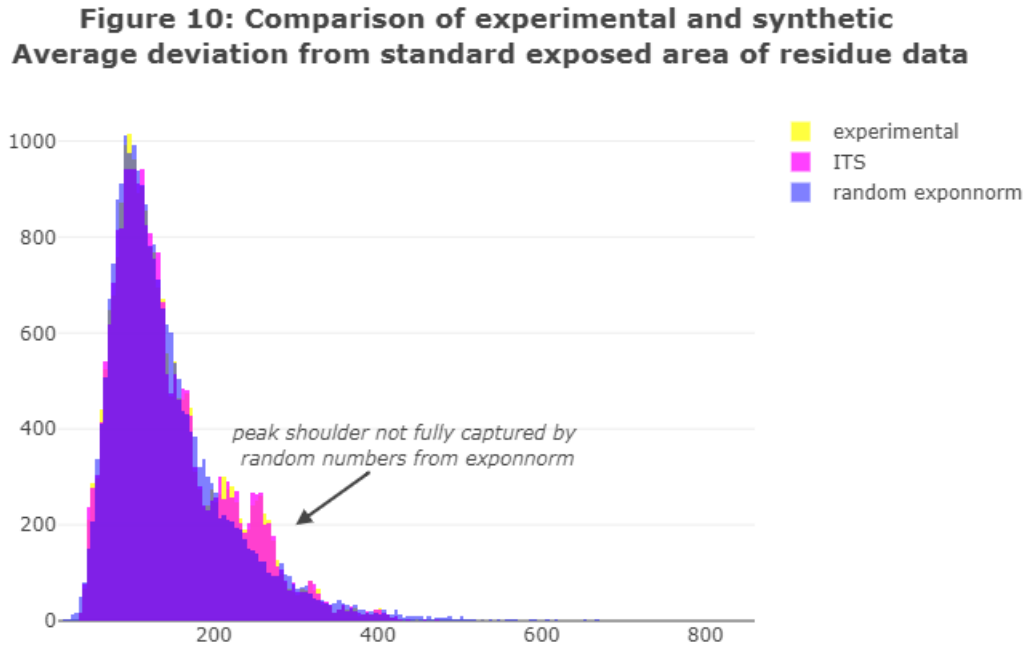


From the Fitter summary above, exponentially-modified normal distribution provides a good description of the data. As such, we can use the automatic fitting parameters to generate random values and compared with ITS and the experimental data.

In [51]:
```
# create synthetic data - inverse_transform_sampling(data, n_bins=200, n_samples=40000)

AverageDeviationFromStandardExposedAreaOfResidue_random_exponnorm = exponnorm.rvs(
                                                                    f_ADSEAR.fitted_param['exponnorm'][
0],
                                                                    f_ADSEAR.fitted_param['exponnorm'][
1],
                                                                    f_ADSEAR.fitted_param['exponnorm'][
2],
                                                                    len(df))

AverageDeviationFromStandardExposedAreaOfResidue_ITS = inverse_transform_sampling(
                                                       df['Average deviation from standard exposed area of
residue'],
                                                       n_samples = len(df))

AverageDeviationFromStandardExposedAreaOfResidue_random_exponnorm = AverageDeviationFromStandardExposed
AreaOfResidue_random_exponnorm.astype(
                                     dtype = 'float32', casting = 'same_kind')

AverageDeviationFromStandardExposedAreaOfResidue_ITS = AverageDeviationFromStandardExposedAreaOfResidue
_ITS.astype(
                                     dtype = 'float32', casting = 'same_kind')
```

In [114]:
```
# plot original and synthetic data

trace_experimental = go.Histogram(x = df['Average deviation from standard exposed area of residue'], o
pacity= 0.75,
                                  name = 'experimental',
                                  marker = dict(color = 'yellow'))

trace_ITS = go.Histogram(x = AverageDeviationFromStandardExposedAreaOfResidue_ITS, opacity= 0.75,
                                  name = 'ITS',
                                  marker = dict(color = 'magenta'))

trace_random1 = go.Histogram(x = AverageDeviationFromStandardExposedAreaOfResidue_random_exponnorm, op
acity= 0.5,
                                  name = 'random exponnorm',
                                  marker = dict(color = 'blue'))

data = [trace_experimental, trace_ITS, trace_random1]

layout = go.Layout(barmode = 'overlay',
                   title = '<b>Figure 10: Comparison of experimental and synthetic <br>' +
                   'Average deviation from standard exposed area of residue data</b>',
    annotations=[
        dict(
            x=300,
            y=200,
            xref='x',
            yref='y',
            text='<i>peak shoulder not fully captured by <br>random numbers from exponnorm</i>',
            showarrow=True,
            arrowhead=2,
            arrowwidth = 2,
            arrowsize = 1,
            ax=70,
            ay=-50,
            align = 'center'
        )
    ]
)

fig = go.Figure(data=data, layout=layout)
# iplot(fig)
img_bytes = pio.to_image(fig, format='png')
Image(img_bytes)
```

Out[114]:



With the exception of a shoulder at around $x = 250$ and extreme values at the higher end of x, both ITS and the exponentially-modified normal distribution capture the full experimental data.

Euclidian distance

Euclidian distance refers to the straight-line distance between two points in Euclidean space (https://en.wikipedia.org/wiki/Euclidean_distance). As protein molecules do not follow a regular space, residue depth cannot be calculated easily. Instead, it can be simplified by several methods including Euclidian distance transform (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4098708/).

In [72]:
```
f_EucD = Fitter(df['Euclidian distance'], bins=200)
f_EucD.fit()
```

```
E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_dist
n_infrastructure.py:2305: RuntimeWarning:

invalid value encountered in double_scalars


Fitted alpha distribution with error=1.2675926580739996e-08, AIC=10211.70149104891, BIC=10235.8135789
57188)
Fitted anglit distribution with error=6.150634743776235e-07, AIC=inf, BIC=inf)
Fitted arcsine distribution with error=6.443535154680564e-07, AIC=4538.425426225633, BIC=4554.5001514
97817)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_dist
n_infrastructure.py:1615: RuntimeWarning:

invalid value encountered in log

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:6532: RuntimeWarning:

divide by zero encountered in double_scalars

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:437: RuntimeWarning:

invalid value encountered in sqrt

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\optimize\mi
npack.py:163: RuntimeWarning:

The iteration is not making good progress, as measured by the
  improvement from the last ten iterations.


Fitted argus distribution with error=7.34998827781212e-07, AIC=4472.483959774107, BIC=4496.5960476823
83)
Fitted beta distribution with error=2.2099791866425782e-08, AIC=25042.083328578297, BIC=25074.2327791
22667)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:589: RuntimeWarning:

divide by zero encountered in true_divide

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:593: RuntimeWarning:

divide by zero encountered in true_divide

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_dist
n_infrastructure.py:1037: RuntimeWarning:

invalid value encountered in subtract


Fitted betaprime distribution with error=1.8305695616647037e-08, AIC=12501.949967421766, BIC=12534.09
9417966134)
Fitted bradford distribution with error=6.133189311005922e-07, AIC=4518.762721325921, BIC=4542.874809
2341975)
Fitted burr distribution with error=2.488087930437583e-08, AIC=7647.47206572402, BIC=7679.62151626838
8)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:768: RuntimeWarning:

overflow encountered in power


Fitted burr12 distribution with error=5.741365254986199e-07, AIC=5230.905591976658, BIC=5263.05504252
1027)
Fitted cauchy distribution with error=5.942680263291941e-08, AIC=5912.4099462796075, BIC=5928.4846715
517915)
Fitted chi distribution with error=7.25554065760061e-07, AIC=inf, BIC=inf)
Fitted chi2 distribution with error=7.255540593855281e-07, AIC=3060717.334101947, BIC=3060741.4461898
55)
Fitted cosine distribution with error=6.170724027323735e-07, AIC=inf, BIC=inf)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_dist
n_infrastructure.py:1615: RuntimeWarning:

divide by zero encountered in log
```

```
Fitted crystalball distribution with error=7.25554065760061e-07, AIC=inf, BIC=inf)
Fitted dgamma distribution with error=2.0402754979030203e-08, AIC=19280.235714832106, BIC=19304.34780
2740383)
Fitted dweibull distribution with error=2.9763195587702735e-08, AIC=18792.31895473224, BIC=18816.4310
42640517)
```

