techA_1overf

January 18, 2022

1 RRAM Relaxation 1/f Fitting

This notebook contains the fitting for 1/f RRAM relaxation data across three technologies (A, B, C). It loads and processes the measurements taken for each technology to enable fitting and understanding of the data.

[ ]:
```
# Imports
import gzip
import json
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import scipy.signal
import scipy.stats

%config InlineBackend.figure_format = 'svg'
```

1.1 Load the technology and its settings

Below, choose which technology to load data and settings for:

[ ]:
```
# Choose technology here
TECH = 'A'

# Load settings for technology
with open(f"data/tech{TECH}/settings.json") as sfile:
    settings = json.load(sfile)
```

1.2 Load the conductance data vs. time

Here, we will load the full dataset of conductance over time at room temperature to look at the 1/f behaviors of cells.

[ ]:
```
# Load data for technology
colnames = ["addr", "time", "r", "g"]
data = pd.read_csv(f"data/tech{TECH}/relaxdata.tsv.gz", names=colnames,␣
 ↪sep='\t')
data = data[data["addr"] % 2 == 0] # use only even addresses (odd addresses␣
 ↪have weird behavior due to 2T2R architecture)
```

```
data["time"] -= data.groupby("addr")["time"].transform("first") # start from␣
 ↪t=0 for each addr
data["gi"] = data.groupby("addr")["g"].transform("first") # get initial␣
 ↪conductance each addr was programmed to
data["range"] = np.int32(data["gi"] / settings["gmax"] * 32) # get initial␣
 ↪conductance range each addr was programmed to
data.head()
```

[ ]:

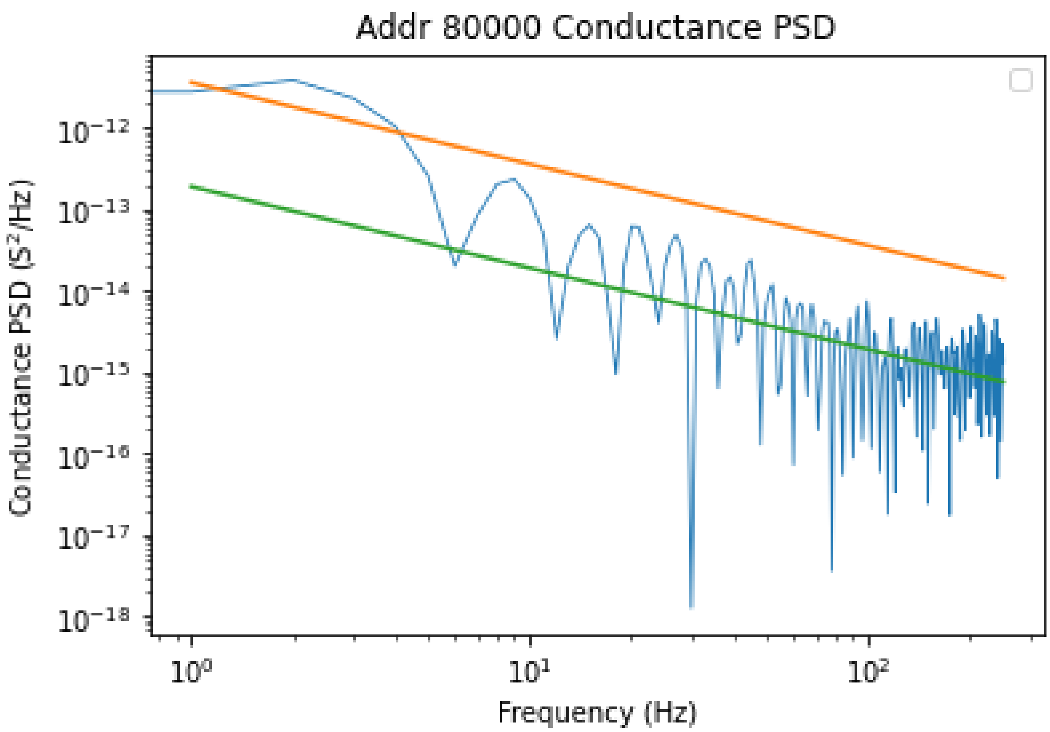
	addr	time	r	g	gi	range
0	80000	0.000000	337532.079408	0.000003	0.000003	0
1	80000	0.002000	373198.486437	0.000003	0.000003	0
2	80000	0.003998	283369.159874	0.000004	0.000003	0
3	80000	0.005997	280555.408797	0.000004	0.000003	0
4	80000	0.007995	266026.890670	0.000004	0.000003	0

[ ]:

```
# Plot some 1/f PSDs
for a in np.array(range(0, 32, 8)) + (80000 if TECH == 'A' else 0):
    # Select which addr is being studied
    print(f"addr = {a}")
    d = data[(data["addr"] == a) & (data["time"] < 10)]
    gvals = d.drop_duplicates(subset=["time"]).sort_values(["time"])

    # Compute sampling time
    fs = 1/np.median(np.gradient(gvals['time']))
    print(f"fs = {fs}")
    freq, p = scipy.signal.welch(gvals["g"], fs, nperseg=len(gvals["g"]))

    # Fit functions
    fitfn1 = lambda f, c: c / f
    fitfn2 = lambda logf, c: c - logf
    fit1 = scipy.optimize.curve_fit(fitfn1, freq[1:], p[1:])
    fit2 = scipy.optimize.curve_fit(fitfn2, np.log10(freq[1:]), np.log10(p[1:]))
    print(f"Fit 1 coefficient = {fit1[0][0]}")
    print(f"Fit 2 coefficient = {10**fit2[0][0]}")

    # Plot PSD and fits
    plt.legend()
    plt.plot(freq, p, linewidth=0.8)
    plt.plot(freq[1:], fitfn1(freq[1:], fit1[0][0]))
    plt.plot(freq[1:], 10**fitfn2(np.log10(freq[1:]), fit2[0][0]))
    plt.title(f"Addr {a} Conductance PSD")
    plt.xlabel("Frequency (Hz)")
    plt.ylabel("Conductance PSD (S$^2$/Hz)")
    plt.xscale("log")
    plt.yscale("log")
    plt.show()
```

```
addr = 80000
fs = 500.2748091603053
Fit 1 coefficient = 3.607381673229551e-12
Fit 2 coefficient = 1.9086629616884373e-13
```

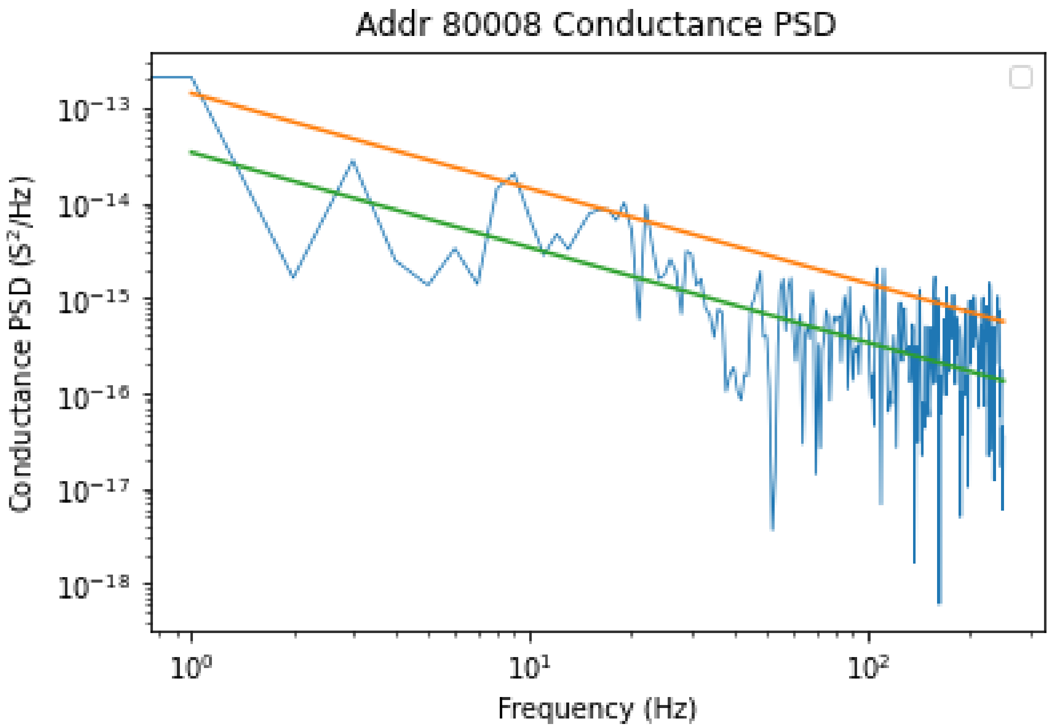
```
No handles with labels found to put in legend.
```




```
addr = 80008
```

```
No handles with labels found to put in legend.
```

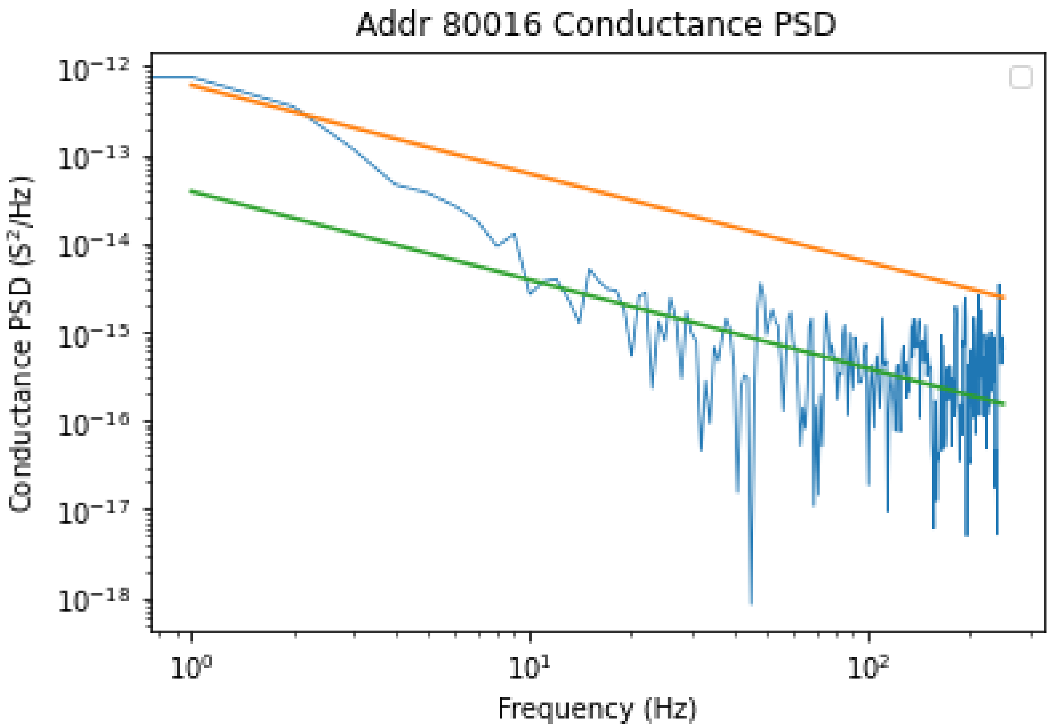
```
fs = 500.304646030894
Fit 1 coefficient = 1.4137518617949344e-13
Fit 2 coefficient = 3.3843958001993084e-14
```




```
addr = 80016
```

```
No handles with labels found to put in legend.
```

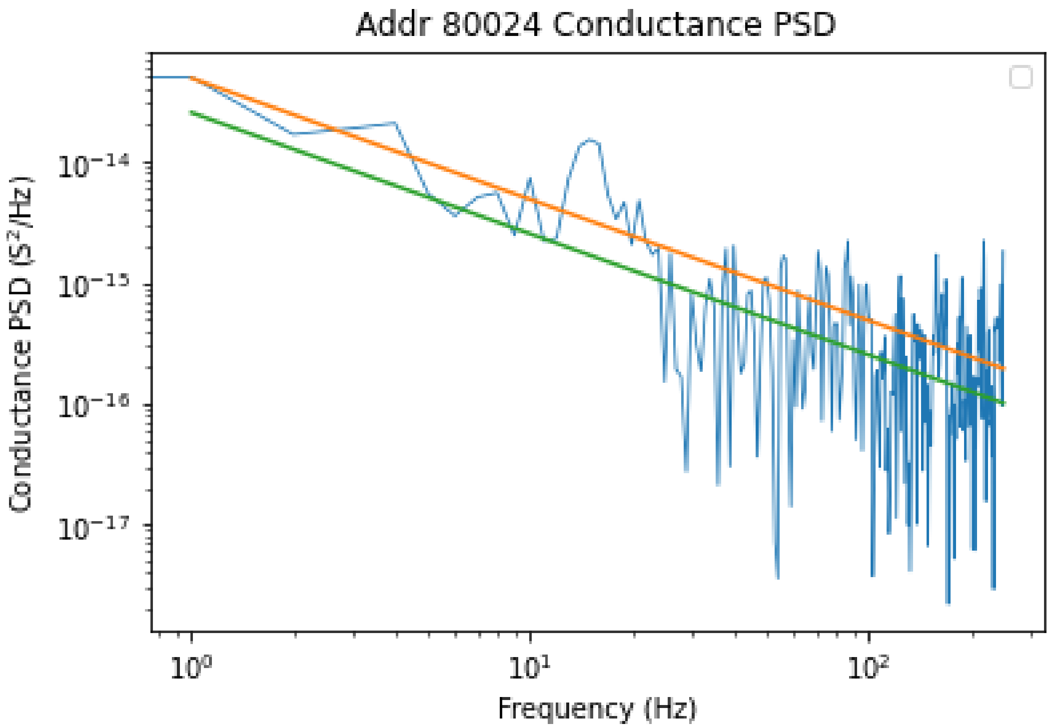
```
fs = 500.304646030894
Fit 1 coefficient = 6.107242904001814e-13
Fit 2 coefficient = 3.851642732588909e-14
```




```
addr = 80024
```

