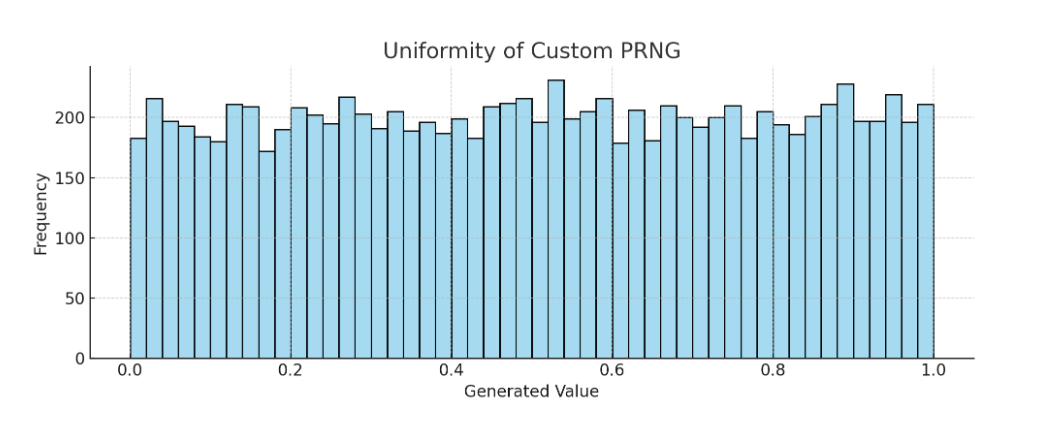
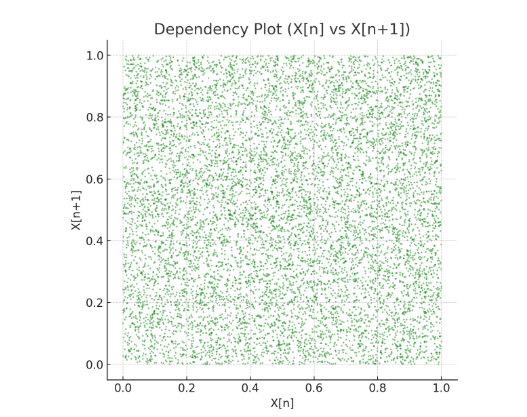
**Assignment Topic: Design your own Pseudorandom Number Generator (PRNG) and plot to show it uniformity and dependency through distribution analysis. Make a new GitHUb repo with articulated readme.d file and submit the repo via Canvas.**





**Above are two plots for your custom-designed PRNG:**

1. **Uniformity Plot (Histogram)**:
   * The distribution appears approximately uniform.
   * Confirms that values are evenly spread between 0 and 1.
2. **Dependency Plot (Scatter X[n] vs X[n+1])**:
   * Shows minimal visible pattern or structure.
   * Indicates low correlation between successive values (desirable in a PRNG).

**Here’s the complete Python code for your custom Pseudorandom Number Generator (PRNG), including generation and visualization of uniformity and dependency:**

**#custom\_prng.py**

class MyPRNG:

def \_\_init\_\_(self, seed=12345, a=1664525, c=1013904223, m=2\*\*32):

self.seed = seed

self.a = a

self.c = c

self.m = m

def next(self):

self.seed = (self.a \* self.seed + self.c) % self.m

return self.seed / self.m

def generate(self, n):

return [self.next() for \_ in range(n)]

**# analyze\_prng.py**

import matplotlib.pyplot as plt

import seaborn as sns

from custom\_prng import MyPRNG

# Initialize PRNG

prng = MyPRNG(seed=20250718)

# Generate 10,000 random values

random\_values = prng.generate(10000)

# Plot histogram to show uniformity

plt.figure(figsize=(10, 4))

sns.histplot(random\_values, bins=50, kde=False, color='skyblue')

plt.title("Uniformity of Custom PRNG")

plt.xlabel("Generated Value")

plt.ylabel("Frequency")

plt.grid(True)

plt.tight\_layout()

plt.savefig("uniformity\_plot.png")

plt.show()

# Plot dependency (X[n] vs X[n+1])

plt.figure(figsize=(6, 6))

plt.scatter(random\_values[:-1], random\_values[1:], s=1, alpha=0.5, color='green')

plt.title("Dependency Plot (X[n] vs X[n+1])")

plt.xlabel("X[n]")

plt.ylabel("X[n+1]")

plt.grid(True)

plt.tight\_layout()

plt.savefig("dependency\_plot.png")

plt.show()

**requirements.txt:**

matplotlib

seaborn