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 0.00025.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 0.00016666666666666
666.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 0.00027777777777777
78.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 0.00046296296296296
3.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 0.00034722222222222
224.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 0.00041666666666666
675.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 0.00054166666666666
69.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 0.00036574074074074
086.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 0.00037500000000000
02.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 0.00050000000000000
02.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 0.00061111111111111
14.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 0.00076851851851851
91.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 0.00102777777777777
85.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 0.00080092592592592
65.

```
Fitted erlang distribution with error=2.3584273208416777e-08, AIC=24465.328338940024, BIC=24489.44042
68483)
Fitted expon distribution with error=4.3124063356245683e-07, AIC=8192.291247915382, BIC=8208.36597318
7565)
Fitted exponnorm distribution with error=7.116860416099453e-09, AIC=16701.48126599565, BIC=16725.5933
5390393)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:1428: RuntimeWarning:

overflow encountered in exp


Fitted exponpow distribution with error=7.25554065760061e-07, AIC=inf, BIC=inf)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:1380: RuntimeWarning:

invalid value encountered in add


Fitted exponweib distribution with error=8.541484953021614e-07, AIC=5206.098616586117, BIC=5238.24806
7130486)
Fitted f distribution with error=1.8378286114299383e-08, AIC=12474.052187198607, BIC=12506.2016377429
75)
Fitted fatiguelife distribution with error=5.789737106777637e-07, AIC=6760.957440252942, BIC=6785.069
528161219)
Fitted fisk distribution with error=7.042833373948027e-09, AIC=9019.137373517737, BIC=9043.2494614260
14)
Fitted foldcauchy distribution with error=5.345831643494148e-08, AIC=5729.037034065801, BIC=5753.1491
21974077)
Fitted foldnorm distribution with error=7.25554065760061e-07, AIC=inf, BIC=inf)
Fitted frechet_l distribution with error=7.489960402064539e-07, AIC=5654.4749342069845, BIC=5678.5870
22115261)
Fitted frechet_r distribution with error=6.934415714024307e-07, AIC=5689.583128120137, BIC=5713.69521
6028414)
Fitted gamma distribution with error=7.25554065760061e-07, AIC=5202871.34001998, BIC=5202895.45210788
9)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:3080: RuntimeWarning:

divide by zero encountered in power


Fitted gausshyper distribution with error=5.079627483445889e-07, AIC=5297.8147032324105, BIC=5346.038
879048963)
Fitted genexpon distribution with error=4.3132772177717065e-07, AIC=8198.330713388095, BIC=8238.51752
6568558)
Fitted genextreme distribution with error=7.019750317373037e-07, AIC=5557.568971609964, BIC=5581.6810
595182405)
Fitted gengamma distribution with error=8.143611552917993e-07, AIC=5358.389482919769, BIC=5390.538933
464138)
Fitted genhalflogistic distribution with error=3.1488006998996966e-07, AIC=11905.77885166568, BIC=119
29.890939573957)
Fitted genlogistic distribution with error=6.372234597048781e-09, AIC=18380.447561034925, BIC=18404.5
59648943203)
Fitted gennorm distribution with error=4.622175592330753e-08, AIC=16151.623022053653, BIC=16175.73510
996193)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2159: RuntimeWarning:

divide by zero encountered in true_divide
```

```
Fitted genpareto distribution with error=3.877830563674023e-07, AIC=10910.04018293495, BIC=10934.1522
70843228)
Fitted gilbrat distribution with error=3.042456661332231e-07, AIC=5733.666320448383, BIC=5749.7410457
20567)
Fitted gompertz distribution with error=7.25554065760061e-07, AIC=inf, BIC=inf)
Fitted gumbel_l distribution with error=5.600551496713761e-07, AIC=79311.78450868909, BIC=79327.85923
396127)
Fitted gumbel_r distribution with error=2.1859106691527695e-08, AIC=15609.029912062064, BIC=15625.104
637334249)
Fitted halfcauchy distribution with error=3.9273529393105166e-07, AIC=5229.181651525421, BIC=5245.256
376797605)
Fitted halfgennorm distribution with error=7.867305126114551e-07, AIC=5126.320043494083, BIC=5150.432
131402359)
Fitted halflogistic distribution with error=3.146536642079409e-07, AIC=10054.658787839271, BIC=10070.
733513111456)
Fitted halfnorm distribution with error=7.25554065760061e-07, AIC=inf, BIC=inf)
Fitted hypsecant distribution with error=1.335464357225575e-08, AIC=19635.91086864976, BIC=19651.9855
93921945)
Fitted invgamma distribution with error=1.503996551450588e-08, AIC=12906.895045286084, BIC=12931.0071
33194362)
Fitted invgauss distribution with error=6.017106521301861e-07, AIC=8334.334029689575, BIC=8358.446117
597852)
Fitted invweibull distribution with error=2.5081477896889113e-08, AIC=12110.447843766025, BIC=12134.5
59931674303)
Fitted johnsonsb distribution with error=1.8030626308738957e-08, AIC=15407.484990068495, BIC=15439.63
4440612863)
Fitted johnsonsu distribution with error=8.957774475462055e-09, AIC=9609.40077638843, BIC=9641.550226
932797)
Fitted kappa3 distribution with error=1.9242550492780578e-07, AIC=11274.7014954548, BIC=11298.8135833
63077)
Fitted kappa4 distribution with error=4.90458299091727e-07, AIC=5334.492647047641, BIC=5366.642097592
01)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\integrate\q
uadpack.py:385: IntegrationWarning:

The maximum number of subdivisions (50) has been achieved.
  If increasing the limit yields no improvement it is advised to analyze
  the integrand in order to determine the difficulties.  If the position of a
  local difficulty can be determined (singularity, discontinuity) one will
  probably gain from splitting up the interval and calling the integrator
  on the subranges.  Perhaps a special-purpose integrator should be used.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:44: RuntimeWarning:

floating point number truncated to an integer


SKIPPED ksone distribution(taking more than 30 seconds)
Fitted kstwobign distribution with error=9.235125156506661e-08, AIC=nan, BIC=nan)
Fitted laplace distribution with error=3.987423998874584e-08, AIC=17354.87895226437, BIC=17370.953677
536556)
Fitted levy distribution with error=4.1626069173774194e-07, AIC=4992.98707355397, BIC=5009.0617988261
54)
Fitted levy_l distribution with error=7.111937781583881e-07, AIC=5299.884341967803, BIC=5315.95906723
9987)
SKIPPED levy_stable distribution(taking more than 30 seconds)
Fitted loggamma distribution with error=2.011756198045891e-07, AIC=86284.70250058128, BIC=86308.81458
848955)
Fitted logistic distribution with error=1.0372400031460902e-08, AIC=22597.232557604584, BIC=22613.307
28287677)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:3785: RuntimeWarning:

divide by zero encountered in true_divide


Fitted loglaplace distribution with error=4.281024444992251e-08, AIC=7280.720443481829, BIC=7304.8325
31390105)
Fitted lognorm distribution with error=7.921788424021688e-07, AIC=5394.039455207281, BIC=5418.1515431
15558)
Fitted lomax distribution with error=3.831576107782904e-07, AIC=7429.93030680248, BIC=7454.0423947107
565)
Fitted maxwell distribution with error=5.2991000963507906e-08, AIC=77035.39920793312, BIC=77051.47393
32053)
```

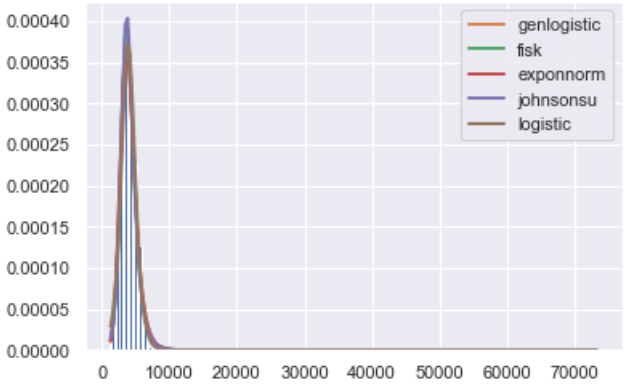
```
E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\integrate\q
uadpack.py:385: IntegrationWarning:

Extremely bad integrand behavior occurs at some points of the
  integration interval.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\integrate\q
uadpack.py:385: IntegrationWarning:

The integral is probably divergent, or slowly convergent.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:4064: RuntimeWarning:

overflow encountered in power

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:4064: RuntimeWarning:

overflow encountered in multiply

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:4064: RuntimeWarning:

invalid value encountered in true_divide


Fitted mielke distribution with error=3.6536638371099533e-07, AIC=5127.721096860297, BIC=5159.8705474
04666)
Fitted moyal distribution with error=4.88620259340702e-08, AIC=12494.211269695988, BIC=12510.28599496
8173)
Fitted nakagami distribution with error=6.618969151352352e-08, AIC=68442.86669982973, BIC=68466.97878
7738)
SKIPPED ncf distribution(taking more than 30 seconds)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:4768: RuntimeWarning:

invalid value encountered in multiply

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:4764: RuntimeWarning:

overflow encountered in multiply


SKIPPED nct distribution(taking more than 30 seconds)
Fitted ncx2 distribution with error=7.255540525963911e-07, AIC=2125363.112187224, BIC=2125395.2616377
682)
Fitted norm distribution with error=9.784590030678773e-08, AIC=107901.46797474971, BIC=107917.5427000
2189)
Fitted norminvgauss distribution with error=2.5934357302494537e-08, AIC=18000.410698128348, BIC=18032
.560148672717)
Fitted pareto distribution with error=8.053868437674458e-07, AIC=5254.2693310197665, BIC=5278.3814189
28043)
Fitted pearson3 distribution with error=2.358427721351268e-08, AIC=24465.327340824602, BIC=24489.4394
2873288)
Fitted powerlaw distribution with error=6.02009630371879e-07, AIC=4695.1820097832, BIC=4719.294097691
476)
Fitted powerlognorm distribution with error=9.885466555569772e-07, AIC=5223.1124834137445, BIC=5255.2
61933958113)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5203: RuntimeWarning:

divide by zero encountered in power

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5203: RuntimeWarning:

invalid value encountered in multiply
```

```
Fitted powernorm distribution with error=nan, AIC=67512.84625605479, BIC=67536.95834396307)
Fitted rayleigh distribution with error=7.87925730064084e-08, AIC=63727.61723807686, BIC=63743.691963
34905)
Fitted rdist distribution with error=1.1552166213699207e-07, AIC=97708.17153978314, BIC=97732.2836276
9142)
Fitted recipinvgauss distribution with error=6.94100460612941e-07, AIC=6507.352441631799, BIC=6531.46
4529540075)
Fitted reciprocal distribution with error=7.25554065760061e-07, AIC=inf, BIC=inf)
Fitted rice distribution with error=7.879257121762574e-08, AIC=63729.61893058149, BIC=63753.731018489
765)
SKIPPED rv_continuous distribution(taking more than 30 seconds)
SKIPPED rv_histogram distribution(taking more than 30 seconds)
Fitted semicircular distribution with error=6.363572025557082e-07, AIC=inf, BIC=inf)
Fitted skewnorm distribution with error=4.734583529041351e-08, AIC=61467.387486384236, BIC=61491.4995
7429251)
Fitted t distribution with error=1.3568286278510766e-08, AIC=8706.92541362509, BIC=8731.037501533367)
Fitted trapz distribution with error=7.432891160347741e-07, AIC=4614.3431083459345, BIC=4646.49255889
0303)
Fitted triang distribution with error=6.30502128759377e-07, AIC=4594.498258536021, BIC=4618.610346444
298)
Fitted truncexpon distribution with error=6.459932724987945e-07, AIC=4570.590490148341, BIC=4594.7025
78056617)
Fitted truncnorm distribution with error=7.25554065760061e-07, AIC=inf, BIC=inf)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5823: RuntimeWarning:

divide by zero encountered in log

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5823: RuntimeWarning:

invalid value encountered in log

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5883: RuntimeWarning:

divide by zero encountered in power


SKIPPED tukeylambda distribution(taking more than 30 seconds)
Fitted uniform distribution with error=6.873694234630106e-07, AIC=4479.829960565585, BIC=4495.9046858
37769)
Fitted vonmises distribution with error=6.872244930462576e+51, AIC=-22484.79502526099, BIC=-22460.682
937352714)
Fitted vonmises_line distribution with error=7.919557641431161e-08, AIC=78555.07376240045, BIC=78579.
18585030873)
Fitted wald distribution with error=2.670856834191378e-07, AIC=6358.690652957792, BIC=6374.7653782299
76)
Fitted weibull_max distribution with error=7.489960402064539e-07, AIC=5654.4749342069845, BIC=5678.58
7022115261)
Fitted weibull_min distribution with error=6.934415714024307e-07, AIC=5689.583128120137, BIC=5713.695
216028414)
Fitted wrapcauchy distribution with error=nan, AIC=nan, BIC=nan)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\optimize\op
timize.py:563: RuntimeWarning:

invalid value encountered in subtract

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\numpy\core\fromnu
meric.py:83: RuntimeWarning:

invalid value encountered in reduce
```

In [73]:
```
f_EucD.summary()
```

Out[73]:

	sumsquare_error	AIC	BIC
genlogistic	6.372235e-09	18380.447561	18404.559649
fisk	7.042833e-09	9019.137374	9043.249461
exponnorm	7.116860e-09	16701.481266	16725.593354
johnsonsu	8.957774e-09	9609.400776	9641.550227
logistic	1.037240e-08	22597.232558	22613.307283

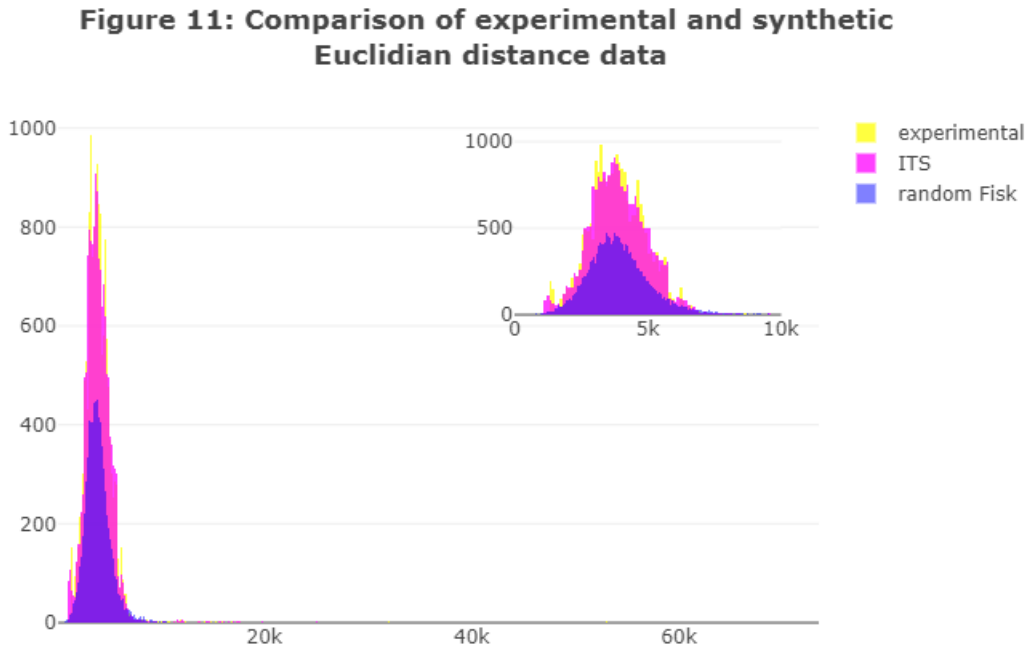


As there are several distributions with good fits to the experimental data, we will use the Fisk distribution as it has the lowest AIC/BIC with only a marginally higher error compared to the other distributions.