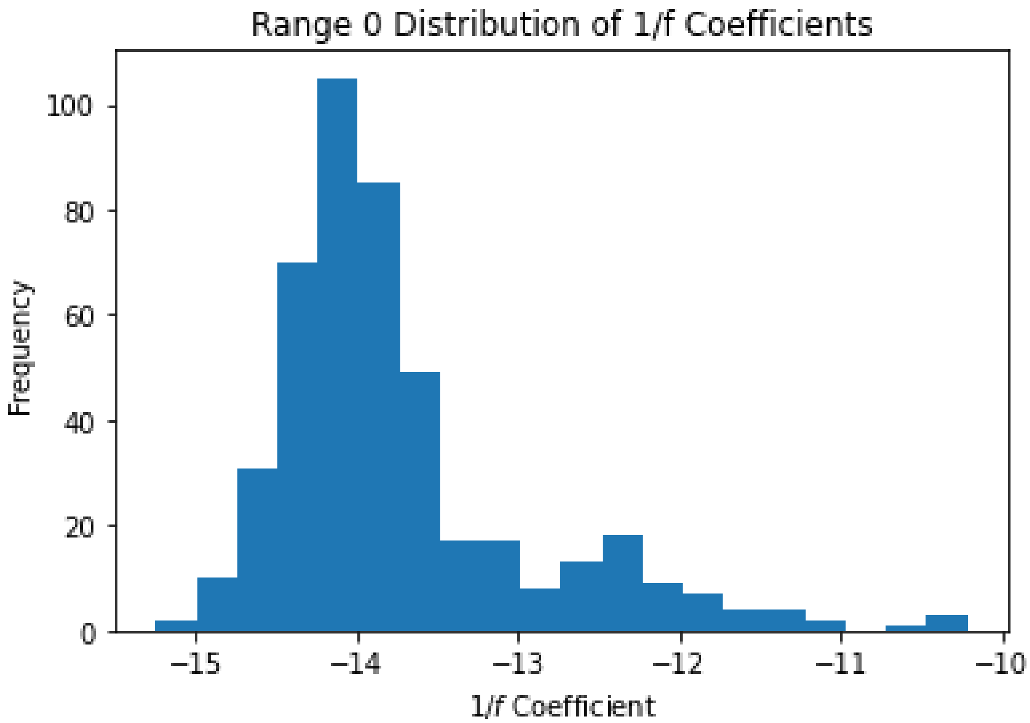
```
No handles with labels found to put in legend.
```

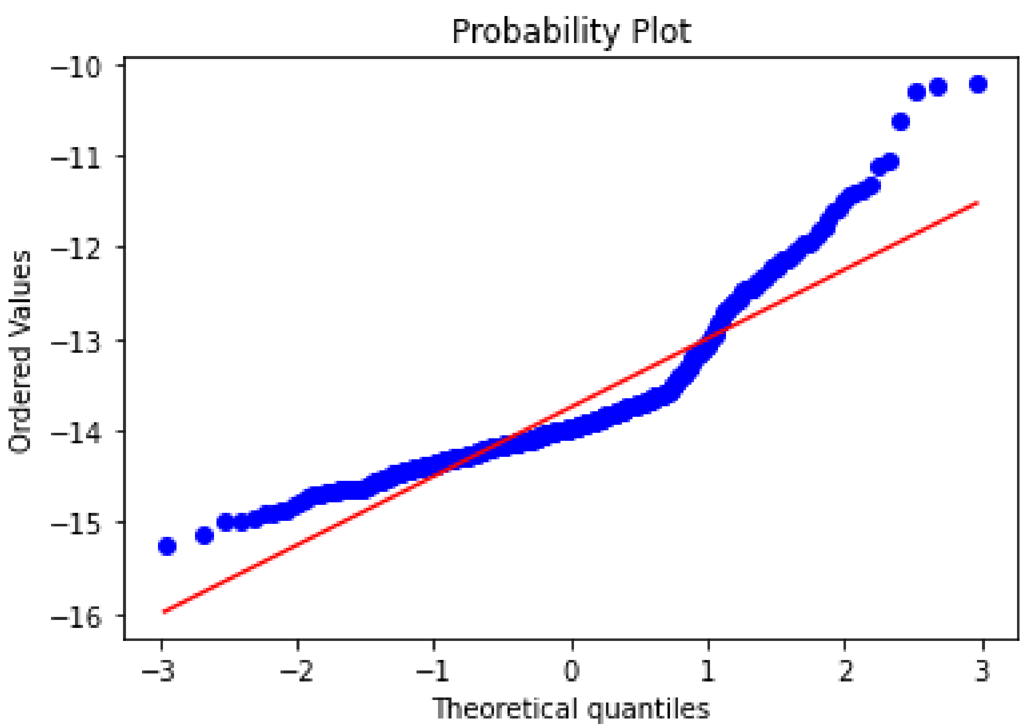
```
fs = 500.304646030894
Fit 1 coefficient = 4.863652821956458e-14
Fit 2 coefficient = 2.526169963985075e-14
```

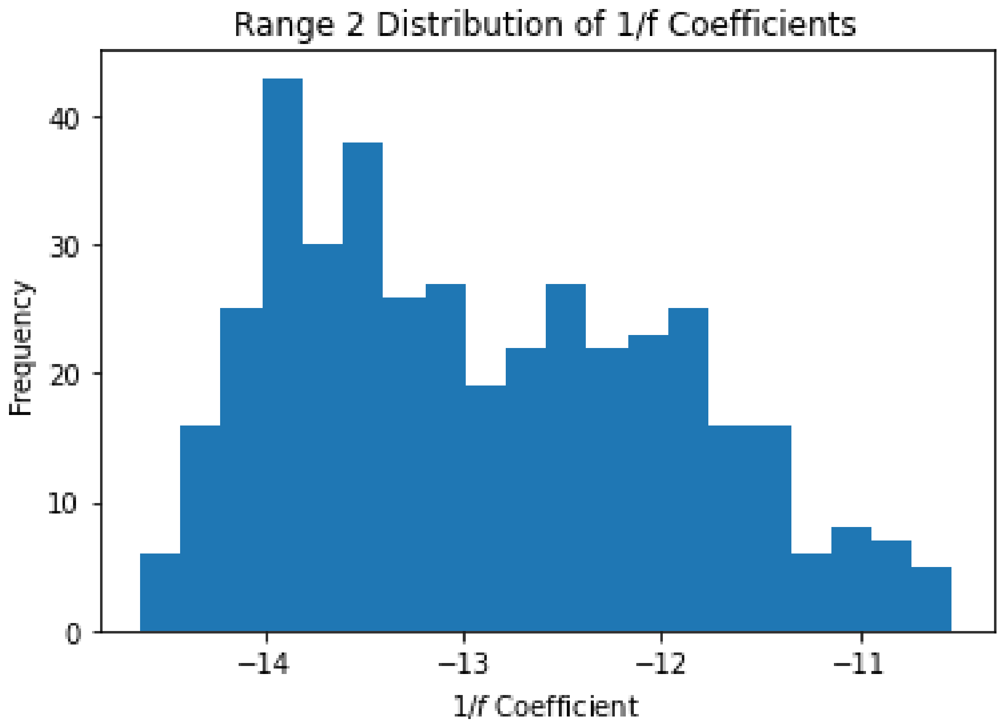


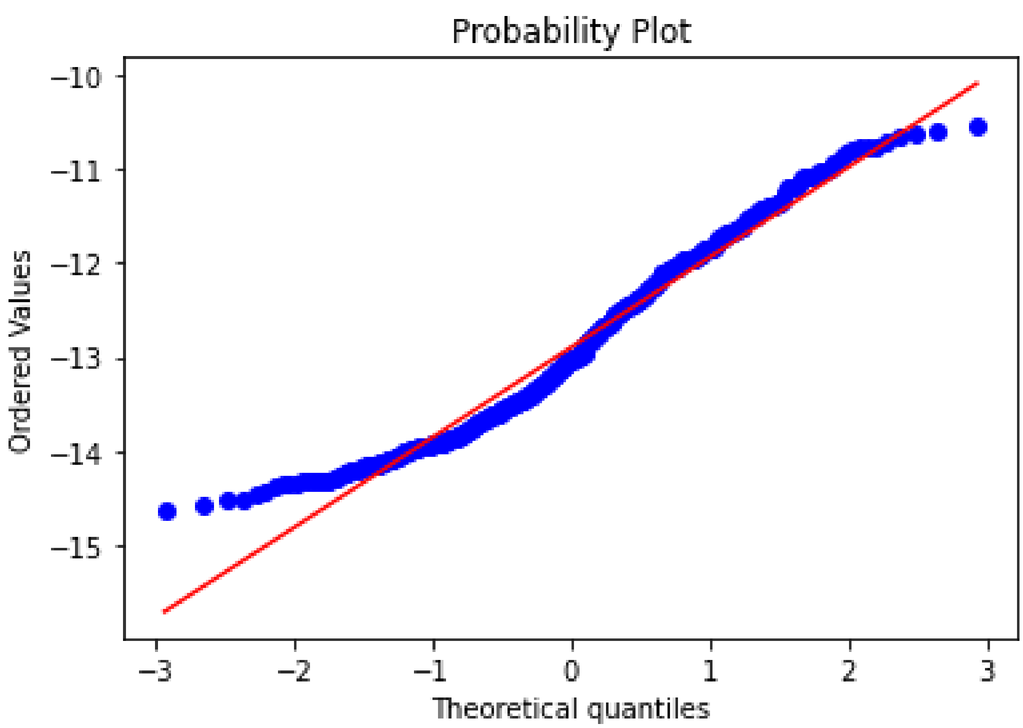
[ ]:
```
# 1/f coefficient statistics
fitfn = lambda f, c: c / f
fitfnloglog = lambda logf, c: c - logf

# Noise fitting function
def noise_fit(gvals, loglogfit=False):
    gvals = gvals.drop_duplicates(subset=['time']).sort_values(["time"])
    freq, p = scipy.signal.welch(gvals["g"], fs=1/np.median(np.
↪gradient(gvals["time"])), nperseg=len(gvals["g"]))
    if loglogfit:
        popt, pcov = scipy.optimize.curve_fit(fitfnloglog, np.log10(freq[1:]),␣
↪np.log10(p[1:]))
        return 10**popt[0]
    else:
        popt, pcov = scipy.optimize.curve_fit(fitfn, freq[1:], p[1:])
        return popt[0]

# Log-normal fitting of 1/f coefficients
mus, sigmas = [], []
ranges = np.arange(0, 32, 2)
for r in ranges:
    # Fit 1/f coefficients
```

