REPORT

Classes: Analog and Digital Electronic Circuits

Presenter: prof. dr hab. inż. Vasyl Martsenyuk

Laboratorium No. 3 Date: 02.12.2023 Topic: „Random Signals" Version 6	Adam Kubliński Informatyka II stopień, niestacjonarne, zaoczne, I semestr, gr. 1A

GitHub Repository:
https://github.com/Adamadacho/Analog_and_Digital_Electronic_Circuits.git

1. Topic of the laboratory

The topic of this exercise is "Statistical analysis of random signals". The exercise focuses on generating, analyzing and visualizing random signals, using statistical methods to study their properties.

2. Task

Generate ensemble of random signals of the form xn(k) = Acos(2f π /k)+ BWn(k)1, where Wn(k) is normally distributed in [0,1] numbers, A, f,B are determined in the table below.

1. Estimate the linear mean as ensemble average
2. Estimate the linear mean and squared linear mean
3. Estimate the quadratic mean and variance.
4. Plot 1-4 graphically.
5. Estimate and plot the auto-correlation function (ACF)

No	f	A	B	N
1	300	300.25	299.75	2000
2	400	400.25	399.75	3000
3	500	500.25	499.75	1800
4	600	600.25	599.75	2000
5	300	300.25	299.75	2000
6	600	600.25	599.75	2000
7	400	400.25	399.75	3000
8	500	500.25	499.75	1800
9	600	600.25	599.75	2000
10	300	300.25	299.75	2000
11	200	200.25	199.75	2000
12	400	400.25	399.75	3000
13	500	500.25	499.75	1800
14	600	600.25	599.75	2000
15	500	500.25	499.75	2000

Table 1: Variants

The code has been done according to instruction and has been adjusted to the variant 6. It was made in Jupyther Notebook.

```
k = np.arange(1, N+1)
ensemble = A * np.cos(2 * np.pi * f / k) + B * np.random.normal(0, 1, N)

linear_mean = np.mean(ensemble)
linear_mean_squared = linear_mean ** 2
quadratic_mean = np.mean(ensemble ** 2)
variance = np.var(ensemble)

print("Linear mean =", linear_mean)
print("Linear mean squared =",linear_mean_squared)
print("Quadratic mean =",quadratic_mean)
print("Variance =",variance)

plt.figure(figsize=(15, 10))

plt.subplot(2, 2, 1)
plt.plot(k, ensemble, label="Random signal")
plt.title("Random signal")
plt.xlabel("k")
plt.ylabel("x_n(k)")
plt.legend()

plt.subplot(2, 2, 2)
plt.axhline(y=linear_mean, color='r', linestyle='-', label="Linear mean")
plt.axhline(y=linear_mean_squared, color='g', linestyle='-', label="Linear
mean squared")
plt.axhline(y=quadratic_mean, color='b', linestyle='-', label="Quadratic
mean")
plt.axhline(y=variance, color='y', linestyle='-', label="Variance")
plt.title("Means and Variance")
plt.xlabel("k")
plt.legend()

acf = np.correlate(ensemble - linear_mean, ensemble - linear_mean,
mode='full') / N
acf = acf[N-1:]

plt.subplot(2, 2, 3)
plt.plot(acf, label="ACF")
plt.title("Autocorrelation Function (ACF)")
plt.xlabel("Delay")
plt.ylabel("ACF")
plt.legend()
plt.tight_layout()
plt.show()
```

3. Conclusions

A set of random signals was simulated with a specific mathematical formula combining a deterministic element (cosine function) and a random element (Gaussian noise). Statistical calculations such as linear mean, linear mean squared, mean squared, and variance were performed to understand the characteristics of the signal set. Graphs were created showing both individual signals and calculated statistical values, which allowed for graphical interpretation of the results.

This exercise was a practical application of random signal theory, combining theoretical and computational elements, aimed at a deeper understanding of the nature and properties of random signals.