In [79]:
```
# create synthetic data

EuclidianDistance_random_fisk = fisk.rvs(f_EucD.fitted_param['fisk'][0],
                                         f_EucD.fitted_param['fisk'][1], f_EucD.fitted_param['fisk'][2],
                                                 len(df))

EuclidianDistance_ITS = inverse_transform_sampling(df['Euclidian distance'], n_samples=len(df))

EuclidianDistance_random_fisk = EuclidianDistance_random_fisk.astype(
                                         dtype = 'float32', casting = 'same_kind')

EuclidianDistance_ITS = EuclidianDistance_ITS.astype(
                                         dtype = 'float32', casting = 'same_kind')
```

In [115]:
```
# plot original and synthetic data
# for inset: https://plot.ly/python/insets/
# we use showlegend=False for the inset traces to hide them from the legend, otherwise we'll be seeing
double entries
# https://plot.ly/python/legend/#hide-legend-entries

trace_experimental = go.Histogram(x = df['Euclidian distance'], opacity= 0.75,
                                  name = 'experimental',
                                  marker = dict(color = 'yellow'))

trace_ITS = go.Histogram(x = EuclidianDistance_ITS, opacity= 0.75,
                                  name = 'ITS',
                                  marker = dict(color = 'magenta'))

trace_random1 = go.Histogram(x = EuclidianDistance_random_fisk, opacity= 0.5,
                                  name = 'random Fisk',
                                  marker = dict(color = 'blue'))

trace_experimental_inset = go.Histogram(x = df['Euclidian distance'], opacity= 0.75,
                                  name = 'experimental', showlegend = False,
                                  marker = dict(color = 'yellow'), xaxis = 'x2', yaxis='y2')

trace_ITS_inset = go.Histogram(x = EuclidianDistance_ITS, opacity= 0.75,
                                  name = 'synthetic by ITS', showlegend = False,
                                  marker = dict(color = 'magenta'), xaxis = 'x2', yaxis='y2')

trace_random1_inset = go.Histogram(x = EuclidianDistance_random_fisk, opacity= 0.5,
                                  name = 'random Fisk', showlegend = False,
                                  marker = dict(color = 'blue'), xaxis = 'x2', yaxis='y2')

data = [trace_experimental, trace_ITS, trace_random1,
        trace_experimental_inset, trace_ITS_inset, trace_random1_inset]

# https://plot.ly/python/text-and-annotations/

layout = go.Layout(barmode = 'overlay',
                   title = '<b>Figure 11: Comparison of experimental and synthetic <br>' +
                   'Euclidian distance data</b>',

    xaxis2 = dict(domain = [0.6, 0.95], anchor = 'y2', range = [0,10000]),
    yaxis2 = dict(domain = [0.6, 0.95], anchor = 'x2')
)

fig = go.Figure(data=data, layout=layout)
# iplot(fig)
img_bytes = pio.to_image(fig, format='png')
Image(img_bytes)
```

Out[115]:



For the plot above, an inset plot focused around the mode of the distribution was created to provide a better visual for comparative purposes, as there are extreme x values with very low counts. This is also apparent from the box plots (Figure 1) where the number of outliers are much higher than that for the other variables. Perhaps due to this, the peak from the Fisk distribution does not entirely reach the same height (y value) as the experimental and ITS traces.

Secondary structure penalty

We will now look at the secondary structure penalty, a measure often used in protein sequence alignment and structure prediction (https://www.ncbi.nlm.nih.gov/pubmed/25524475).

In [81]:
```
f_SSP = Fitter(df['Secondary structure penalty'], bins=200)
f_SSP.fit()
```

```
E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_dist
n_infrastructure.py:2305: RuntimeWarning:

invalid value encountered in double_scalars


Fitted alpha distribution with error=0.04464622997672264, AIC=3560.934463191752, BIC=3585.04655110002
84)
Fitted anglit distribution with error=0.002543502458486693, AIC=2467.560109778178, BIC=2483.634835050
362)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:312: RuntimeWarning:

divide by zero encountered in true_divide


Fitted arcsine distribution with error=0.003588788140973162, AIC=2405.7592825693164, BIC=2421.8340078
41501)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:6532: RuntimeWarning:

divide by zero encountered in double_scalars

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:437: RuntimeWarning:

invalid value encountered in sqrt


Fitted argus distribution with error=0.003844236171999918, AIC=2290.667732544917, BIC=2314.7798204531
93)
Fitted beta distribution with error=0.0004515588074433137, AIC=2874.0722284513904, BIC=2906.221678995
7592)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:589: RuntimeWarning:

divide by zero encountered in true_divide

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:593: RuntimeWarning:

divide by zero encountered in true_divide

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_dist
n_infrastructure.py:1037: RuntimeWarning:

invalid value encountered in subtract


Fitted betaprime distribution with error=0.0003187904697997799, AIC=2770.9360273046523, BIC=2803.0854
77849021)
Fitted bradford distribution with error=0.001493763384540603, AIC=2445.505065013162, BIC=2469.6171529
214384)
Fitted burr distribution with error=0.00032775430062092256, AIC=2794.5548264567174, BIC=2826.70427700
10858)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:768: RuntimeWarning:

overflow encountered in power


Fitted burr12 distribution with error=0.0010391440287315073, AIC=2713.585323892028, BIC=2745.73477443
6397)
Fitted cauchy distribution with error=0.0007466123769649962, AIC=2903.6597239770654, BIC=2919.7344492
4925)
Fitted chi distribution with error=0.0006173924880447689, AIC=2941.3609935727527, BIC=2965.4730814810
29)
Fitted chi2 distribution with error=0.0003547772890838453, AIC=2835.21239288759, BIC=2859.32448079586
63)
Fitted cosine distribution with error=0.00216559256718824, AIC=2619.4959594339252, BIC=2635.570684706
1097)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_dist
n_infrastructure.py:1615: RuntimeWarning:

divide by zero encountered in log
```

```
Fitted crystalball distribution with error=0.0011649360844200106, AIC=3311.244714925716, BIC=3343.394
1654700848)
Fitted dgamma distribution with error=0.0007232217615525091, AIC=3091.0708732786297, BIC=3115.1829611
86906)
Fitted dweibull distribution with error=0.0007679071711770963, AIC=3074.5356017372383, BIC=3098.64768
96455147)
```

```
E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 1.05.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 1.0333333333333332.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 1.0499999999999998.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 1.0666666666666664.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 1.0999999999999996.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 1.0499999999999996.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 1.0499999999999994.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 1.1333333333333324.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 1.1999999999999984.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 1.1833333333333318.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 1.2499999999999978.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 1.2333333333333307.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 1.2999999999999963.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 1.4499999999999957.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 1.6499999999999941.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 1.4666666666666615.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2562: RuntimeWarning:
```

```
Fitted erlang distribution with error=0.06039922336602512, AIC=34063.45243501333, BIC=34087.564522921
6)
Fitted expon distribution with error=0.001089007226461183, AIC=2702.9124138219986, BIC=2718.987139094
183)
Fitted exponnorm distribution with error=0.00033561755344236043, AIC=2782.1559657308226, BIC=2806.268
053639099)
Fitted exponpow distribution with error=0.0009122195115796587, AIC=2909.383554868536, BIC=2933.495642
7768125)
Fitted exponweib distribution with error=0.0003067532758023914, AIC=2785.527917750619, BIC=2817.67736
8294988)
Fitted f distribution with error=0.00030944761635652996, AIC=2799.779482701526, BIC=2831.928933245894
8)
Fitted fatiguelife distribution with error=0.0003089797140472678, AIC=2791.602244717881, BIC=2815.714
3326261576)
Fitted fisk distribution with error=0.02565279485399956, AIC=3113.24729854824, BIC=3137.3593864565164
)
Fitted foldcauchy distribution with error=0.10102994529409544, AIC=3850.4163278878023, BIC=3874.52841
57960787)
Fitted foldnorm distribution with error=0.0006894716309899854, AIC=2902.301882513297, BIC=2926.413970
4215736)
Fitted frechet_l distribution with error=0.009679985820796003, AIC=3214.2597842423797, BIC=3238.37187
2150656)
Fitted frechet_r distribution with error=0.013293343927612524, AIC=3222.4674501710306, BIC=3246.57953
8079307)
Fitted gamma distribution with error=0.0003593638409667925, AIC=2831.6560381443055, BIC=2855.76812605
2582)
Fitted gausshyper distribution with error=0.0005051605133199809, AIC=2890.7379381636442, BIC=2938.962
113980197)
Fitted genexpon distribution with error=0.0009484689792029332, AIC=2714.072516512136, BIC=2754.259329
692597)
Fitted genextreme distribution with error=0.00032659923545055495, AIC=2791.922476753596, BIC=2816.034
5646618725)
Fitted gengamma distribution with error=0.0003110541418048203, AIC=2797.6323777184125, BIC=2829.78182
82627813)
Fitted genhalflogistic distribution with error=0.0006547226486179823, AIC=2819.6331579362723, BIC=284
3.7452458445487)
Fitted genlogistic distribution with error=0.0005261977259231485, AIC=3000.717129532216, BIC=3024.829
2174404924)
Fitted gennorm distribution with error=0.0007731151625180572, AIC=3038.0531293345502, BIC=3062.165217
2428267)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:2159: RuntimeWarning:

divide by zero encountered in true_divide


Fitted genpareto distribution with error=0.052129395269450575, AIC=3365.0833380525555, BIC=3389.19542
5960832)
Fitted gilbrat distribution with error=0.0004931621882508216, AIC=2677.8200547662664, BIC=2693.894780
038451)
Fitted gompertz distribution with error=0.0008213754563945307, AIC=2788.300009241969, BIC=2812.412097
1502452)
Fitted gumbel_l distribution with error=0.001954402045641952, AIC=3496.048398853752, BIC=3512.1231241
259366)
Fitted gumbel_r distribution with error=0.0005263413403342083, AIC=2997.5429471503926, BIC=3013.61767
2422577)
Fitted halfcauchy distribution with error=0.0010617093933861807, AIC=2665.3861578090577, BIC=2681.460
883081242)
Fitted halfgennorm distribution with error=0.004191765081191715, AIC=2692.599247930728, BIC=2716.7113
358390043)
Fitted halflogistic distribution with error=0.0006728583147994729, AIC=2794.0382363733033, BIC=2810.1
129616454878)
Fitted halfnorm distribution with error=0.0006894717338142359, AIC=2900.301590283338, BIC=2916.376315
5555225)
Fitted hypsecant distribution with error=0.0007803163930561001, AIC=3115.8291059191897, BIC=3131.9038
31191374)
Fitted invgamma distribution with error=0.0003170930697119594, AIC=2790.2903174196454, BIC=2814.40240
53279218)
Fitted invgauss distribution with error=0.00030652873102415926, AIC=2787.56219850502, BIC=2811.674286
4132966)
Fitted invweibull distribution with error=0.00032659909167669863, AIC=2791.9225393459237, BIC=2816.03
46272542)
Fitted johnsonsb distribution with error=0.0003118582023327396, AIC=2793.6136777697834, BIC=2825.7631
28314152)
Fitted johnsonsu distribution with error=0.00030649571967220133, AIC=2788.241584515671, BIC=2820.3910
3506004)
Fitted kappa3 distribution with error=0.0006001906781693453, AIC=2800.876162427865, BIC=2824.98825033
61414)
Fitted kappa4 distribution with error=0.0020313551623986235, AIC=2599.9430915872977, BIC=2632.0925421
31666)
```

```
E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\integrate\q
uadpack.py:385: IntegrationWarning:

The maximum number of subdivisions (50) has been achieved.
  If increasing the limit yields no improvement it is advised to analyze
  the integrand in order to determine the difficulties.  If the position of a
  local difficulty can be determined (singularity, discontinuity) one will
  probably gain from splitting up the interval and calling the integrator
  on the subranges.  Perhaps a special-purpose integrator should be used.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:44: RuntimeWarning:

floating point number truncated to an integer


SKIPPED ksone distribution(taking more than 30 seconds)
Fitted kstwobign distribution with error=0.0005982540884769699, AIC=3060.2795168447774, BIC=3076.3542
42116962)
Fitted laplace distribution with error=0.0007720291207910318, AIC=3053.9864085376926, BIC=3070.061133
809877)
Fitted levy distribution with error=0.0012669013139914806, AIC=2702.052216014364, BIC=2718.1269412865
486)
Fitted levy_l distribution with error=0.00380346698287466, AIC=2985.126960266846, BIC=3001.2016855390
3)
SKIPPED levy_stable distribution(taking more than 30 seconds)
Fitted loggamma distribution with error=0.0011970795874855006, AIC=3254.624847452023, BIC=3278.736935
3602994)
Fitted logistic distribution with error=0.0008704549374093911, AIC=3170.6673498296395, BIC=3186.74207
5101824)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:3785: RuntimeWarning:

divide by zero encountered in true_divide


Fitted loglaplace distribution with error=0.024106257529894034, AIC=3198.1664487920157, BIC=3222.2785
36700292)
Fitted lognorm distribution with error=0.0003064786300392661, AIC=2786.382873013691, BIC=2810.4949609
219675)
Fitted lomax distribution with error=0.013579480622170967, AIC=2938.126034176381, BIC=2962.2381220846
573)
Fitted maxwell distribution with error=0.0009044007219934781, AIC=3199.203015620209, BIC=3215.2777408
923935)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\integrate\q
uadpack.py:385: IntegrationWarning:

Extremely bad integrand behavior occurs at some points of the
  integration interval.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\integrate\q
uadpack.py:385: IntegrationWarning:

The integral is probably divergent, or slowly convergent.


Fitted mielke distribution with error=0.00032639338461363073, AIC=2794.801850439516, BIC=2826.9513009
838847)
Fitted moyal distribution with error=0.00039685325323790316, AIC=2908.8978408694866, BIC=2924.9725661
41671)
Fitted nakagami distribution with error=0.000617392574062157, AIC=2941.360985326866, BIC=2965.4730732
351422)
SKIPPED ncf distribution(taking more than 30 seconds)
SKIPPED nct distribution(taking more than 30 seconds)
SKIPPED ncx2 distribution(taking more than 30 seconds)
Fitted norm distribution with error=0.0011649361922328307, AIC=3307.2446219939015, BIC=3323.319347266
086)
Fitted norminvgauss distribution with error=0.0003170424288959225, AIC=2783.029588812273, BIC=2815.17
90393566416)
Fitted pareto distribution with error=0.0017573992518241316, AIC=2835.910840676862, BIC=2860.02292858
51383)
Fitted pearson3 distribution with error=0.0003547777110635062, AIC=2835.2113865286647, BIC=2859.32347
4436941)
Fitted powerlaw distribution with error=0.002604548960582904, AIC=2422.2759303260177, BIC=2446.388018
234294)
```