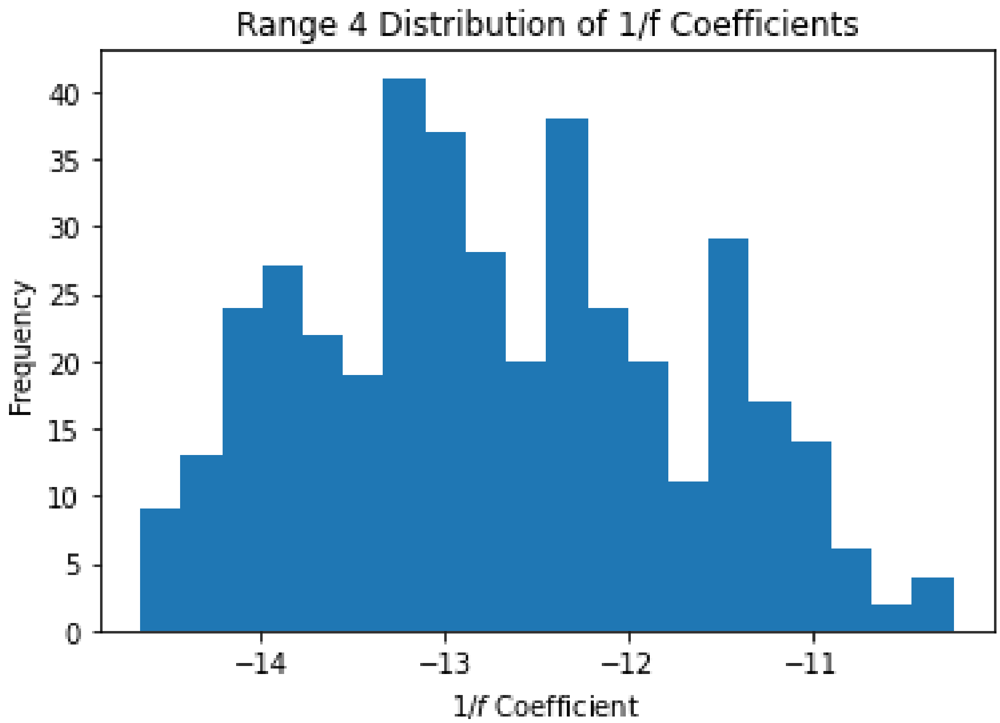
```
    coefs = data[(data["range"]==r) & (data["time"] < 10)].
↪groupby("addr")[["g", "time"]].apply(noise_fit)

    # Plot log-normal histogram
    plt.hist(np.log10(coefs), bins=20)
    plt.title(f"Range {r} Distribution of 1/f Coefficients")
    plt.xlabel("$1/f$ Coefficient")
    plt.ylabel("Frequency")
    plt.show()

    # Plot log-normal fit
    scipy.stats.probplot(np.log10(coefs), dist='norm', plot=plt, fit=True)
    plt.show()

    # Get log-normal fit parameters
    shape, loc, scale = scipy.stats.lognorm.fit(coefs, floc=0)
    mus.append(np.log(scale))
    sigmas.append(shape)

# Plot mus
plt.plot(ranges, mus)
plt.title(f"$\mu$ of Log-Normal $1/f$ Coefficients")
plt.xlabel("Range Number")
plt.ylabel("$\mu$")
plt.show()

# Plot sigmas
plt.plot(ranges, sigmas)
plt.title(f"$\sigma$ of Log-Normal $1/f$ Coefficients")
plt.xlabel("Range Number")
plt.ylabel("$\sigma$")
plt.show()
```

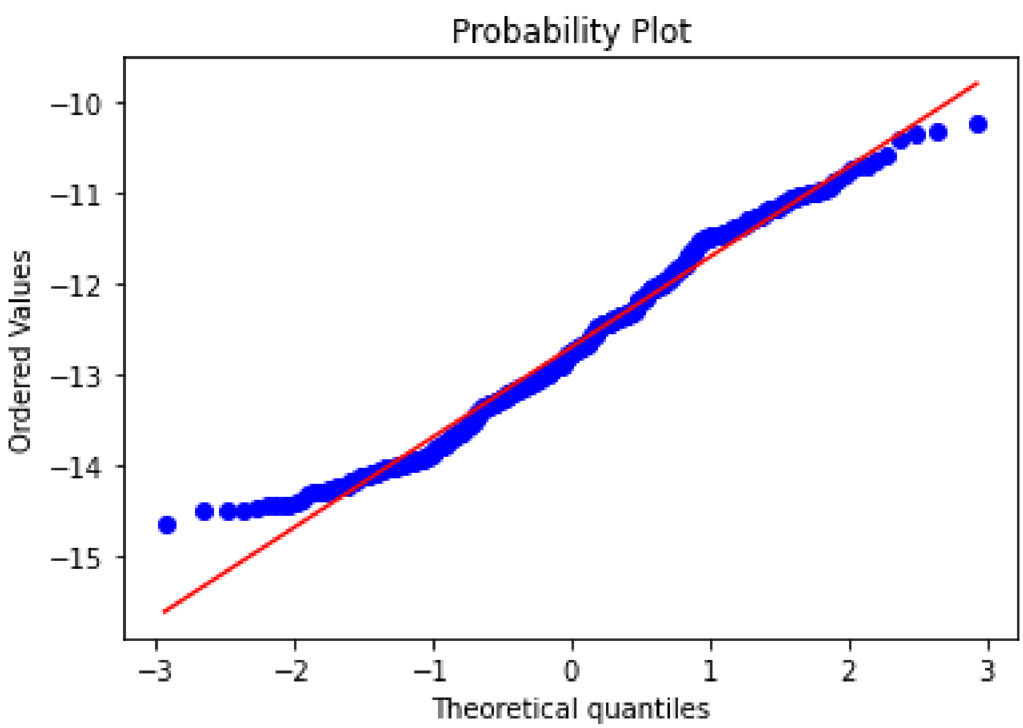

Range 0 Distribution of 1/f Coefficients
Frequency
1/f Coefficient
0
20
40
60
80
100
−15
−14
−13
−12
−11
−10

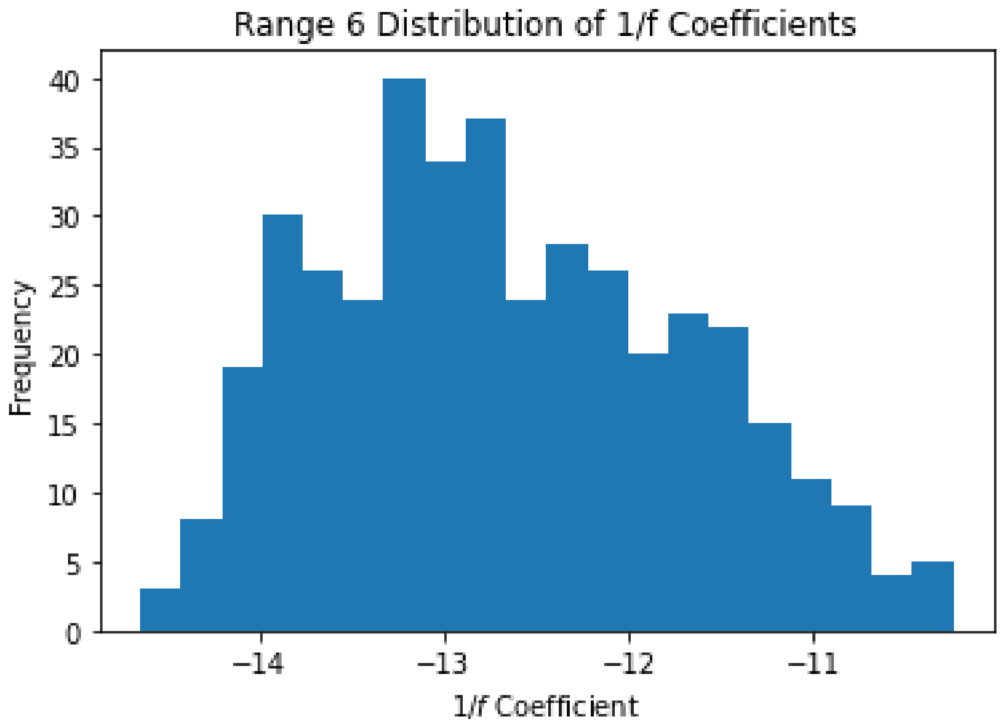

Probability Plot
Ordered Values
Theoretical quantiles
−10
−11
−12
−13
−14
−15
−16
−3
−2
−1
0
1
2
3

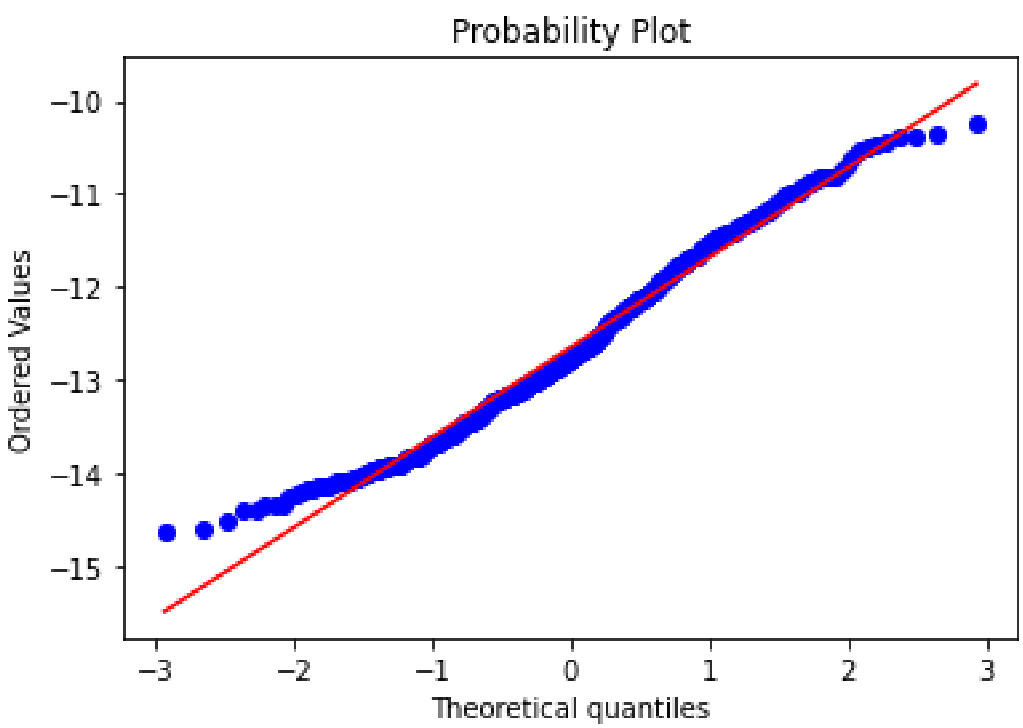

Range 2 Distribution of 1/f Coefficients
Frequency
1/f Coefficient
0
10
20
30
40
−14
−13
−12
−11

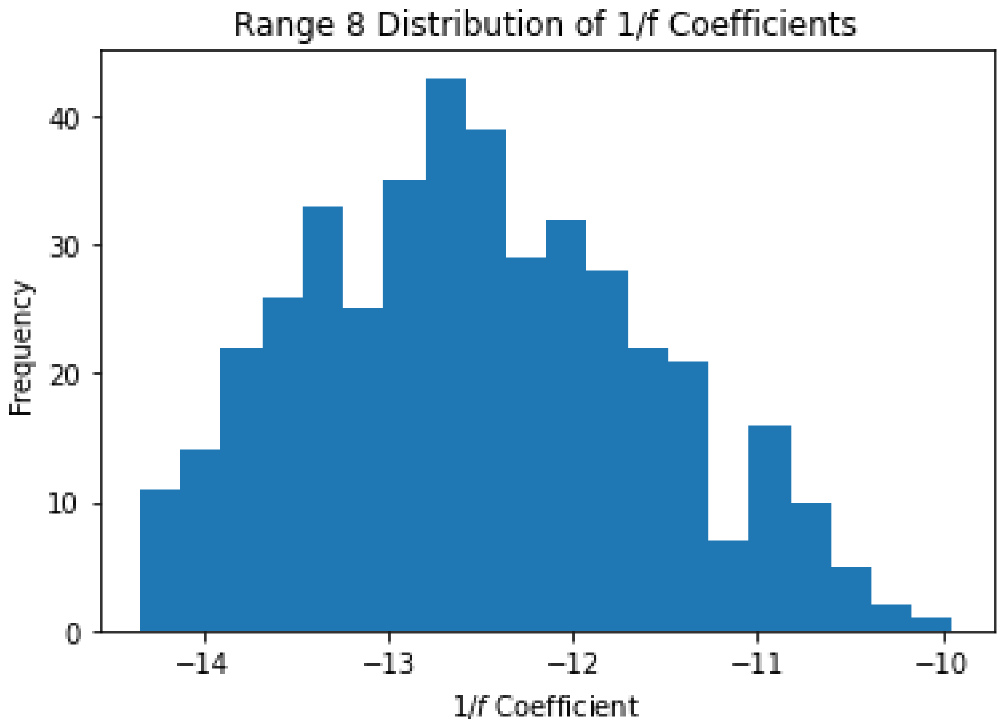

Probability Plot
Ordered Values
Theoretical quantiles
−10
−11
−12
−13
−14
−15
−3
−2
−1
0
1
2
3

