ECEN321 : Noise
Lab 2 Submission

Daniel Eisen : 300447549

May 11, 2020

1 Introduction

Noise, when pertaining to taking measurements, represents an uncertainty of the true value of the measured parameter. When these values are further used, i.e functions are applied to said measurements this error propagates. Hence the necessity in characterising and possibly mitigating the noise and/or its effect. This lab investigates various angles of error propagation and mitigation.

2 Theory

1. Amplification of the (noisy) DC signal represents a linear function of that measurement. Thus the mean and uncertainty of the that functions result can be estimated.

$$U = aX + b = 10X + 0$$

$$\text{mean} : \mu_U = a\mu_X + b = 10 \cdot 3 = 30$$

$$\sigma_U = |a|\sigma_X = |10| \cdot 2 = 20$$

2. To increase precision, multiple measurements can be taken and averaged. The basis of why this works is that the mean for repeated (assumed identical) measurements remains the same where the variance decreases by a factor of $\frac{1}{N}$:

$$\mu_{\frac{1}{N}(X_1+\ldots+X_N)} = \frac{1}{N}E\{X_1 + \ldots + X_N\} = \frac{1}{N}(\mu_X + \ldots + \mu_X) = \frac{N}{N}\mu_X = \mu_X$$

$$\sigma_{\frac{1}{N}(X_1+\ldots+X_N)} = \frac{1}{N^2}N\sigma_X^2 = \frac{1}{N}\sigma_X^2$$

   Therefore, for 16 repeated measurements, the various with decrease by 1/16, or the std by 1/4. i.e $\sigma = 2 \rightarrow \sigma = 0.5$

3. Covariance is a measure of how two variables change together, but its magnitude is unbounded, so it is difficult to interpret.

$$Cov(X, Y) = \mu_{XY} - \mu_X\mu_Y$$

   By dividing covariance by the product of the two standard deviations, one can calculate the normalized version of the statistic. This is the correlation coefficient.

$$\rho_{XY} = \frac{Cov(X, Y)}{\sigma_X\sigma_Y}$$

   The correlation coefficient is a statistical measure of the strength of the relationship between the relative movements of two variables, range[-1.0 1.0].

4. For the linear combination of two variable,eg in the case $X \pm Y$, the combination of their errors can be written as:

$$\sigma_{X+Y} = \sqrt{\sigma_X^2 + \sigma_Y^2}$$

   Note: The convolution method states that for 2+ errors, the distribution for the combined error can be obtained via convolution.

3 Results

1. Amplified Noise

```python
mean_noise = 0
std_noise = 2
DC = 3

noise = mean_noise + std_noise * (np.random.randn(1000))
sig = DC + noise
print(f"Signal Sample: mean={np.mean(sig)}, std={np.std(sig)}")

amped = 10 * sig
print(f"Amplifier Sample: mean={np.mean(amped)}, std={np.std(amped)}")
```

```
>>> q1_amp()
Signal Sample: mean=2.9693316094834987, std=1.945355127360896
Amplifier Sample: mean=29.693316094834984, std=19.45355127360896
```

Result of the included python codes amplifier simulation. These can be seen to confirm the estimated mean and std in section 2.1, with noise being subject to the same scaling (of the function) as the "true" value.

2. Averaged Measurements

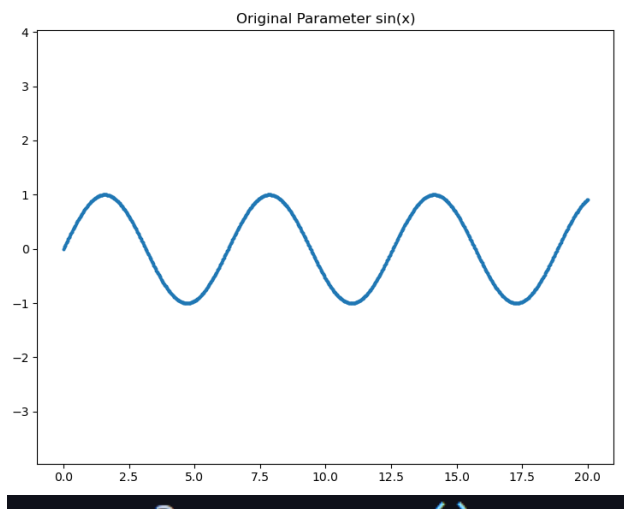
```python
mean_noise = 0
std_noise = 2
DC = 3

noise = mean_noise + std_noise * (np.random.randn(16, 1000))
sigs = DC + noise

print(f"Signal: mean={np.mean(sigs[0])}, std={np.std(sigs[0])}")
normed = np.sum(sigs, axis=0) / 16
print(f"Averaged Signal: mean={np.mean(normed)}, std={np.std(normed)}")
```