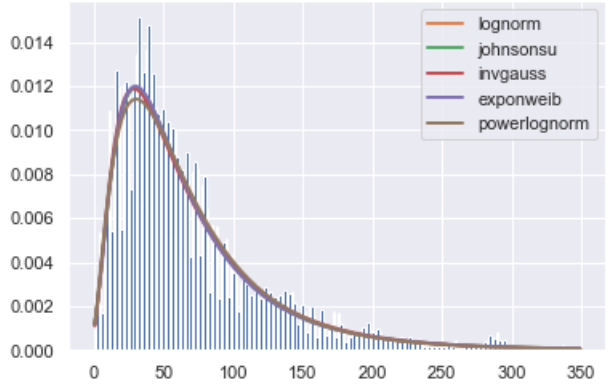
```
E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5166: RuntimeWarning:

divide by zero encountered in power

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5166: RuntimeWarning:

invalid value encountered in multiply


Fitted powerlognorm distribution with error=0.00030896264231406015, AIC=2801.7634712596696, BIC=2833.
912921804038)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5203: RuntimeWarning:

divide by zero encountered in power

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5203: RuntimeWarning:

invalid value encountered in multiply

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5250: RuntimeWarning:

divide by zero encountered in power


Fitted powernorm distribution with error=nan, AIC=3162.564863603529, BIC=3186.6769515118053)
Fitted rayleigh distribution with error=0.0008230363822755095, AIC=3160.469951451144, BIC=3176.544676
7233284)
Fitted rdist distribution with error=0.0011678409324094477, AIC=3305.9929532178426, BIC=3330.10504112
6119)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5477: RuntimeWarning:

divide by zero encountered in true_divide

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5477: RuntimeWarning:

divide by zero encountered in log

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5477: RuntimeWarning:

invalid value encountered in subtract


Fitted recipinvgauss distribution with error=0.005043686845725296, AIC=2697.5813341490207, BIC=2721.6
93422057297)
Fitted reciprocal distribution with error=1.7608232705151825, AIC=inf, BIC=inf)
Fitted rice distribution with error=0.0008230362928581454, AIC=3162.4691701395786, BIC=3186.581258047
855)
SKIPPED rv_continuous distribution(taking more than 30 seconds)
SKIPPED rv_histogram distribution(taking more than 30 seconds)
Fitted semicircular distribution with error=0.002839368956587377, AIC=2392.220895248703, BIC=2408.295
6205208875)
Fitted skewnorm distribution with error=0.0004762048297258544, AIC=2948.075788420819, BIC=2972.187876
3290956)
Fitted t distribution with error=0.000699542124729689, AIC=3014.8987347299726, BIC=3039.010822638249)
Fitted trapz distribution with error=0.004536812679579436, AIC=2480.977019281335, BIC=2513.1264698257
037)
Fitted triang distribution with error=0.0014176202316272864, AIC=2469.89058632181, BIC=2494.002674230
0865)
Fitted truncexpon distribution with error=0.0017716101618681368, AIC=2377.3013289998953, BIC=2401.413
4169081717)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5823: RuntimeWarning:

divide by zero encountered in log

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5823: RuntimeWarning:

invalid value encountered in log


Fitted truncnorm distribution with error=0.9262756882385936, AIC=inf, BIC=inf)
```

```
E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5883: RuntimeWarning:

divide by zero encountered in power

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_dist
n_infrastructure.py:1615: RuntimeWarning:

invalid value encountered in log


SKIPPED tukeylambda distribution(taking more than 30 seconds)
Fitted uniform distribution with error=0.002874455893790633, AIC=2347.173261793384, BIC=2363.24798706
55684)
Fitted vonmises distribution with error=4.387418788830498e+56, AIC=-24928.58259503805, BIC=-24904.470
507129772)
Fitted vonmises_line distribution with error=0.0009919524390159828, AIC=3100.243472200842, BIC=3124.3
555601091184)
Fitted wald distribution with error=0.0004384042483449736, AIC=2689.7901502571094, BIC=2705.864875529
294)
Fitted weibull_max distribution with error=0.009679985820796003, AIC=3214.2597842423797, BIC=3238.371
872150656)
Fitted weibull_min distribution with error=0.013293343927612524, AIC=3222.4674501710306, BIC=3246.579
538079307)
Fitted wrapcauchy distribution with error=nan, AIC=nan, BIC=nan)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\optimize\op
timize.py:563: RuntimeWarning:

invalid value encountered in subtract

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\numpy\core\fromnu
meric.py:83: RuntimeWarning:

invalid value encountered in reduce
```

In [82]:
```
f_SSP.summary()
```

Out[82]:

	sumsquare_error	AIC	BIC
lognorm	0.000306	2786.382873	2810.494961
johnsonsu	0.000306	2788.241585	2820.391035
invgauss	0.000307	2787.562199	2811.674286
exponweib	0.000307	2785.527918	2817.677368
powerlognorm	0.000309	2801.763471	2833.912922

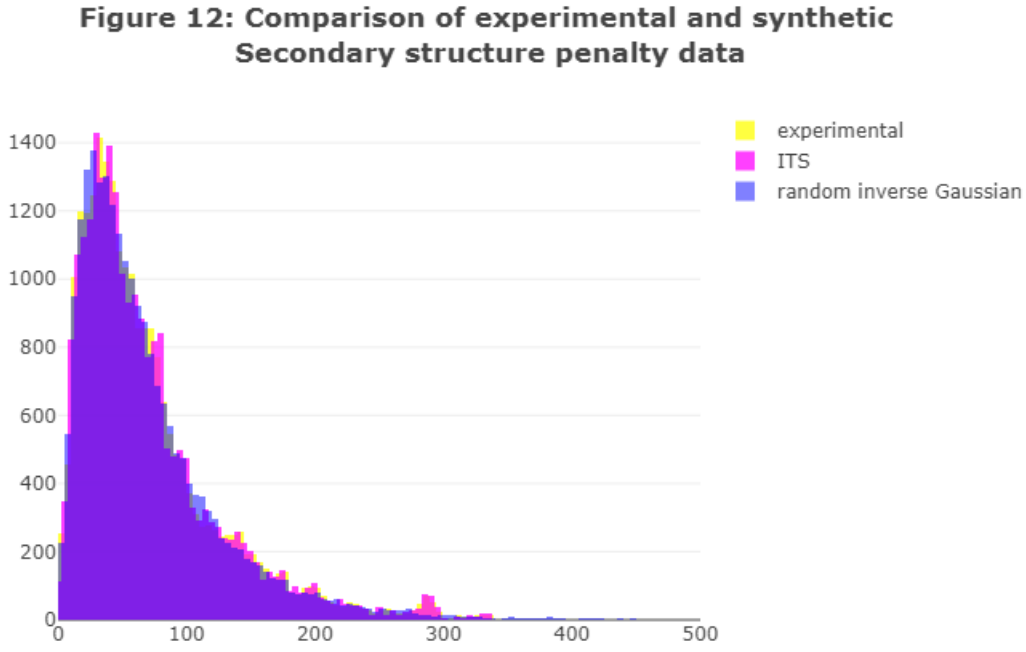


For this case, several distributions have very similar error, AIC and BIC values. Visually, they all seem to overlap with each other and it is not entirely clear if any one distribution has an advantage over the others. Therefore, by random choice, the inverse Gaussian (invgauss) will be used.