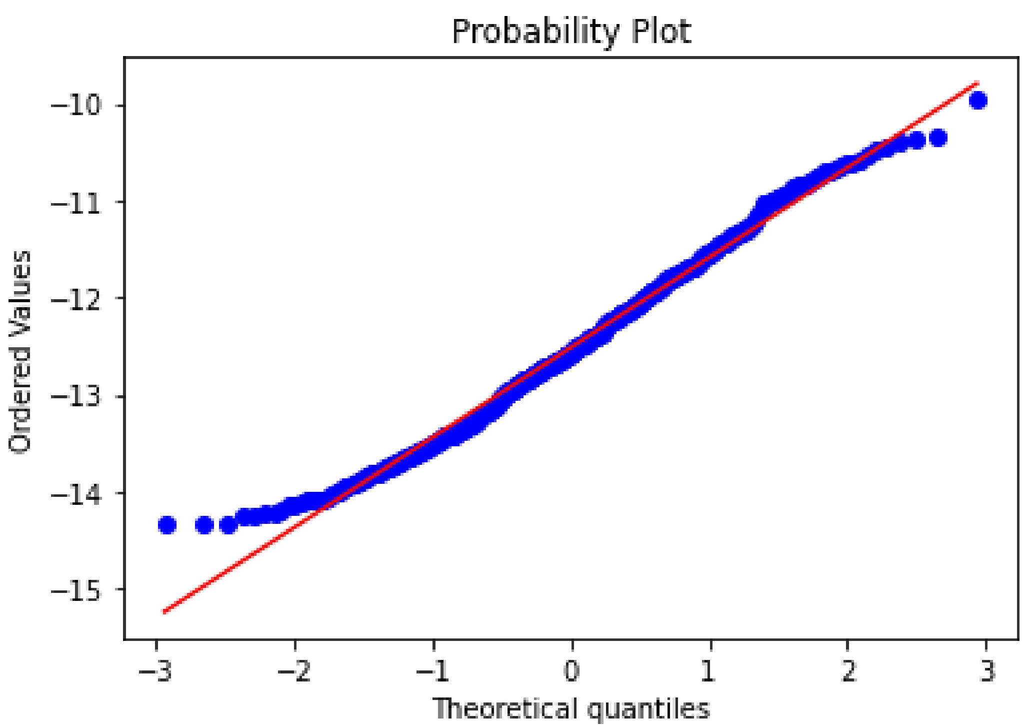

Range 4 Distribution of 1/f Coefficients
Frequency
1/f Coefficient
0
5
10
15
20
25
30
35
40
−14
−13
−12
−11

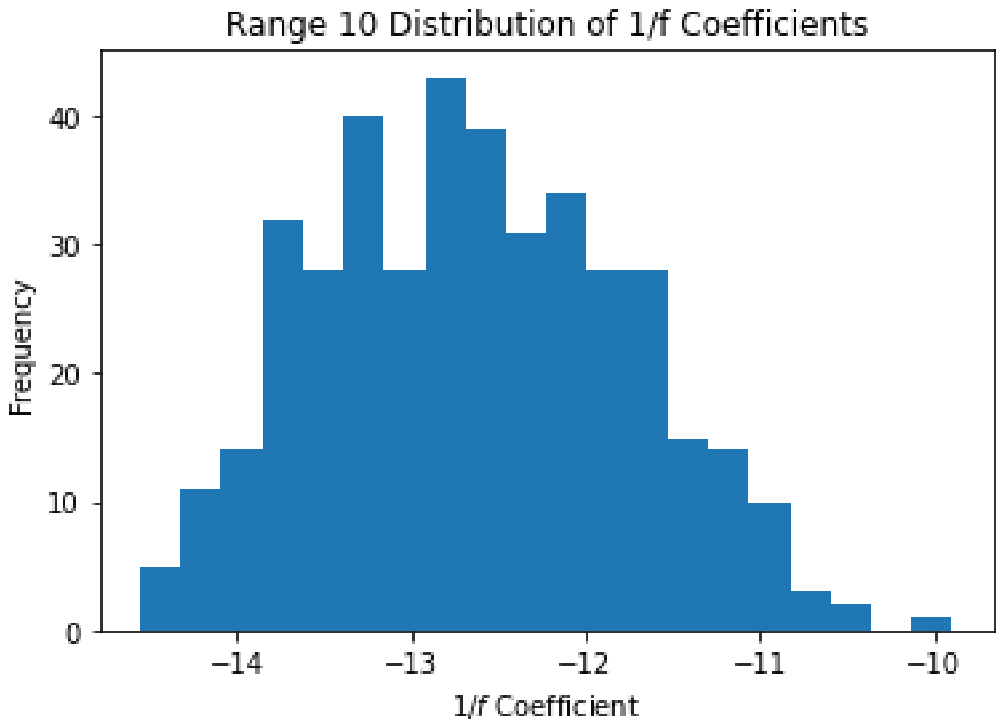

Probability Plot
Ordered Values
Theoretical quantiles
−10
−11
−12
−13
−14
−15
−3
−2
−1
0
1
2
3

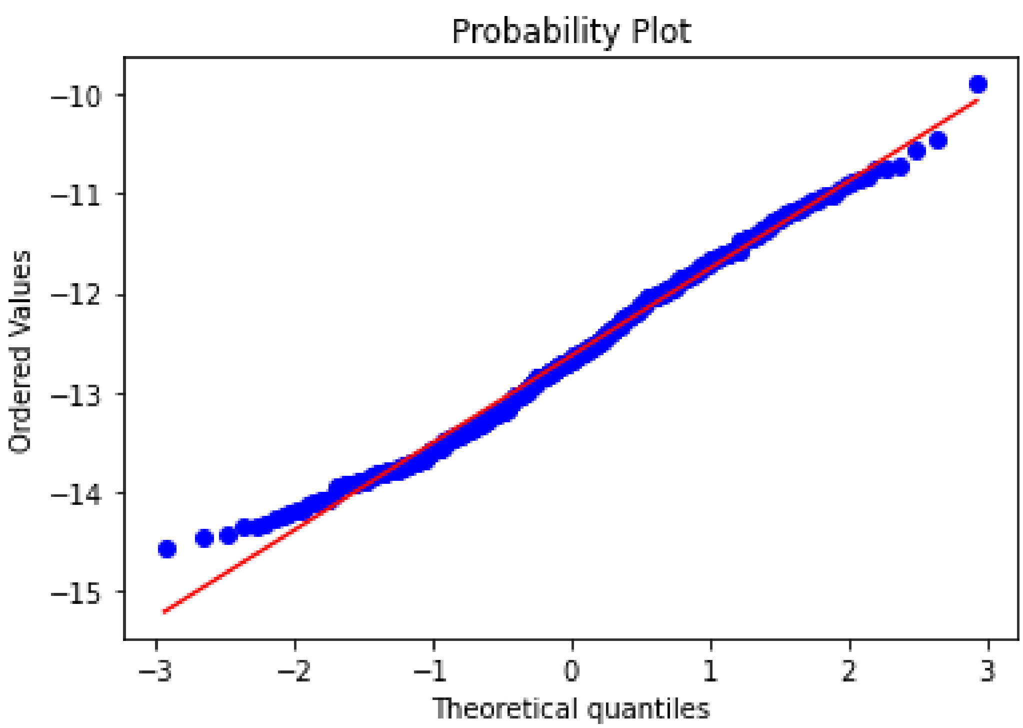

Range 6 Distribution of 1/f Coefficients
Frequency
1/f Coefficient
0
5
10
15
20
25
30
35
40
−14
−13
−12
−11

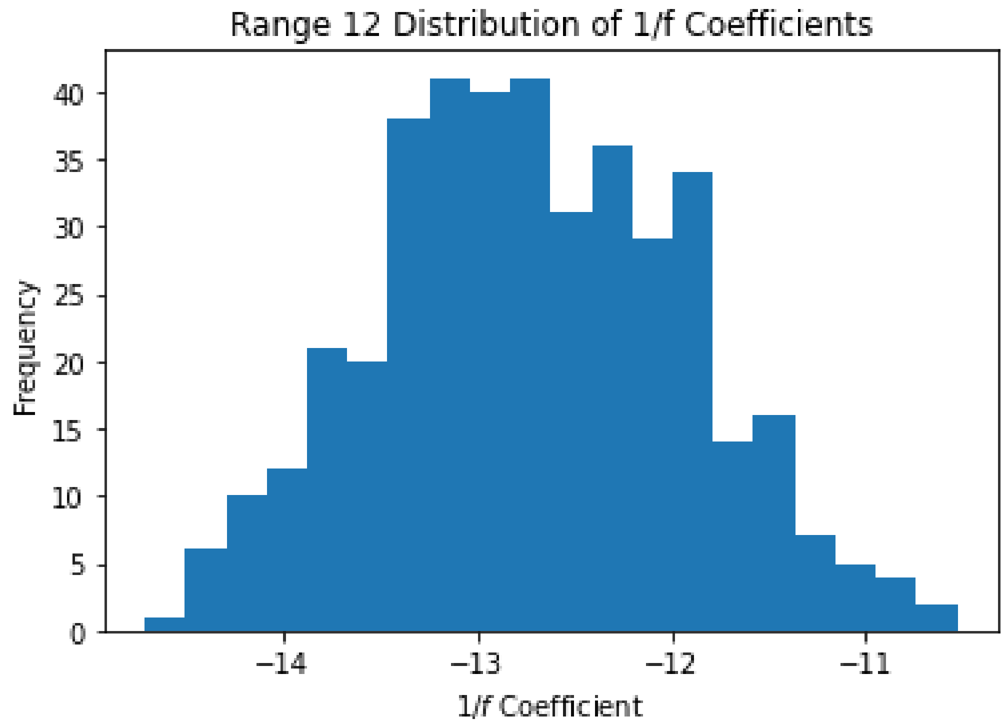

Probability Plot
Ordered Values
Theoretical quantiles
−10
−11
−12
−13
−14
−15
−3
−2
−1
0
1
2
3

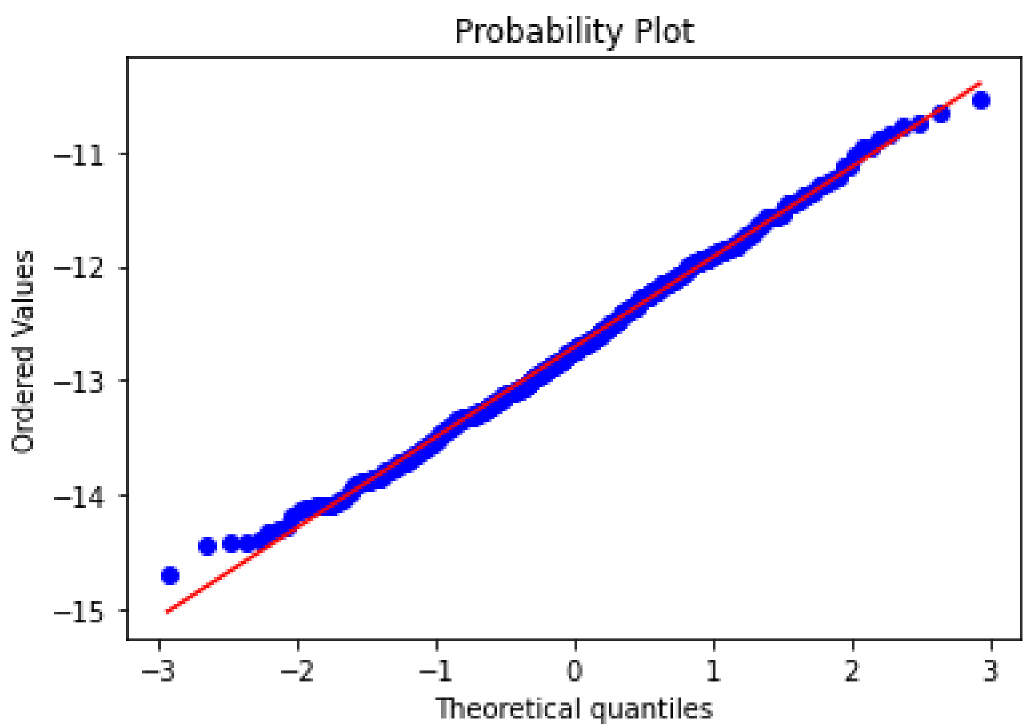

Range 8 Distribution of 1/f Coefficients
Frequency
1/f Coefficient
0
10
20
30
40
−14
−13
−12
−11
−10