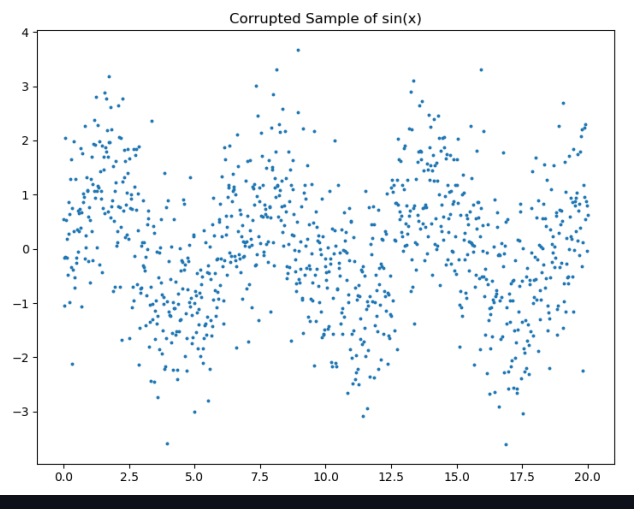
```
>>> q2_avg()
Signal: mean=2.8812649845429417, std=2.0473458742648933
Averaged Signal: mean=2.980104185477144, std=0.4957774846945985
```

Results of script confirm section 2.2. Showing preserved mean with a std reduction of $1/4$, 2 to 0.5. The results of the average would be (as shown in section 2) improved by a greater N of trials.

3. Covariance and Correlation

```python
x = np.linspace(0, 20, 1000)
sin = np.sin(x)
noise = np.random.randn(1000)
sin_noisy = sin + noise

print("Original Signal vs Noise Corrupted Measurement\n")
print(f"Covariance: {np.cov(sin, sin_noisy)[1, 0]}")
print(f"Correlation Coefficient: {np.corrcoef(sin, sin_noisy)[1, 0]}\n")

ax1 = plt.subplot(1, 2, 1)
plt.scatter(x, sin, linewidths=0, s=8)
plt.title("Original Parameter sin(x)")
plt.subplot(1, 2, 2, sharey=ax1)
plt.scatter(x, sin_noisy, linewidths=0, s=8)
plt.title("Corrupted Sample of sin(x)")
```



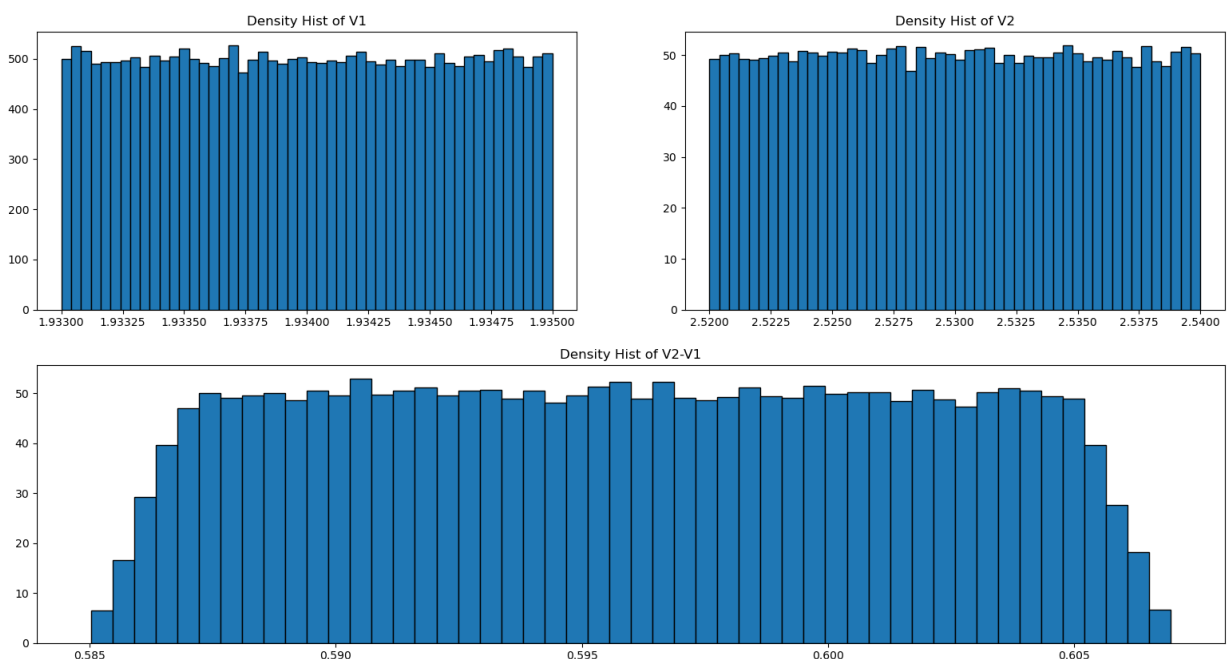



```
>>> q3_cov_corr()
Original Signal vs Noise Corrupted Measurement


Covariance: 0.5094766831270995
Correlation Coefficient: 0.589453683501512
```

Corrupting the original signal with noise to this degree can be see to visually distort the signal to quite a large degree. This breakdown in the samples relationship to its original signal is confirmed be the given calculation of the the correlation coefficient.