In [83]:
```
# create synthetic data

SecondaryStructurePenalty_random_invgauss = invgauss.rvs(f_SSP.fitted_param['invgauss'][0],
                                                         f_SSP.fitted_param['invgauss'][1]
,
                                                         f_SSP.fitted_param['invgauss'][2]
, len(df))

SecondaryStructurePenalty_ITS = inverse_transform_sampling(df['Secondary structure penalty'])

SecondaryStructurePenalty_random_invgauss = SecondaryStructurePenalty_random_invgauss.astype(
                                    dtype = 'float32', casting = 'same_kind')

SecondaryStructurePenalty_ITS = SecondaryStructurePenalty_ITS.astype(
                                    dtype = 'float32', casting = 'same_kind')
```

In [116]:
```
trace_experimental = go.Histogram(x = df['Secondary structure penalty'], opacity= 0.75,
                                  name = 'experimental',
                                  marker = dict(color = 'yellow'))

trace_ITS = go.Histogram(x = SecondaryStructurePenalty_ITS, opacity= 0.75,
                                  name = 'ITS',
                                  marker = dict(color = 'magenta'))

trace_random1 = go.Histogram(x = SecondaryStructurePenalty_random_invgauss, opacity= 0.5,
                                  name = 'random inverse Gaussian',
                                  marker = dict(color = 'blue'))

data = [trace_experimental, trace_ITS, trace_random1]

# the maximum x value in the experimental data is 350 but invgauss.rvs returned data up to about 600
# therefore, we will rescale the plot on xaxis
layout = go.Layout(barmode = 'overlay',
                   title = '<b>Figure 12: Comparison of experimental and synthetic <br>' +
                   'Secondary structure penalty data</b>',
                   xaxis=dict(range=[0, 500]))

fig = go.Figure(data=data, layout=layout)
# iplot(fig)
img_bytes = pio.to_image(fig, format='png')
Image(img_bytes)
```

Out[116]:



For this variable, it seems the inverse Gaussian (chosen randomly) is almost as good as ITS in generating random values matching the experimental distribution. This could be caused by the absence of outliers (see Figure 1).

Spatial distribution constraints

This variable is another penalty-type value used in enhancing the prediction accuracy of protein structures, although it could lead to local energy minima and poor sampling efficiency in structure simulations (https://www.ncbi.nlm.nih.gov/pubmed/16267688).

In [85]:
```
f_SDCons = Fitter(df['Spatial Distribution constraints (N,K Value)'], bins=200)
f_SDCons.fit()
```

```
E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_dist
n_infrastructure.py:2305: RuntimeWarning:

invalid value encountered in double_scalars


Fitted alpha distribution with error=0.0355602015917608, AIC=1653.480936848545, BIC=1677.593024756821
5)
Fitted anglit distribution with error=0.06221774573021478, AIC=1463.761377382549, BIC=1479.8361026547
332)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:312: RuntimeWarning:

divide by zero encountered in true_divide


Fitted arcsine distribution with error=0.23568917954697507, AIC=1501.8600178462173, BIC=1517.93474311
84015)
Fitted argus distribution with error=0.047696071054057286, AIC=1476.619958018383, BIC=1500.7320459266
593)
Fitted beta distribution with error=0.018884884569063086, AIC=1599.8353719019328, BIC=1631.9848224463
015)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:589: RuntimeWarning:

divide by zero encountered in true_divide

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:593: RuntimeWarning:

divide by zero encountered in true_divide

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_dist
n_infrastructure.py:1037: RuntimeWarning:

invalid value encountered in subtract


Fitted betaprime distribution with error=0.030245504626072242, AIC=1627.1512929824694, BIC=1659.30074
3526838)
Fitted bradford distribution with error=0.162581554154079, AIC=1402.238598393286, BIC=1426.3506863015
625)
Fitted burr distribution with error=0.016290866603852996, AIC=1584.9989431623662, BIC=1617.1483937067
349)
Fitted burr12 distribution with error=0.01836187614014651, AIC=1604.7099016226566, BIC=1636.859352167
0252)
Fitted cauchy distribution with error=0.03747268645974972, AIC=1662.9632751846552, BIC=1679.038000456
8397)
Fitted chi distribution with error=0.02926198582216958, AIC=1630.1391301625438, BIC=1654.251218070820
2)
Fitted chi2 distribution with error=0.032795321625667215, AIC=1652.6823598251929, BIC=1676.7944477334
693)
Fitted cosine distribution with error=0.03986222997547587, AIC=1578.0004803191928, BIC=1594.075205591
3772)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_dist
n_infrastructure.py:1615: RuntimeWarning:

divide by zero encountered in log


Fitted crystalball distribution with error=0.02785895152250028, AIC=1632.070269258002, BIC=1664.21971
98023707)
Fitted dgamma distribution with error=0.026561963494166813, AIC=1632.2896867038294, BIC=1656.40177461
21058)
Fitted dweibull distribution with error=0.0337794255891393, AIC=1620.6187370907478, BIC=1644.73082499
90242)
```

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 17.85.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 16.15.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 15.299999999999997.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 16.43333333333333.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 16.574999999999996.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 16.362499999999997.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 16.433333333333337.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 16.36250000000001.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 16.645833333333343.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 16.575000000000006.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 15.725000000000009.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 16.291666666666686.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 15.677777777777795.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 15.229166666666693.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 15.25277777777779.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

The shape parameter of the erlang distribution has been given a non-integer value 15.504629629629651.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats_cont
inuous_distns.py:2562: RuntimeWarning:

```
Fitted erlang distribution with error=0.031553323886738385, AIC=1641.0982772199827, BIC=1665.2103651282591)
Fitted expon distribution with error=0.22283606982596788, AIC=1521.6397167099185, BIC=1537.714441982103)
Fitted exponnorm distribution with error=0.027860509108706473, AIC=1630.0772545145824, BIC=1654.1893424228588)
Fitted exponpow distribution with error=0.017346777625276703, AIC=1605.7069553955148, BIC=1629.8190433037912)
Fitted exponweib distribution with error=0.04943620150078362, AIC=1710.0584417602272, BIC=1742.2078923045958)
Fitted f distribution with error=0.02820972791601973, AIC=1632.8180215614684, BIC=1664.967472105837)
Fitted fatiguelife distribution with error=0.028141334025126773, AIC=1632.9289532584526, BIC=1657.041041166729)
Fitted fisk distribution with error=0.026670627365053858, AIC=1620.0922088605987, BIC=1644.204296768875)
Fitted foldcauchy distribution with error=0.0342721468255375, AIC=1608.4963997234554, BIC=1632.6084876317318)
Fitted foldnorm distribution with error=0.027856206649644645, AIC=1619.4097204319091, BIC=1643.5218083401855)
Fitted frechet_l distribution with error=0.01957333583165306, AIC=1595.553822837483, BIC=1619.6659107457594)
Fitted frechet_r distribution with error=0.01750706994602436, AIC=1595.4419257721615, BIC=1619.554013680438)
Fitted gamma distribution with error=0.031152057379370643, AIC=1631.8825979555052, BIC=1655.9946858637816)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_continuous_distns.py:3080: RuntimeWarning:

divide by zero encountered in power


Fitted gausshyper distribution with error=0.11634727273586078, AIC=1417.2854159013048, BIC=1465.5095917178578)
Fitted genexpon distribution with error=0.17290947148656494, AIC=1462.1722195597526, BIC=1502.3590327402135)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_continuous_distns.py:2276: RuntimeWarning:

invalid value encountered in subtract


Fitted genextreme distribution with error=1.5134323772320695, AIC=2338.580803282375, BIC=2362.6928911906516)
Fitted gengamma distribution with error=0.03704043446124279, AIC=1598.5031301557215, BIC=1630.652580700009)
Fitted genhalflogistic distribution with error=0.09329001232696904, AIC=1408.826645954011, BIC=1432.9387338622873)
Fitted genlogistic distribution with error=0.018133808151300765, AIC=1592.6892908076743, BIC=1616.8013787159507)
Fitted gennorm distribution with error=0.030012672811856787, AIC=1638.952184812683, BIC=1663.0642727209595)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_continuous_distns.py:2159: RuntimeWarning:

divide by zero encountered in true_divide
```

```
Fitted genpareto distribution with error=0.16188375173582553, AIC=1527.162492334986, BIC=1551.2745802
432623)
Fitted gilbrat distribution with error=0.16852743055709796, AIC=1590.6681017650642, BIC=1606.74282703
72484)
Fitted gompertz distribution with error=0.016368706947195697, AIC=1557.0727405235566, BIC=1581.184828
431833)
Fitted gumbel_l distribution with error=0.01689162861386545, AIC=1582.45282892197, BIC=1598.527554194
1542)
Fitted gumbel_r distribution with error=0.059280035676902584, AIC=1740.6077584434033, BIC=1756.682483
7155878)
Fitted halfcauchy distribution with error=0.20330071823442725, AIC=1591.5378957500377, BIC=1607.61262
1022222)
Fitted halfgennorm distribution with error=0.11363347604422838, AIC=1430.2717051432737, BIC=1454.3837
930515501)
Fitted halflogistic distribution with error=0.18390731362573182, AIC=1485.9356926623761, BIC=1502.010
4179345603)
Fitted halfnorm distribution with error=0.17774741660158366, AIC=1465.9152290413972, BIC=1481.9899543
135816)
Fitted hypsecant distribution with error=0.027268898345675592, AIC=1618.4969243875662, BIC=1634.57164
96597506)
Fitted invgamma distribution with error=0.03434835587131887, AIC=1653.2203239559612, BIC=1677.3324118
642377)
Fitted invgauss distribution with error=0.06001328707187271, AIC=1705.3026874015443, BIC=1729.4147753
098207)
Fitted invweibull distribution with error=0.06710302491962301, AIC=1824.3795393185994, BIC=1848.49162
72268758)
Fitted johnsonsb distribution with error=0.018471398957682628, AIC=1600.999880821551, BIC=1633.149331
3659196)
Fitted johnsonsu distribution with error=0.017256361020074745, AIC=1594.5228699430759, BIC=1626.67232
04874445)
Fitted kappa3 distribution with error=0.11247720657713255, AIC=1420.918898624646, BIC=1445.0309865329
225)
Fitted kappa4 distribution with error=0.12327923867363043, AIC=1410.4384966579132, BIC=1442.587947202
2818)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\integrate\q
uadpack.py:385: IntegrationWarning:

The maximum number of subdivisions (50) has been achieved.
  If increasing the limit yields no improvement it is advised to analyze
  the integrand in order to determine the difficulties.  If the position of a
  local difficulty can be determined (singularity, discontinuity) one will
  probably gain from splitting up the interval and calling the integrator
  on the subranges.  Perhaps a special-purpose integrator should be used.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:44: RuntimeWarning:

floating point number truncated to an integer


SKIPPED ksone distribution(taking more than 30 seconds)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_dist
n_infrastructure.py:1615: RuntimeWarning:

invalid value encountered in log


Fitted kstwobign distribution with error=0.06845878762400337, AIC=1792.7491903122936, BIC=1808.823915
584478)
Fitted laplace distribution with error=0.03627488772647625, AIC=1621.871815210546, BIC=1637.946540482
7301)
Fitted levy distribution with error=0.20548252422727858, AIC=1884.7454081832634, BIC=1900.82013345544
78)
Fitted levy_l distribution with error=0.15174726736263602, AIC=1766.9230884133867, BIC=1782.997813685
5711)
SKIPPED levy_stable distribution(taking more than 30 seconds)
Fitted loggamma distribution with error=0.01739769944039718, AIC=1593.040329385766, BIC=1617.15241729
40424)
Fitted logistic distribution with error=0.026705297830134006, AIC=1618.6208597432458, BIC=1634.695585
01543)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:3785: RuntimeWarning:

divide by zero encountered in true_divide
```