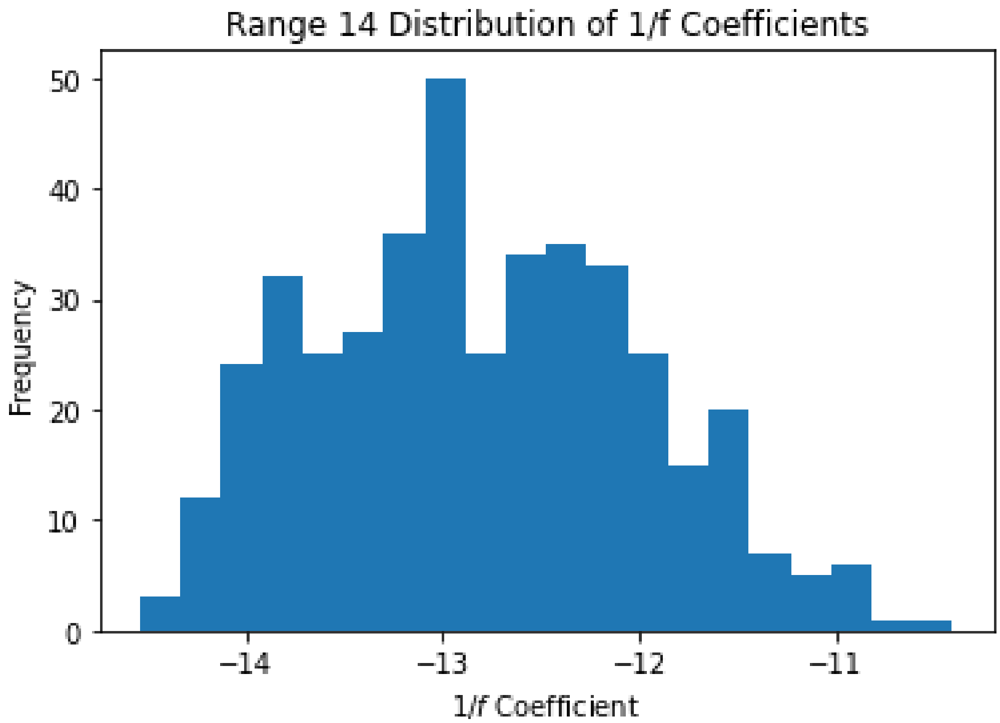

Probability Plot
Ordered Values
Theoretical quantiles
−10
−11
−12
−13
−14
−15
−3
−2
−1
0
1
2
3

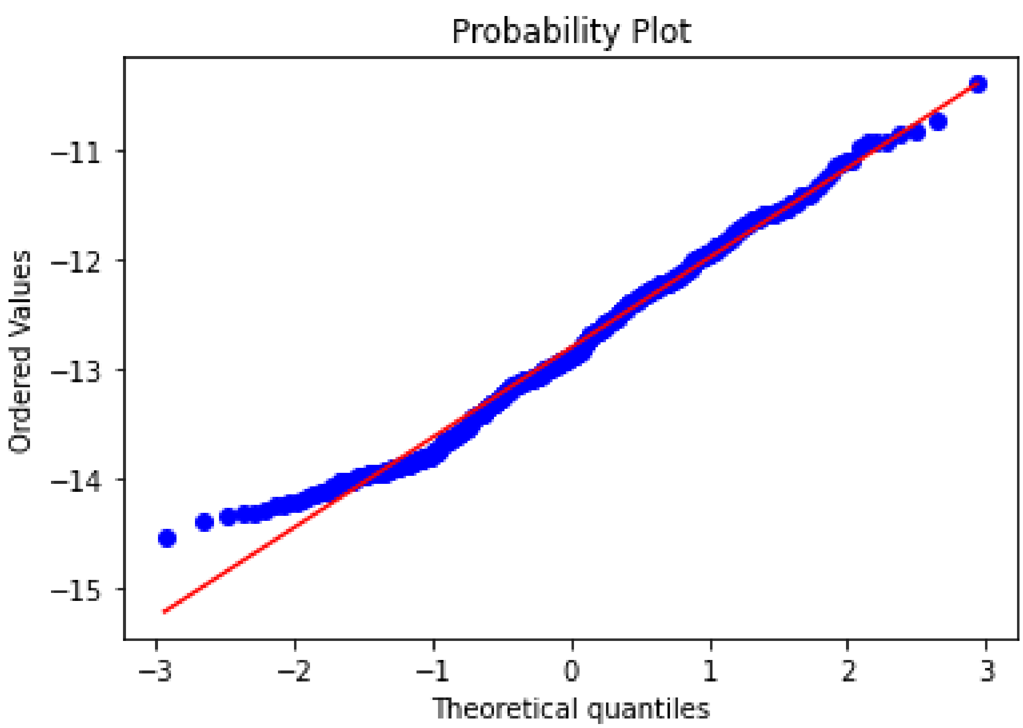

Range 10 Distribution of 1/f Coefficients
Frequency
1/f Coefficient
0
10
20
30
40
−14
−13
−12
−11
−10

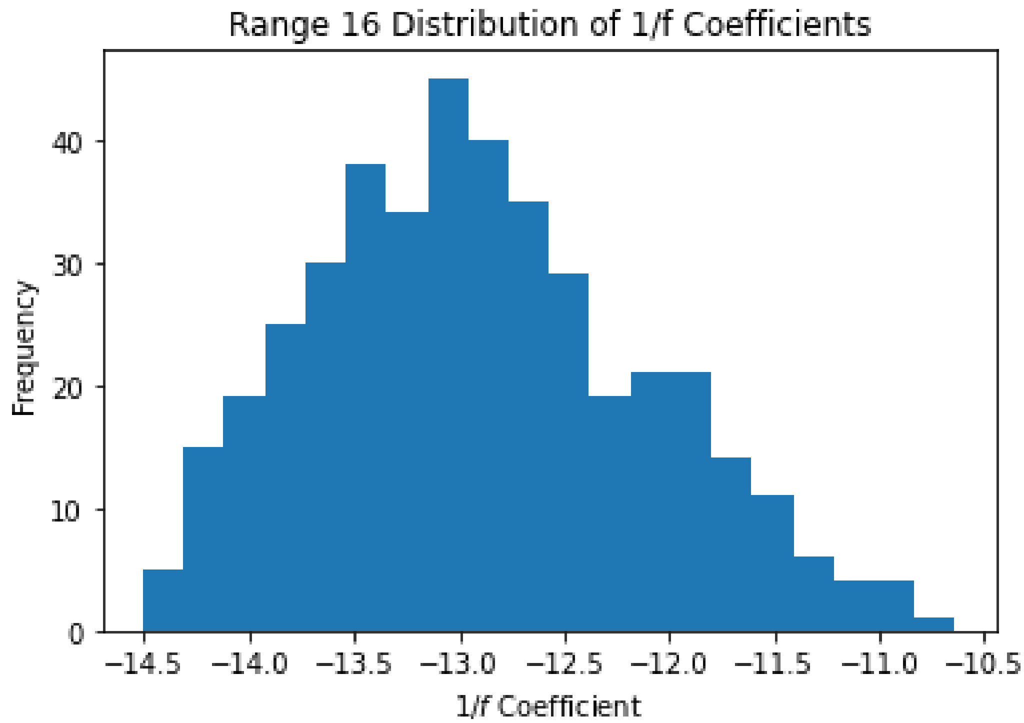

Probability Plot
Ordered Values
Theoretical quantiles
−10
−11
−12
−13
−14
−15
−3
−2
−1
0
1
2
3

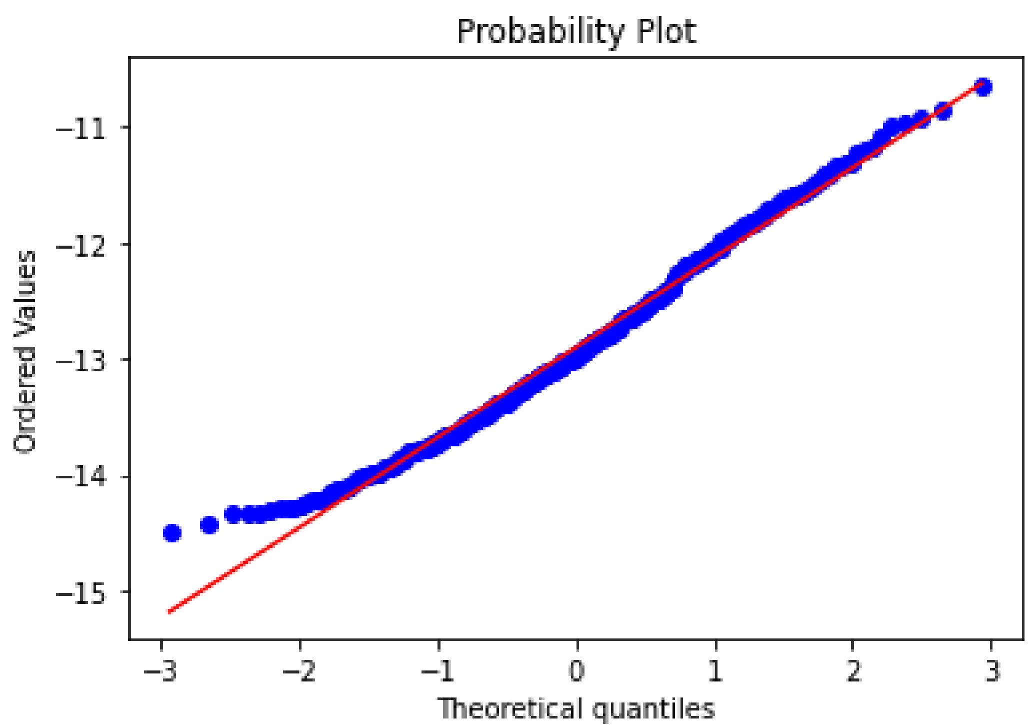

Range 12 Distribution of 1/f Coefficients
Frequency
0
5
10
15
20
25
30
35
40
−14
−13
−12
−11
1/f Coefficient

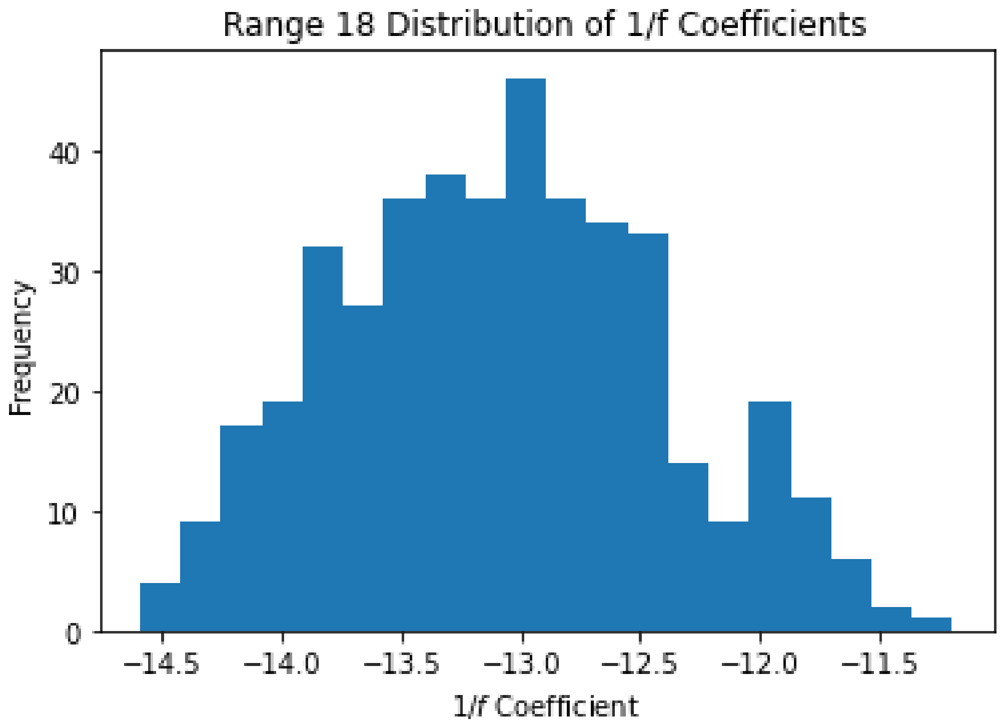

Probability Plot
Ordered Values
−11
−12
−13
−14
−15
−3
−2
−1
0
1
2
3
Theoretical quantiles

