Q5

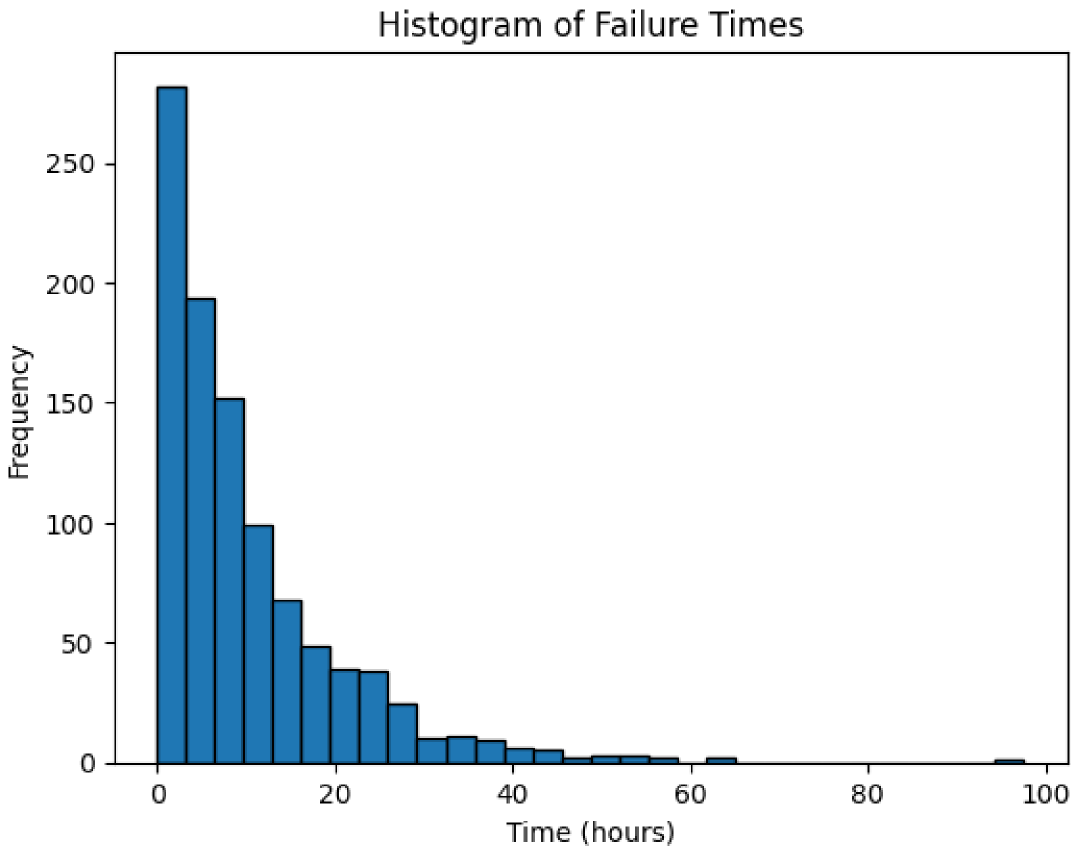
October 1, 2024

1 Exponential Distribution and Reliability

A machine part fails according to an exponential distribution with a mean time between failures of 10 hours. Let X represent the time (in hours) until the next failure.

3. (7 points): Suppose the machine part is replaced after each failure, and you track 1000 failure times. Simulate the total downtime over these 1000 failures, assuming each failure follows the exponential distribution with a mean of 10 hours. Plot the histogram of failure times and discuss the practical implications for managing the machine's operational efficiency.

[1]:
```python
import numpy as np
import matplotlib.pyplot as plt

# Parameters
mean_time_between_failures = 10
num_failures = 1000

# Simulate failure times
failure_times = np.random.exponential(mean_time_between_failures, num_failures)

# Calculate total downtime
total_downtime = np.sum(failure_times)

# Plot histogram of failure times
plt.hist(failure_times, bins=30, edgecolor='black')
plt.title('Histogram of Failure Times')
plt.xlabel('Time (hours)')
plt.ylabel('Frequency')
plt.show()

# Print total downtime
print(f'Total downtime over {num_failures} failures: {total_downtime} hours')
```




```
Total downtime over 1000 failures: 10266.544054280792 hours
```

1.0.1 Practical Implications for Managing the Machine's Operational Efficiency

The simulation of 1000 failure times, assuming each failure follows an exponential distribution with a mean of 10 hours, provides valuable insights into the machine's operational efficiency.