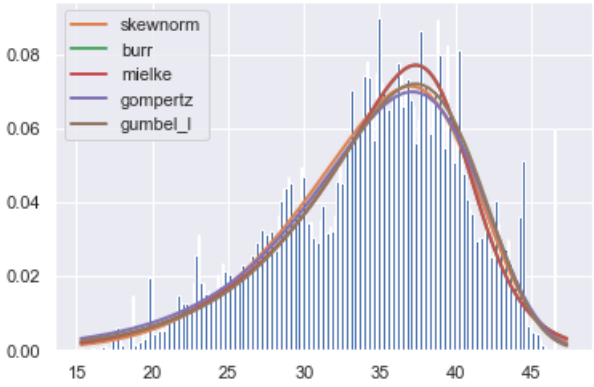
```
Fitted loglaplace distribution with error=0.036379265524075846, AIC=1656.1002716689177, BIC=1680.2123
59577194)
Fitted lognorm distribution with error=0.028348922939348625, AIC=1630.7679149696303, BIC=1654.8800028
779067)
Fitted lomax distribution with error=0.2164311736130175, AIC=1536.3858484421335, BIC=1560.49793635040
99)
Fitted maxwell distribution with error=0.05852273438771309, AIC=1570.4690278576645, BIC=1586.54375312
9849)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\integrate\q
uadpack.py:385: IntegrationWarning:

Extremely bad integrand behavior occurs at some points of the
  integration interval.

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\integrate\q
uadpack.py:385: IntegrationWarning:

The integral is probably divergent, or slowly convergent.


Fitted mielke distribution with error=0.016305400633825424, AIC=1585.0629724642472, BIC=1617.21242300
86158)
Fitted moyal distribution with error=0.07507799837411104, AIC=1803.6835239886088, BIC=1819.7582492607
93)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:4502: RuntimeWarning:

overflow encountered in double_scalars


Fitted nakagami distribution with error=0.02890139370804016, AIC=1633.6029647960668, BIC=1657.7150527
043432)
SKIPPED ncf distribution(taking more than 30 seconds)
SKIPPED nct distribution(taking more than 30 seconds)
Fitted ncx2 distribution with error=0.03236141554710886, AIC=1640.3850200321494, BIC=1672.53447057651
8)
Fitted norm distribution with error=0.027858943192317815, AIC=1628.0701594358097, BIC=1644.1448847079
942)
Fitted norminvgauss distribution with error=0.028141154338864237, AIC=1637.370919459494, BIC=1669.520
3700038626)
Fitted pareto distribution with error=0.308929847878725, AIC=1611.9608095603062, BIC=1636.07289746858
26)
Fitted pearson3 distribution with error=0.017168471546985465, AIC=1591.4451386928258, BIC=1615.557226
6011022)
Fitted powerlaw distribution with error=0.0972269358753526, AIC=1466.789113871757, BIC=1490.901201780
0335)
Fitted powerlognorm distribution with error=0.24907105059337167, AIC=2255.472449138276, BIC=2287.6218
996826447)
Fitted powernorm distribution with error=0.017125375289806483, AIC=1592.2156578408474, BIC=1616.32774
57491238)
Fitted rayleigh distribution with error=0.08884355170789815, AIC=1480.4362215247972, BIC=1496.5109467
969814)
Fitted rdist distribution with error=0.029599107320576447, AIC=1645.292855154457, BIC=1669.4049430627
333)
Fitted recipinvgauss distribution with error=0.031522148346545605, AIC=1643.419151745412, BIC=1667.53
12396536883)
Fitted reciprocal distribution with error=0.31667743229519707, AIC=inf, BIC=inf)
Fitted rice distribution with error=0.02786611588570663, AIC=1630.2154003482067, BIC=1654.32748825648
3)
SKIPPED rv_continuous distribution(taking more than 30 seconds)
SKIPPED rv_histogram distribution(taking more than 30 seconds)
Fitted semicircular distribution with error=0.08331734070989286, AIC=1418.10666372603, BIC=1434.18138
89982145)
Fitted skewnorm distribution with error=0.016223535018960115, AIC=1590.918437618583, BIC=1615.0305255
268595)
Fitted t distribution with error=0.027856347747901564, AIC=1630.0987645460687, BIC=1654.210852454345)
Fitted trapz distribution with error=0.10711198742605989, AIC=1531.3102270193112, BIC=1563.4596775636
799)
Fitted triang distribution with error=0.023801190874322763, AIC=1515.393762870499, BIC=1539.505850778
7753)
Fitted truncexpon distribution with error=0.16408309845472185, AIC=1405.403916746347, BIC=1429.516004
6546234)
```

```
E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5823: RuntimeWarning:

divide by zero encountered in log

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5823: RuntimeWarning:

invalid value encountered in log


Fitted truncnorm distribution with error=0.31667743229519707, AIC=inf, BIC=inf)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:5883: RuntimeWarning:

divide by zero encountered in power

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:4762: RuntimeWarning:

overflow encountered in exp

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:4768: RuntimeWarning:

invalid value encountered in multiply

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:4765: RuntimeWarning:

overflow encountered in multiply

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\stats\_cont
inuous_distns.py:4767: RuntimeWarning:

overflow encountered in multiply


SKIPPED tukeylambda distribution(taking more than 30 seconds)
Fitted uniform distribution with error=0.12375368367655376, AIC=1392.755523913875, BIC=1408.830249186
0595)
Fitted vonmises distribution with error=5.382650627688179e+54, AIC=-23929.861820202867, BIC=-23905.74
973229459)
Fitted vonmises_line distribution with error=0.02640929017431922, AIC=1596.7541185900504, BIC=1620.86
62064983268)
Fitted wald distribution with error=0.15075630699853995, AIC=1836.2360702797175, BIC=1852.31079555190
17)
Fitted weibull_max distribution with error=0.01957333583165306, AIC=1595.553822837483, BIC=1619.66591
07457594)
Fitted weibull_min distribution with error=0.01750706994602436, AIC=1595.4419257721615, BIC=1619.5540
13680438)
Fitted wrapcauchy distribution with error=nan, AIC=nan, BIC=nan)

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\scipy\optimize\op
timize.py:563: RuntimeWarning:

invalid value encountered in subtract

E:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\lib\site-packages\numpy\core\fromnu
meric.py:83: RuntimeWarning:

invalid value encountered in reduce
```

In [86]:
```
f_SDCons.summary()
```

Out[86]:

	sumsquare_error	AIC	BIC
skewnorm	0.016224	1590.918438	1615.030526
burr	0.016291	1584.998943	1617.148394
mielke	0.016305	1585.062972	1617.212423
gompertz	0.016369	1557.072741	1581.184828
gumbel_l	0.016892	1582.452829	1598.527554



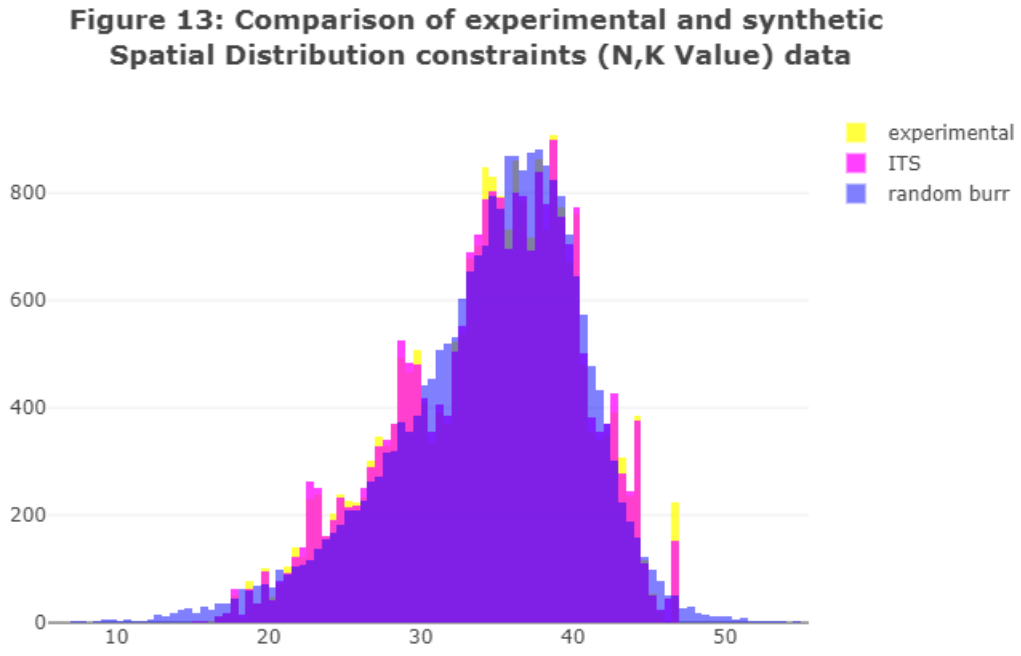
This is an interesting case of two distributions, Burr (Type III) and skew-normal, presenting opposing results vis-à-vis the sum of squared error and AIC/BIC. The Burr distribution has a slightly higher error but the AIC/BIC values are slightly lower than skew-normal.

As there is no other distinguishing feature that we can use to choose one over another, and probably they do not have practical differences, the Burr distribution will be used to generate random numbers along with ITS.