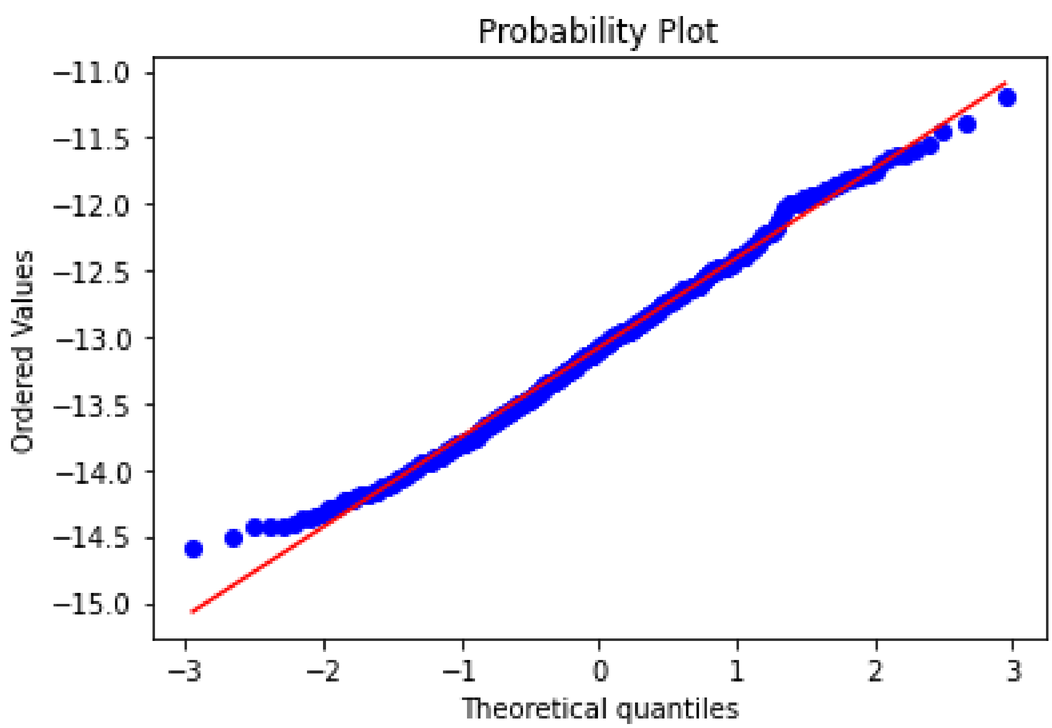

Range 14 Distribution of 1/f Coefficients
Frequency
1/f Coefficient
0
10
20
30
40
50
−14
−13
−12
−11

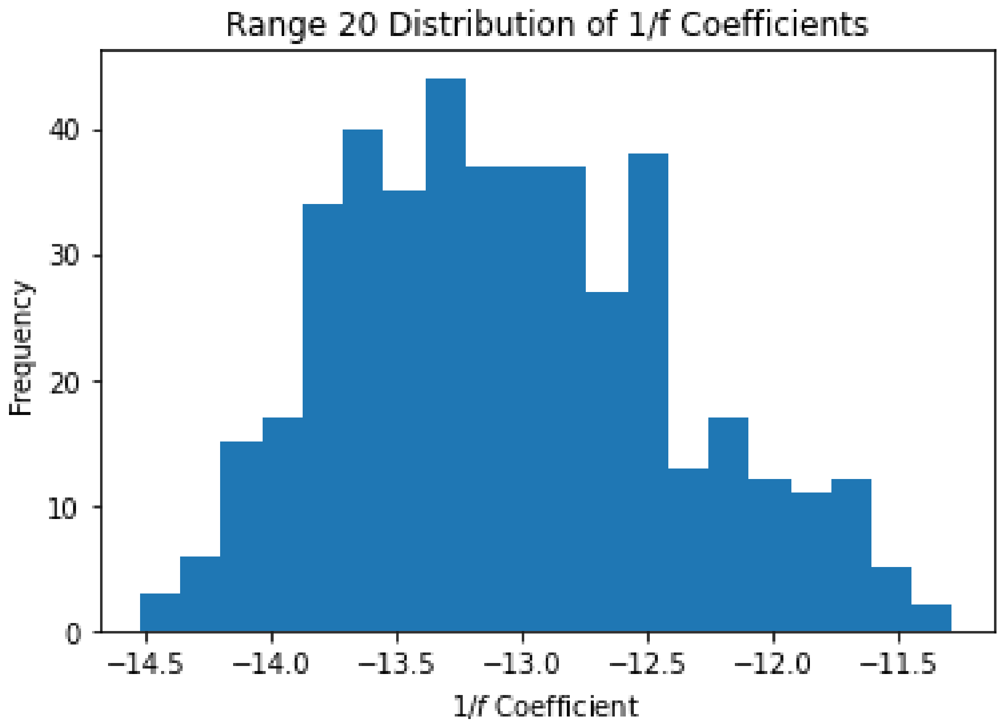

Probability Plot
Ordered Values
Theoretical quantiles
−11
−12
−13
−14
−15
−3
−2
−1
0
1
2
3

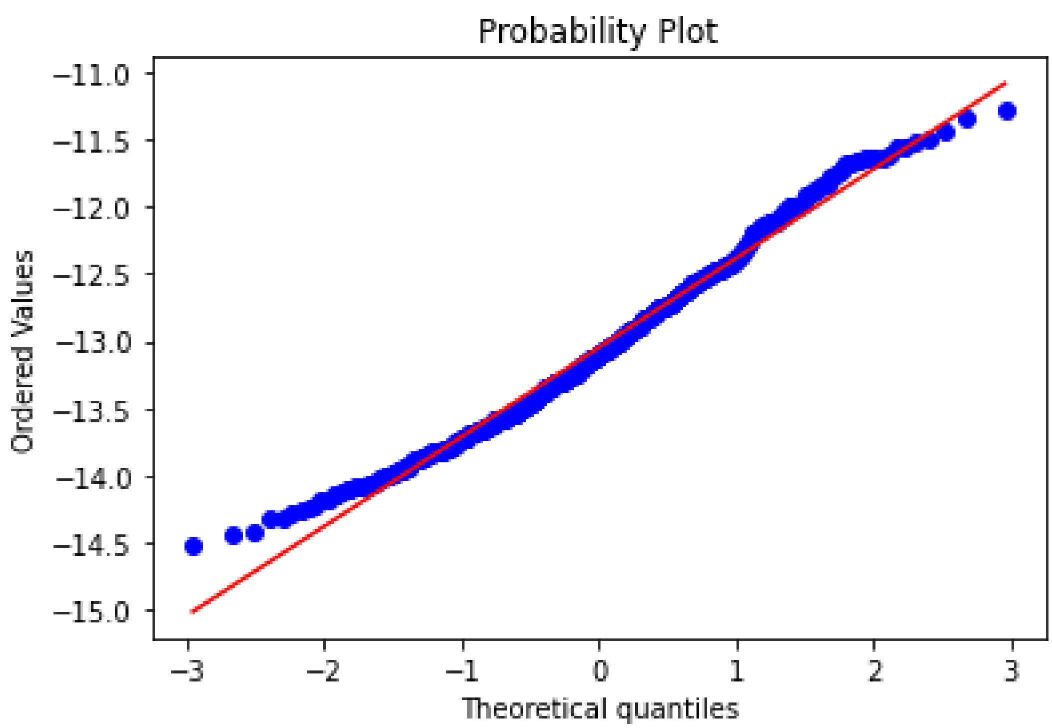

Range 16 Distribution of 1/f Coefficients
Frequency
1/f Coefficient
0
10
20
30
40
−14.5
−14.0
−13.5
−13.0
−12.5
−12.0
−11.5
−11.0
−10.5

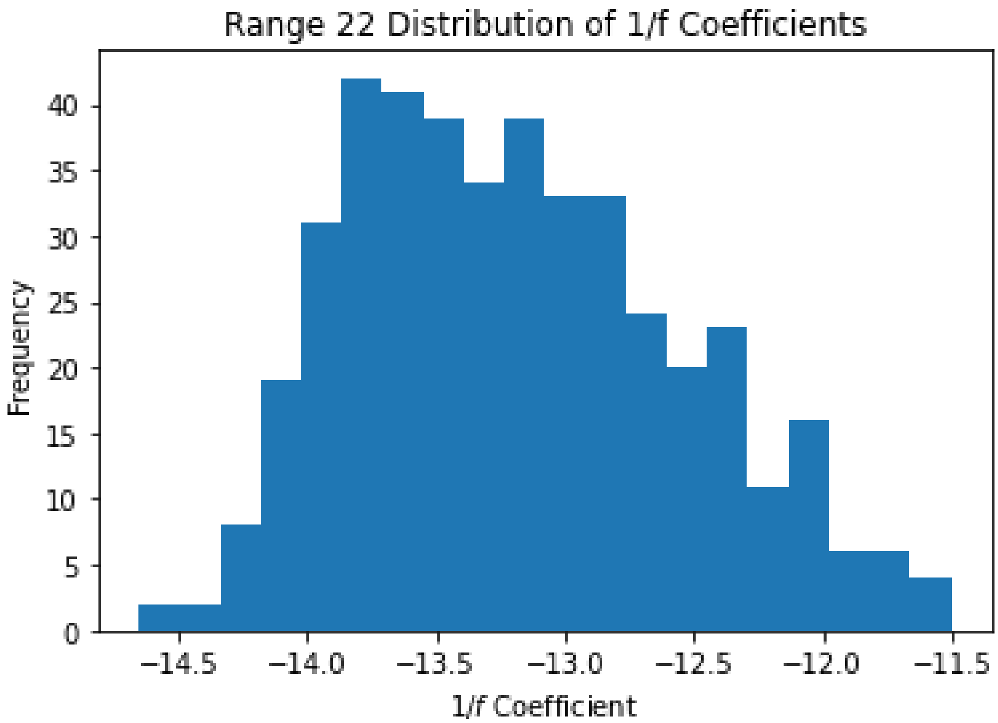

Probability Plot
Ordered Values
Theoretical quantiles
−11
−12
−13
−14
−15
−3
−2
−1
0
1
2
3

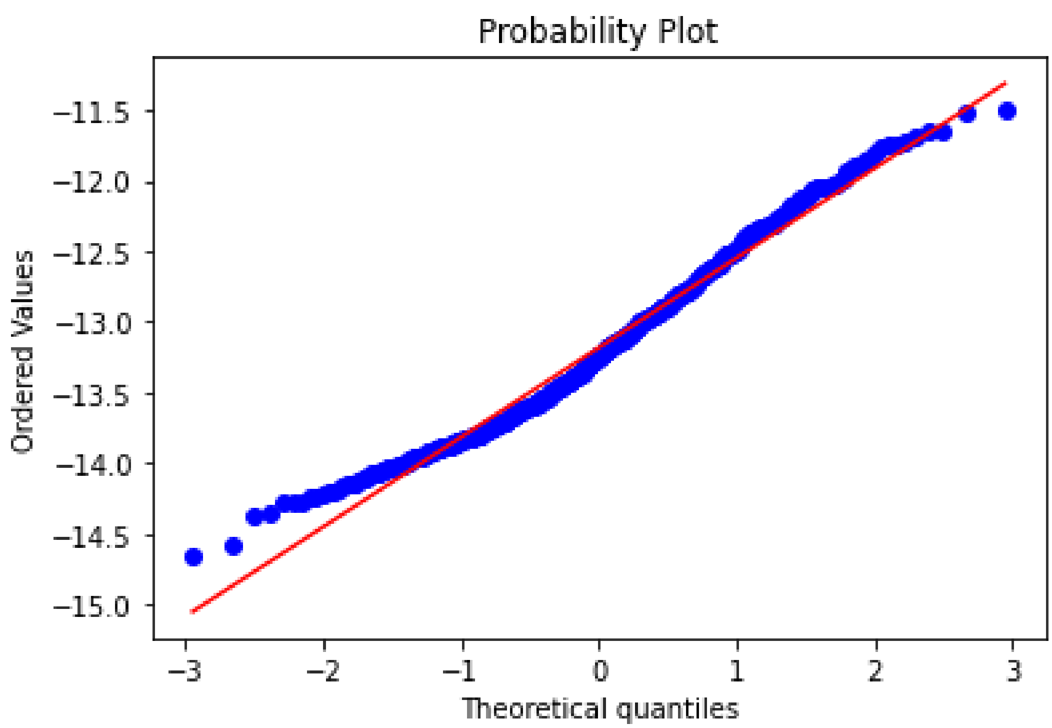

Range 18 Distribution of 1/f Coefficients
Frequency
0
10
20
30
40
−14.5
−14.0
−13.5
−13.0
−12.5
−12.0
−11.5
1/f Coefficient