1. **Total Downtime**: The total downtime over 1000 failures was approximately 10266.54 hours. This metric is crucial for understanding the overall impact of failures on the machine's availability and productivity.

2. **Failure Time Distribution**: The histogram of failure times shows the frequency of failures over different time intervals. Most failures occur within a shorter time frame, with fewer failures occurring at longer intervals. This pattern is typical of an exponential distribution and indicates that while most failures happen relatively quickly, there is a long tail of less frequent, longer-duration failures.

3. **Maintenance Scheduling**: Understanding the distribution of failure times can help in planning maintenance schedules. Regular maintenance can be scheduled during periods of expected lower failure rates to minimize the impact on production.

4. **Spare Parts Inventory**: Knowing the mean time between failures and the distribution of failure times can help in optimizing the inventory of spare parts. Keeping an adequate stock

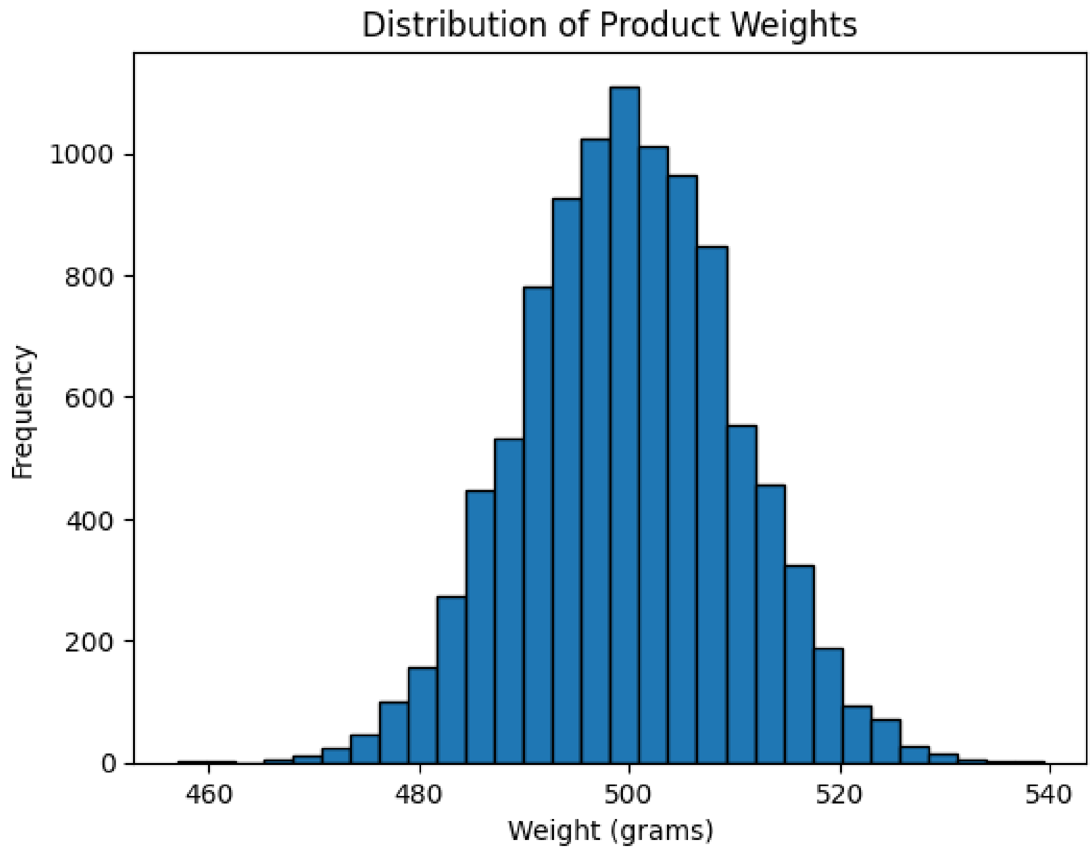
of frequently failing parts can reduce downtime caused by waiting for replacements.

2 NORMAL DISTRIBUTION AND PRODUCT QUALITY

In a factory, the weight of a certain product follows a normal distribution with a mean of 500 grams and a standard deviation of 10 grams.