In [87]:
```
# create synthetic data

SpatialDistributionConstraints_random_burr = burr.rvs(f_SDCons.fitted_param['burr'][0],
                                                      f_SDCons.fitted_param['burr'][1],
                                                      f_SDCons.fitted_param['burr'][2],
                                                      f_SDCons.fitted_param['burr'][3], len(df))

SpatialDistributionConstraints_ITS = inverse_transform_sampling(df['Spatial Distribution constraints (N
,K Value)'])

SpatialDistributionConstraints_random_burr = SpatialDistributionConstraints_random_burr.astype(
                                    dtype = 'float32', casting = 'same_kind')

SpatialDistributionConstraints_ITS = SpatialDistributionConstraints_ITS.astype(
                                    dtype = 'float32', casting = 'same_kind')
```

In [117]:
```
trace_experimental = go.Histogram(x = df['Spatial Distribution constraints (N,K Value)'], opacity= 0.7
5,
                                  name = 'experimental',
                                  marker = dict(color = 'yellow'))

trace_ITS = go.Histogram(x = SpatialDistributionConstraints_ITS, opacity= 0.75,
                                  name = 'ITS',
                                  marker = dict(color = 'magenta'))

trace_random1 = go.Histogram(x = SpatialDistributionConstraints_random_burr, opacity= 0.5,
                                  name = 'random burr',
                                  marker = dict(color = 'blue'))

data = [trace_experimental, trace_ITS, trace_random1]

layout = go.Layout(barmode = 'overlay', title = '<b>Figure 13: Comparison of experimental and syntheti
c <br>' +
                   'Spatial Distribution constraints (N,K Value) data</b>')

fig = go.Figure(data=data, layout=layout)
# iplot(fig)
img_bytes = pio.to_image(fig, format='png')
Image(img_bytes)
```

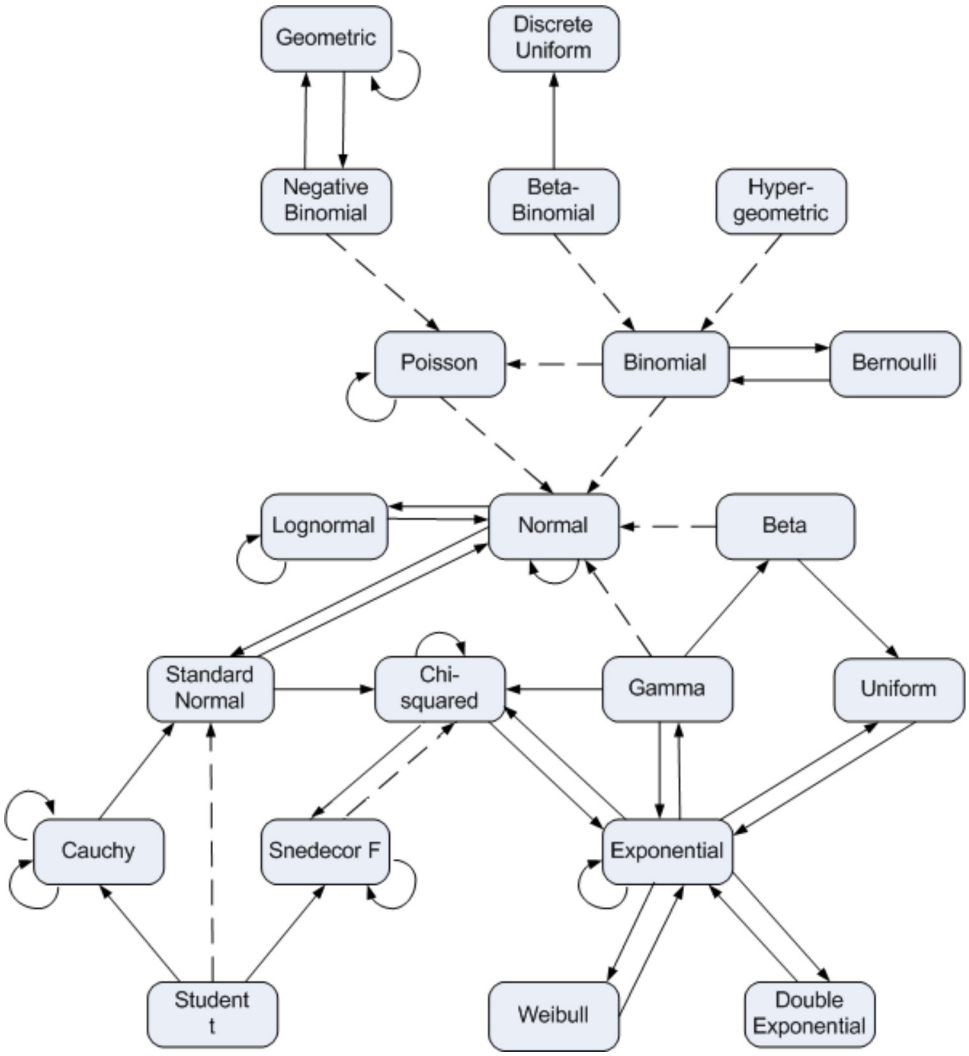
Out[117]:



From Figure 11, we can conclude that both the Burr distribution and ITS allow for a good reproduction of the experimental data.

Discussion

By fitting different distributions for each of the variable in the CASP dataset, we have seen that for most of them, we can fit more than one distribution. In fact, as the figure below shows, [probability distributions are related to one another (https://en.wikipedia.org/wiki/Relationships_among_probability_distributions)](https://en.wikipedia.org/wiki/Relationships_among_probability_distributions).



Taken from the blog of [John D. Cook (https://www.johndcook.com/blog/distribution_chart/)](https://www.johndcook.com/blog/distribution_chart/) with an original reference to [a paper by L. M. Leemis and J. T. McQueston (http://www.math.wm.edu/~leemis/2008amstat.pdf)](http://www.math.wm.edu/~leemis/2008amstat.pdf)

In these instances, the most parsimonious model (*i.e.* having the least number of parameters) should be used as this will lower the risk of overfitting (or fitting the random noise). The AIC and BIC measures are a good criterion for penalizing the more complicated models.

Having synthesized all the variables separately with ITS and `scipy.stats` distributions, we can now collate them into Pandas dataframes, write the values to csv files for future reference and also visually compare them by using box plots.

In [98]:

```
# because the numpy.ndarray values are horizontal and pd.DataFrame expects vertical arrangment,
# we have to transpose the arrays first if we are passing more than 1 ndarray.

# let's process the ITS datasets first

df_synthetic_ITS = pd.DataFrame(data = [RMSD_ITS, TotalArea_ITS, NonpolarExposedArea_ITS,
                                        FractionalAreaExposedNonPolarResidue_ITS,
                                        FractionalAreaExposedNonPolarPartOfResidue_ITS,
                                        MolecularMassWeightedExposedArea_ITS,
                                        AverageDeviationFromStandardExposedAreaOfResidue_ITS,
                                        EuclidianDistance_ITS, SecondaryStructurePenalty_ITS,
                                        SpatialDistributionConstraints_ITS]).T

# change the column headers to match the experimental columns
df_synthetic_ITS.columns = df.columns

# optimize data types

df_synthetic_ITS['RMSD'] = pd.to_numeric(df_synthetic_ITS['RMSD'], downcast='float')
df_synthetic_ITS['Total surface area'] = pd.to_numeric(df_synthetic_ITS['Total surface area'], downcast
='unsigned')
df_synthetic_ITS['Non-polar exposed area'] = pd.to_numeric(df_synthetic_ITS['Non-polar exposed area'],
downcast='float')

df_synthetic_ITS['Fractional area of exposed non-polar residue'] = pd.to_numeric(
    df_synthetic_ITS['Fractional area of exposed non-polar residue'], downcast='float')
df_synthetic_ITS['Fractional area of exposed non-polar part of residue'] = pd.to_numeric(
    df_synthetic_ITS['Fractional area of exposed non-polar part of residue'], downcast='float')

df_synthetic_ITS['Molecular mass weighted exposed area'] = pd.to_numeric(
    df_synthetic_ITS['Molecular mass weighted exposed area'], downcast='float')

df_synthetic_ITS['Average deviation from standard exposed area of residue'] = pd.to_numeric(
    df_synthetic_ITS['Average deviation from standard exposed area of residue'], downcast='float')

df_synthetic_ITS['Euclidian distance'] = pd.to_numeric(df_synthetic_ITS['Euclidian distance'], downcast
='float')
df_synthetic_ITS['Secondary structure penalty'] = pd.to_numeric(
    df_synthetic_ITS['Secondary structure penalty'], downcast='unsigned')

df_synthetic_ITS['Spatial Distribution constraints (N,K Value)'] = pd.to_numeric(
    df_synthetic_ITS['Spatial Distribution constraints (N,K Value)'], downcast='float')

# write the dataframe to a csv file
df_synthetic_ITS.to_csv('CASP_synthetic_inverse_transform_sampling.csv')
```

In [99]:

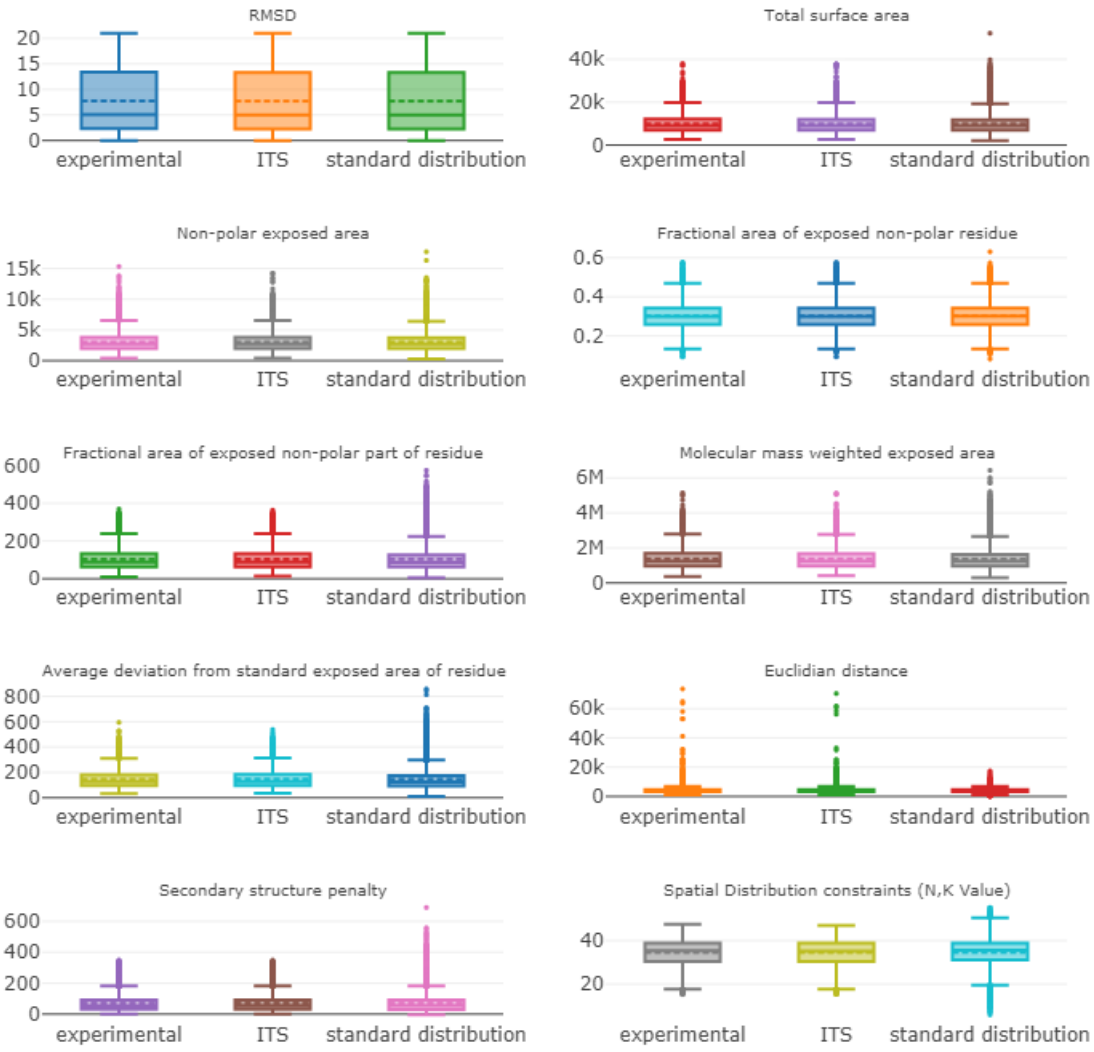
```
# do the same for rvs stuffs
# because we couldn't replicate RMSD by using a standard distribution, we'll take the one from ITS
# for cases with multiple distributions, we'll take the best one (lowest error + AIC/BIC)

df_synthetic_standardDistr = pd.DataFrame(data = [RMSD_ITS, TotalArea_random_moyal, NonpolarExposedArea
_random_moyal,
                                          FractionalAreaExposedNonPolarResidue_random_johnsonsu,
                                          FractionalAreaExposedNonPolarPartOfResidue_random_exponnorm,
                                          MolecularMassWeightedExposedArea_random_moyal,
                                          AverageDeviationFromStandardExposedAreaOfResidue_random_exponno
rm,
                                          EuclidianDistance_random_fisk, SecondaryStructurePenalty_random
_invgauss,
                                          SpatialDistributionConstraints_random_burr]).T

df_synthetic_standardDistr.columns = df.columns

df_synthetic_standardDistr['RMSD'] = pd.to_numeric(df_synthetic_standardDistr['RMSD'], downcast='float'
)

df_synthetic_standardDistr['Total surface area'] = pd.to_numeric(
    df_synthetic_standardDistr['Total surface area'], downcast='unsigned')

df_synthetic_standardDistr['Non-polar exposed area'] = pd.to_numeric(
    df_synthetic_standardDistr['Non-polar exposed area'], downcast='float')

df_synthetic_standardDistr['Fractional area of exposed non-polar residue'] = pd.to_numeric(
    df_synthetic_standardDistr['Fractional area of exposed non-polar residue'], downcast='float')

df_synthetic_standardDistr['Fractional area of exposed non-polar part of residue'] = pd.to_numeric(
    df_synthetic_standardDistr['Fractional area of exposed non-polar part of residue'], downcast='float
')

df_synthetic_standardDistr['Molecular mass weighted exposed area'] = pd.to_numeric(
    df_synthetic_standardDistr['Molecular mass weighted exposed area'], downcast='float')

df_synthetic_standardDistr['Average deviation from standard exposed area of residue'] = pd.to_numeric(
    df_synthetic_standardDistr['Average deviation from standard exposed area of residue'], downcast='fl
oat')

df_synthetic_standardDistr['Euclidian distance'] = pd.to_numeric(
    df_synthetic_standardDistr['Euclidian distance'], downcast='float')

df_synthetic_standardDistr['Secondary structure penalty'] = pd.to_numeric(
    df_synthetic_standardDistr['Secondary structure penalty'], downcast='unsigned')

df_synthetic_standardDistr['Spatial Distribution constraints (N,K Value)'] = pd.to_numeric(
    df_synthetic_standardDistr['Spatial Distribution constraints (N,K Value)'], downcast='float')

# write the dataframe to a csv file
df_synthetic_standardDistr.to_csv('CASP_synthetic_standard_distributions.csv')
```

In [118]:

```
trace_exp_0 = go.Box(y=df.iloc[:,0], marker = dict(size=3), name = 'experimental', boxmean = True)
trace_exp_1 = go.Box(y=df.iloc[:,1], marker = dict(size=3), name = 'experimental', boxmean = True)
trace_exp_2 = go.Box(y=df.iloc[:,2], marker = dict(size=3), name = 'experimental', boxmean = True)
trace_exp_3 = go.Box(y=df.iloc[:,3], marker = dict(size=3), name = 'experimental', boxmean = True)
trace_exp_4 = go.Box(y=df.iloc[:,4], marker = dict(size=3), name = 'experimental', boxmean = True)
trace_exp_5 = go.Box(y=df.iloc[:,5], marker = dict(size=3), name = 'experimental', boxmean = True)
trace_exp_6 = go.Box(y=df.iloc[:,6], marker = dict(size=3), name = 'experimental', boxmean = True)
trace_exp_7 = go.Box(y=df.iloc[:,7], marker = dict(size=3), name = 'experimental', boxmean = True)
trace_exp_8 = go.Box(y=df.iloc[:,8], marker = dict(size=3), name = 'experimental', boxmean = True)
trace_exp_9 = go.Box(y=df.iloc[:,9], marker = dict(size=3), name = 'experimental', boxmean = True)

trace_ITS_0 = go.Box(y=df_synthetic_ITS.iloc[:,0], marker = dict(size=3), name = 'ITS', boxmean = True
)
trace_ITS_1 = go.Box(y=df_synthetic_ITS.iloc[:,1], marker = dict(size=3), name = 'ITS', boxmean = True
)
trace_ITS_2 = go.Box(y=df_synthetic_ITS.iloc[:,2], marker = dict(size=3), name = 'ITS', boxmean = True
)
trace_ITS_3 = go.Box(y=df_synthetic_ITS.iloc[:,3], marker = dict(size=3), name = 'ITS', boxmean = True
)
trace_ITS_4 = go.Box(y=df_synthetic_ITS.iloc[:,4], marker = dict(size=3), name = 'ITS', boxmean = True
)
trace_ITS_5 = go.Box(y=df_synthetic_ITS.iloc[:,5], marker = dict(size=3), name = 'ITS', boxmean = True
)
trace_ITS_6 = go.Box(y=df_synthetic_ITS.iloc[:,6], marker = dict(size=3), name = 'ITS', boxmean = True
)
trace_ITS_7 = go.Box(y=df_synthetic_ITS.iloc[:,7], marker = dict(size=3), name = 'ITS', boxmean = True
)
trace_ITS_8 = go.Box(y=df_synthetic_ITS.iloc[:,8], marker = dict(size=3), name = 'ITS', boxmean = True
)
trace_ITS_9 = go.Box(y=df_synthetic_ITS.iloc[:,9], marker = dict(size=3), name = 'ITS', boxmean = True
)

trace_rvs_0 = go.Box(y=df_synthetic_standardDistr.iloc[:,0], marker = dict(size=3), name = 'standard d
istribution', boxmean = True)
trace_rvs_1 = go.Box(y=df_synthetic_standardDistr.iloc[:,1], marker = dict(size=3), name = 'standard d
istribution', boxmean = True)
trace_rvs_2 = go.Box(y=df_synthetic_standardDistr.iloc[:,2], marker = dict(size=3), name = 'standard d
istribution', boxmean = True)
trace_rvs_3 = go.Box(y=df_synthetic_standardDistr.iloc[:,3], marker = dict(size=3), name = 'standard d
istribution', boxmean = True)
trace_rvs_4 = go.Box(y=df_synthetic_standardDistr.iloc[:,4], marker = dict(size=3), name = 'standard d
istribution', boxmean = True)
trace_rvs_5 = go.Box(y=df_synthetic_standardDistr.iloc[:,5], marker = dict(size=3), name = 'standard d
istribution', boxmean = True)
trace_rvs_6 = go.Box(y=df_synthetic_standardDistr.iloc[:,6], marker = dict(size=3), name = 'standard d
istribution', boxmean = True)
trace_rvs_7 = go.Box(y=df_synthetic_standardDistr.iloc[:,7], marker = dict(size=3), name = 'standard d
istribution', boxmean = True)
trace_rvs_8 = go.Box(y=df_synthetic_standardDistr.iloc[:,8], marker = dict(size=3), name = 'standard d
istribution', boxmean = True)
trace_rvs_9 = go.Box(y=df_synthetic_standardDistr.iloc[:,9], marker = dict(size=3), name = 'standard d
istribution', boxmean = True)

# we have to make subplots because the scales are different for the variables
fig = tools.make_subplots(rows=5, cols=2, print_grid=False,
                          subplot_titles= (df.columns[0], df.columns[1], df.columns[2], df.columns[3],
                                           df.columns[4], df.columns[5], df.columns[6], df.columns[7],
                                           df.columns[8], df.columns[9]
                                                          ))

fig.append_trace(trace_exp_0, 1, 1)
fig.append_trace(trace_ITS_0, 1, 1)
fig.append_trace(trace_rvs_0, 1, 1)

fig.append_trace(trace_exp_1, 1, 2)
fig.append_trace(trace_ITS_1, 1, 2)
fig.append_trace(trace_rvs_1, 1, 2)

fig.append_trace(trace_exp_2, 2, 1)
fig.append_trace(trace_ITS_2, 2, 1)
fig.append_trace(trace_rvs_2, 2, 1)

fig.append_trace(trace_exp_3, 2, 2)
fig.append_trace(trace_ITS_3, 2, 2)
fig.append_trace(trace_rvs_3, 2, 2)

fig.append_trace(trace_exp_4, 3, 1)
fig.append_trace(trace_ITS_4, 3, 1)
fig.append_trace(trace_rvs_4, 3, 1)

fig.append_trace(trace_exp_5, 3, 2)
fig.append_trace(trace_ITS_5, 3, 2)
```

Out[118]:

Figure 14: Box Plot of experimental and synthetic CASP datasets



From the comparative box plot above, the following can be noted:

1. both ITS and several of the standard probability distributions can be used to synthesize datasets that have similar characteristics as the experimental dataset
2. ITS provides a method for a more accurate data simulation compared to the standard distributions, due to simplified parametrization leading to constrained fitting of the experimental data to fit the model. This is especially so for a few variables where outliers (not present in the experimental dataset) were generated by the standard distribution methods.
3. for many variables, the ranges of synthesized data by standard distributions are much wider than the experimental and ITS datasets.

Conclusion

Experimental dataset are valuable as they capture the nuances of the system under study. However, in many instances, it is not possible to obtain a large number of datapoints, especially when it involves long duration or high expenses. For those situations, it would be useful to generate a set of random values that resemble the experimental data. By having a combined 'experimental + synthetic' dataset, we can gain the ability to train machine learning algorithms, where highly-variant large datasets is a pre-requisite.

Similarly, the use of synthetic datasets helps in the case of parameter estimation for complex (highly parametrized) mechanistic models. Bayesian techniques such as Markov chain Monte Carlo (MCMC) simulations can be performed from a simulated dataset following a verification on the experimental dataset.

Through this notebook, we have learned how the CASP dataset is organized and the different abilities of `numpy.random` and `scipy.stats` libraries as well as their augmentation by inverse transform sampling.