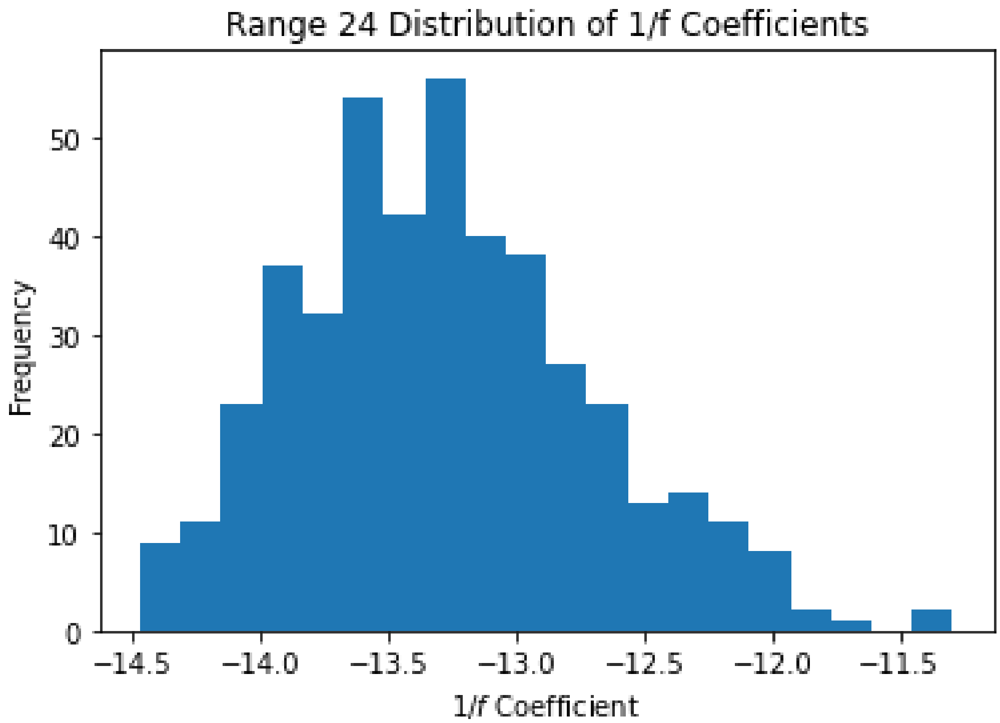

Probability Plot
Ordered Values
−11.0
−11.5
−12.0
−12.5
−13.0
−13.5
−14.0
−14.5
−15.0
−3
−2
−1
0
1
2
3
Theoretical quantiles

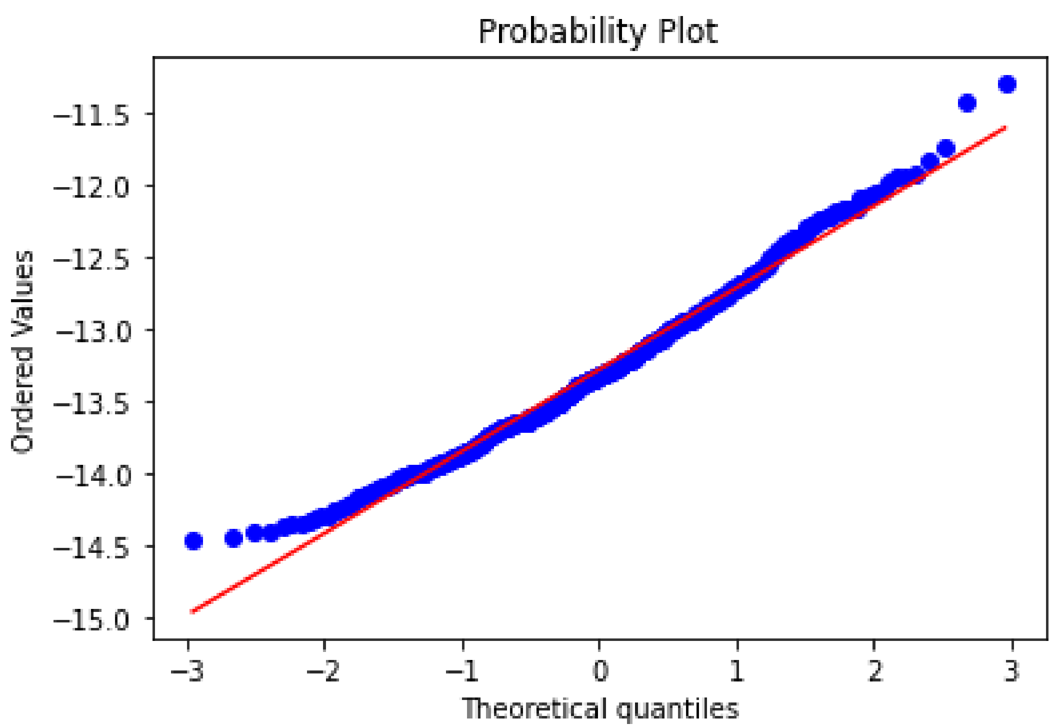

Range 20 Distribution of 1/f Coefficients
Frequency
0
10
20
30
40
−14.5
−14.0
−13.5
−13.0
−12.5
−12.0
−11.5
1/f Coefficient

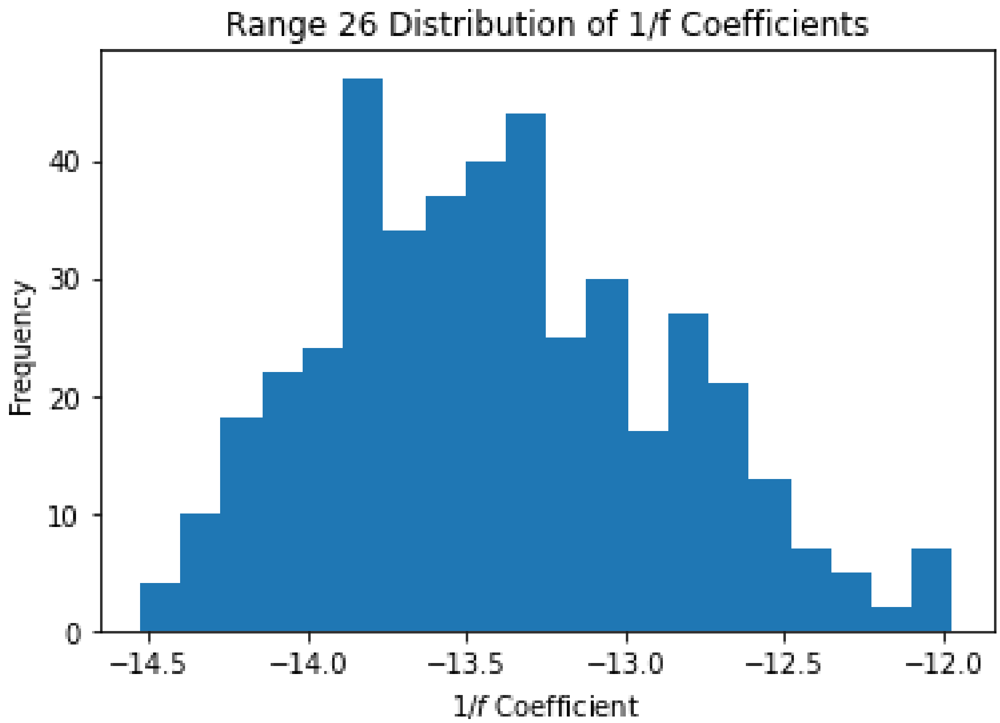

Probability Plot
Ordered Values
−11.0
−11.5
−12.0
−12.5
−13.0
−13.5
−14.0
−14.5
−15.0
−3
−2
−1
0
1
2
3
Theoretical quantiles

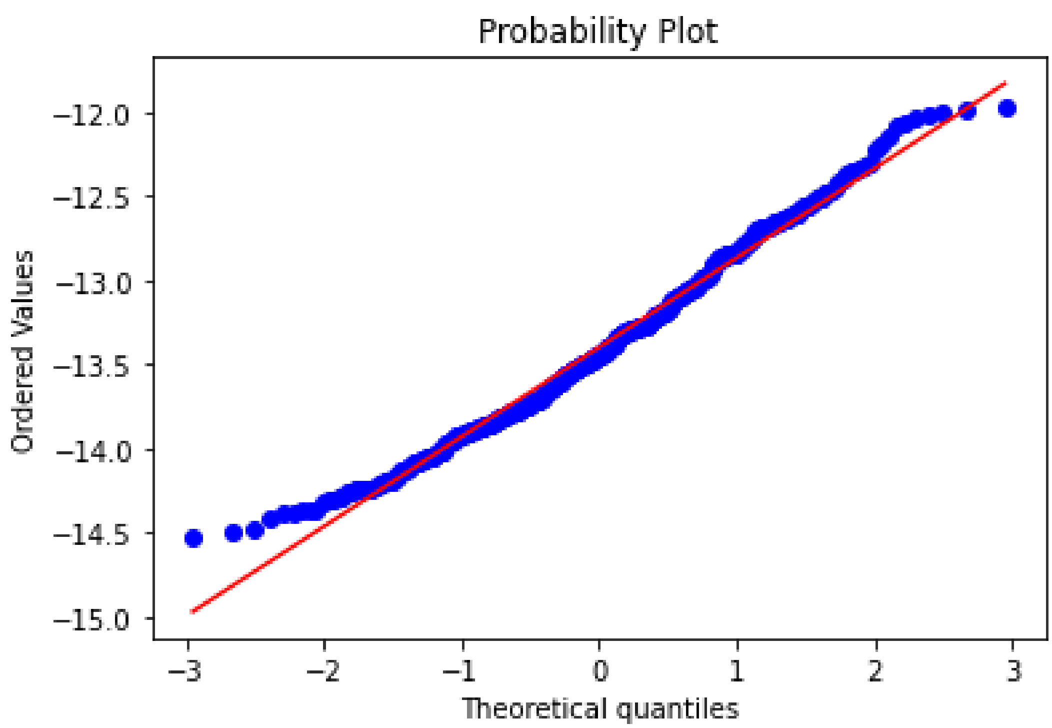

Range 22 Distribution of 1/f Coefficients
Frequency
0
5
10
15
20
25
30
35
40
−14.5
−14.0
−13.5
−13.0
−12.5
−12.0
−11.5
1/f Coefficient

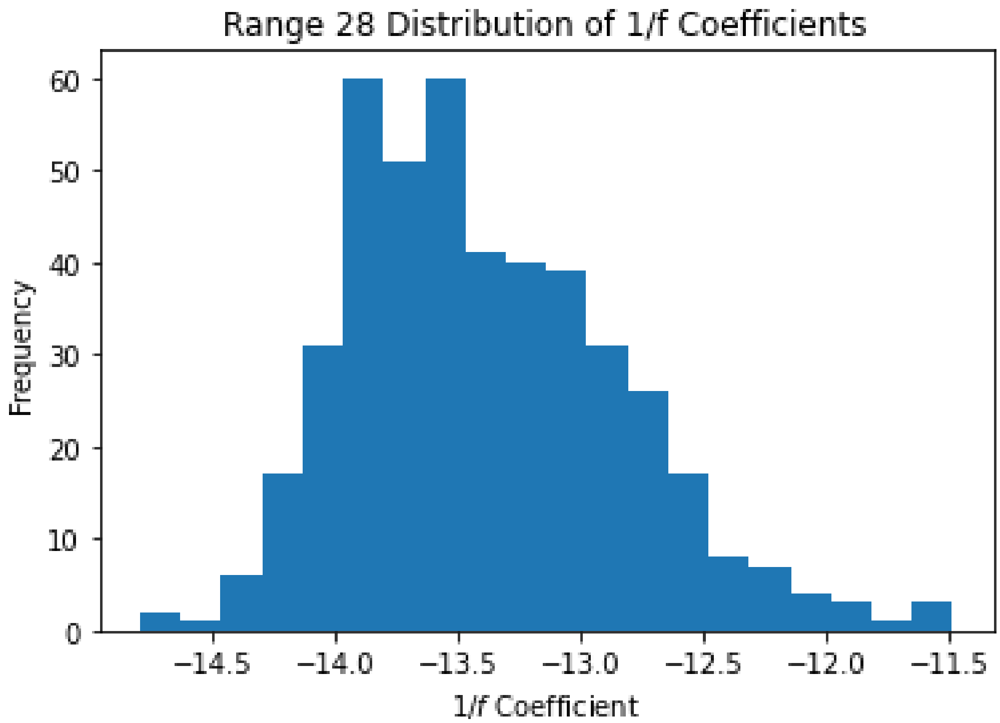

Probability Plot
Ordered Values
−11.5
−12.0
−12.5
−13.0
−13.5
−14.0
−14.5
−15.0
−3
−2
−1
0
1
2
3
Theoretical quantiles