4. Combined Uncertainty

```python
V1_dc = 1.934
std1 = 0.001

V2_dc = 2.53
std2 = 0.01

V1 = V1_dc + (np.random.uniform(low=-std1, high=std1, size=100000))
V2 = V2_dc + (np.random.uniform(low=-std2, high=std2, size=100000))
V = V2 - V1

print(f"V2-V2 = {V2_dc - V1_dc}+-{np.sqrt(std2 ** 2 + std1 ** 2)}")

plt.subplot(2, 2, 1)
plt.hist(V1, bins=50, density=True, edgecolor="black")
plt.title("Density Hist of V1")

plt.subplot(2, 2, 2)
plt.hist(V2, bins=50, density=True, edgecolor="black")
plt.title("Density Hist of V2")

plt.subplot(2, 2, (3, 4))
plt.hist(V, bins=50, density=True, edgecolor="black")
plt.title("Density Hist of V2-V1")
```




```
>>> q4_combined_uncertainties()
V2-V2 = 0.5959999999999999+-0.010049875621120889
```

Numpy's histogram function was used to produce the above density plots of the V1, V2 and V2-V1. The plotted estimate of V2-V1's pdf corroborates with the outputs calculation of 0.596 ± 0.01. It shape is also consistent with the convolution method referenced in section 2.4, i.e the convolution of 2 square pulses resulting in a trapezium.

Appendix

Full Python Script

```
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats


def q1_amp():
    mean_noise = 0
    std_noise = 2
    DC = 3

    noise = mean_noise + std_noise * (np.random.randn(1000))
    sig = DC + noise
    print(f"Signal Sample: mean={np.mean(sig)}, std={np.std(sig)}")

    amped = 10 * sig
    print(f"Amplifier Sample: mean={np.mean(amped)}, std={np.std(amped)}")


def q2_avg():
    mean_noise = 0
    std_noise = 2
    DC = 3

    noise = mean_noise + std_noise * (np.random.randn(16, 1000))
    sigs = DC + noise

    print(f"Signal: mean={np.mean(sigs[0])}, std={np.std(sigs[0])}")
    normed = np.sum(sigs, axis=0) / 16
    print(f"Averaged Signal: mean={np.mean(normed)}, std={np.std(normed)}")


def q3_cov_corr():
    x = np.linspace(0, 20, 1000)
    sin = np.sin(x)
    noise = np.random.randn(1000)
    sin_noisy = sin + noise

    print("Original Signal vs Noise Corrupted Measurement\n")
    print(f"Covariance: {np.cov(sin, sin_noisy)[1, 0]}")
    print(f"Correlation Coefficient: {np.corrcoef(sin, sin_noisy)[1, 0]}\n")

    ax1 = plt.subplot(1, 2, 1)
    plt.scatter(x, sin, linewidths=0, s=8)
    plt.title("Original Parameter sin(x)")
    plt.subplot(1, 2, 2, sharey=ax1)
    plt.scatter(x, sin_noisy, linewidths=0, s=8)
    plt.title("Corrupted Sample of sin(x)")


def q4_combined_uncertainties():
    V1_dc = 1.934
    std1 = 0.001

    V2_dc = 2.53
    std2 = 0.01

    V1 = V1_dc + (np.random.uniform(low=-std1, high=std1, size=100000))
    V2 = V2_dc + (np.random.uniform(low=-std2, high=std2, size=100000))
    V = V2 - V1

    print(f"V2-V2 = {V2_dc - V1_dc}+-{np.sqrt(std2 ** 2 + std1 ** 2)}")

    plt.subplot(2, 2, 1)
    plt.hist(V1, bins=50, density=True, edgecolor="black")
    plt.title("Density Hist of V1")

    plt.subplot(2, 2, 2)
    plt.hist(V2, bins=50, density=True, edgecolor="black")
    plt.title("Density Hist of V2")

    plt.subplot(2, 2, (3, 4))
    plt.hist(V, bins=50, density=True, edgecolor="black")
    plt.title("Density Hist of V2-V1")
```