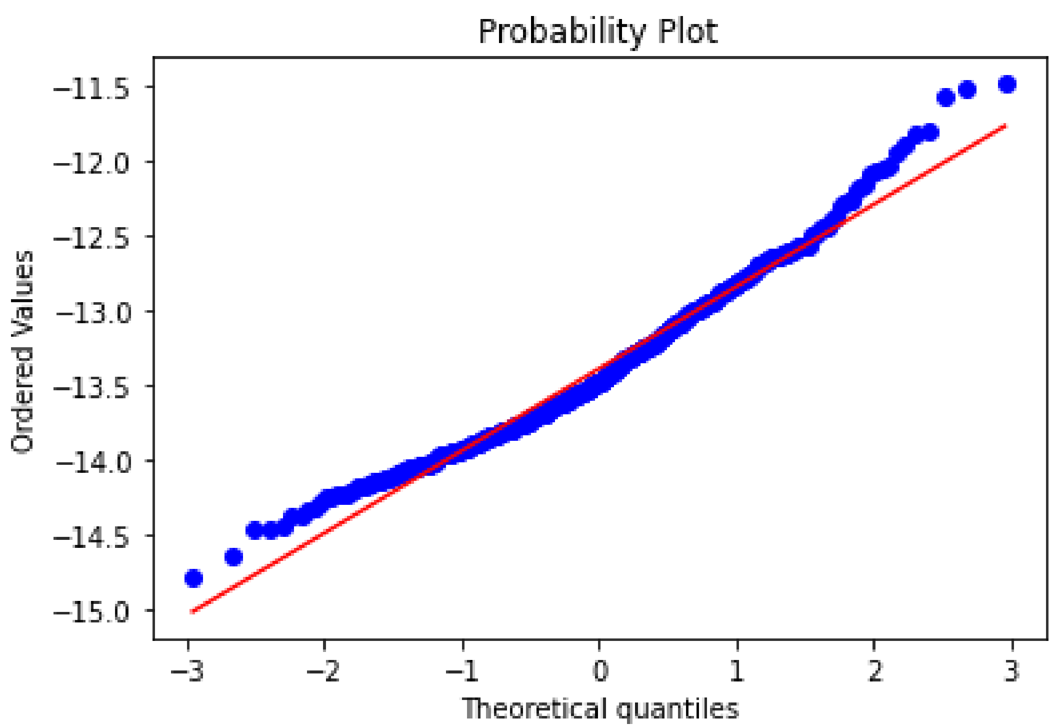

Range 24 Distribution of 1/f Coefficients
Frequency
1/f Coefficient
0
10
20
30
40
50
−14.5
−14.0
−13.5
−13.0
−12.5
−12.0
−11.5

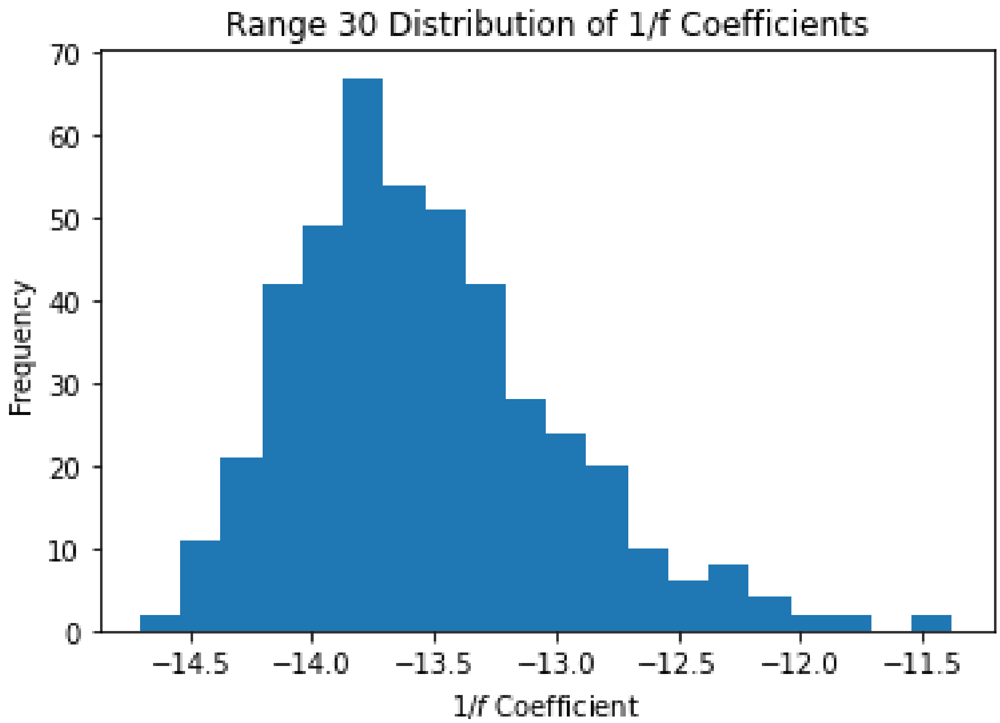

Probability Plot
Ordered Values
Theoretical quantiles
−11.5
−12.0
−12.5
−13.0
−13.5
−14.0
−14.5
−15.0
−3
−2
−1
0
1
2
3

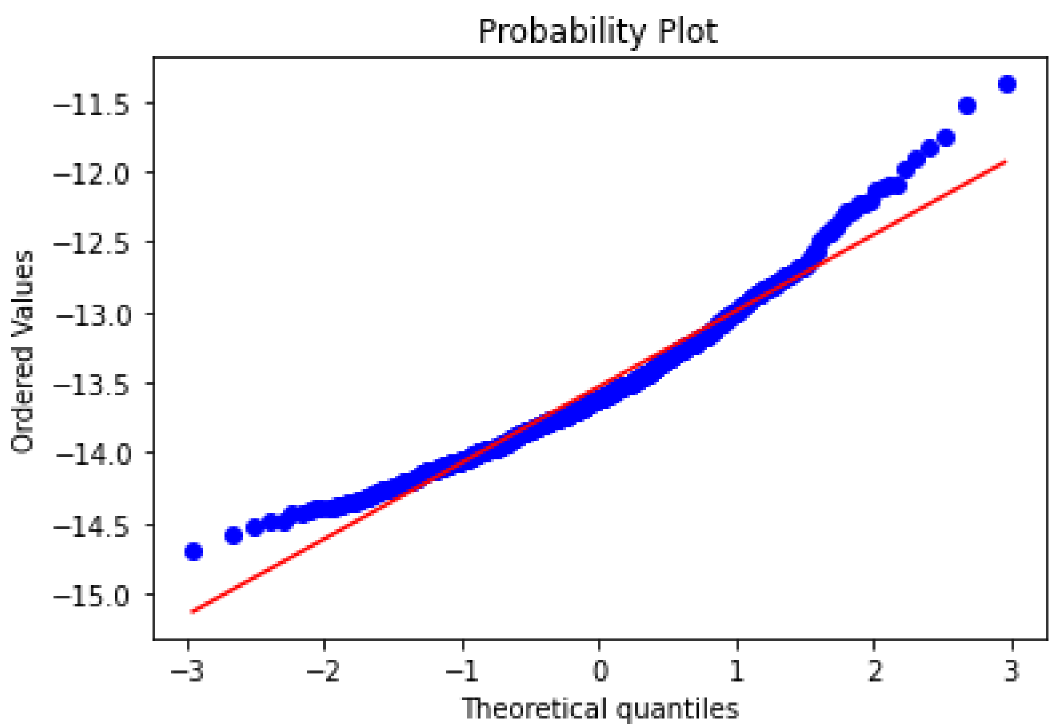

Range 26 Distribution of 1/f Coefficients
Frequency
1/f Coefficient
40
30
20
10
0
−14.5
−14.0
−13.5
−13.0
−12.5
−12.0

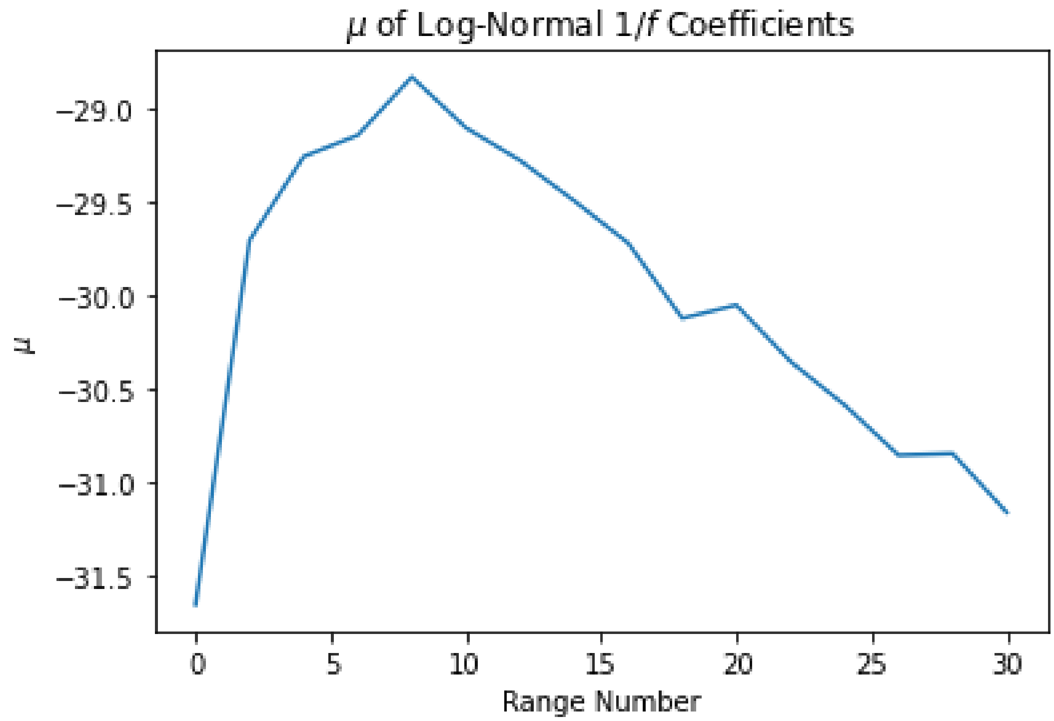

Probability Plot
Ordered Values
Theoretical quantiles
−12.0
−12.5
−13.0
−13.5
−14.0
−14.5
−15.0
−3
−2
−1
0
1
2
3

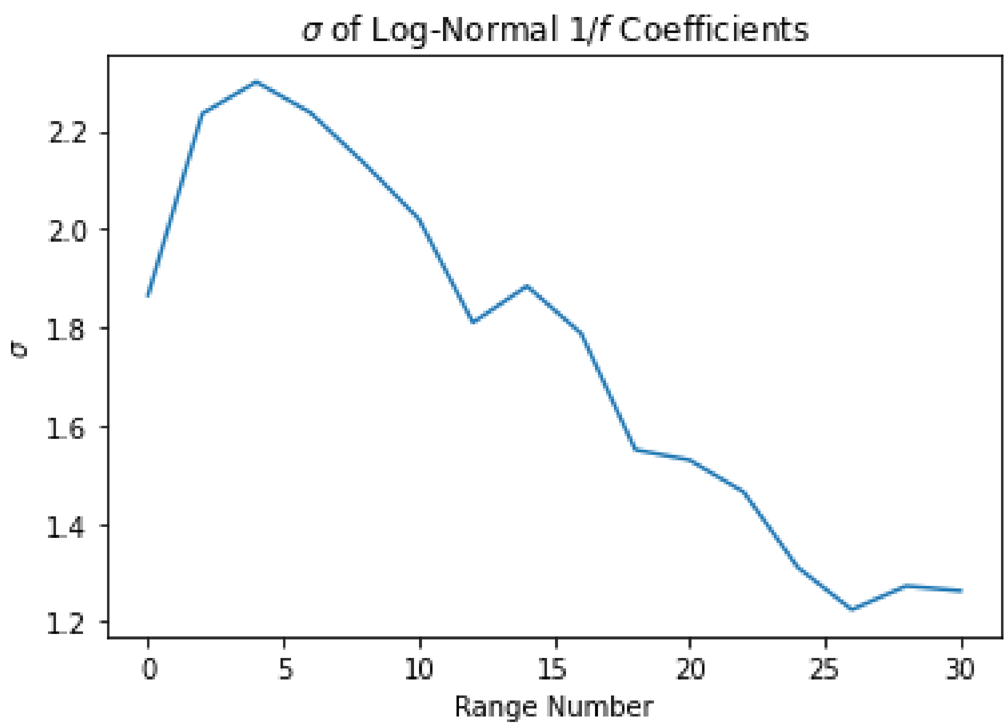

Range 28 Distribution of 1/f Coefficients
Frequency
0
10
20
30
40
50
60
−14.5
−14.0
−13.5
−13.0
−12.5
−12.0
−11.5
1/f Coefficient


Probability Plot
Ordered Values
−11.5
−12.0
−12.5
−13.0
−13.5
−14.0
−14.5
−15.0
−3
−2
−1
0
1
2
3
Theoretical quantiles


Range 30 Distribution of 1/f Coefficients
Frequency
70
60
50
40
30
20
10
0
−14.5
−14.0
−13.5
−13.0
−12.5
−12.0
−11.5
1/f Coefficient


Probability Plot
Ordered Values
−11.5
−12.0
−12.5
−13.0
−13.5
−14.0
−14.5
−15.0
−3
−2
−1
0
1
2
3
Theoretical quantiles


μ of Log-Normal $1/f$ Coefficients
−29.0
−29.5
−30.0
−30.5
−31.0
−31.5
μ
0
5
10
15
20
25
30
Range Number


σ of Log-Normal $1/f$ Coefficients
2.2
2.0
1.8
1.6
1.4
1.2
σ
0
5
10
15
20
25
30
Range Number