(3 points) Simulate 10,000 product weights using the normal distribution with the given parameters. Plot the distribution and calculate the percentage of products that fall within the range of 495 to 505 grams. Compare this with the theoretical value and discuss any discrepancies.

[2]:
```python
# Parameters for the normal distribution
mean_weight = 500
std_dev_weight = 10
num_products = 10000

# Simulate product weights
product_weights = np.random.normal(mean_weight, std_dev_weight, num_products)

# Plot the distribution of product weights
plt.hist(product_weights, bins=30, edgecolor='black')
plt.title('Distribution of Product Weights')
plt.xlabel('Weight (grams)')
plt.ylabel('Frequency')
plt.show()

# Calculate the percentage of products within the range 495 to 505 grams
within_range = np.sum((product_weights >= 495) & (product_weights <= 505))
percentage_within_range = (within_range / num_products) * 100

# Print the percentage
print(f'Percentage of products within 495 to 505 grams: 
 ↪{percentage_within_range:.2f}%')
```




```
Percentage of products within 495 to 505 grams: 37.69%
```

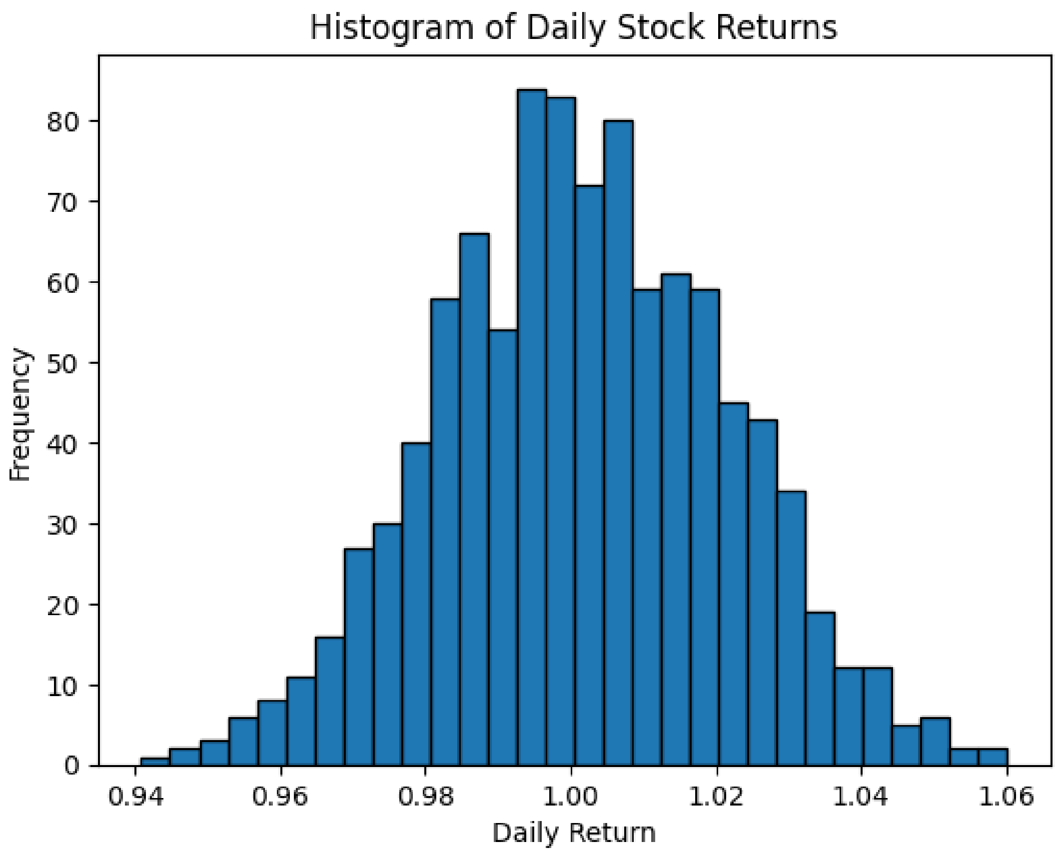
3 LOGNORMAL DISTRIBUTION AND STOCK RETURNS

The daily returns of a stock are modeled using a lognormal distribution with parameters = 0.001 and = 0.02.

3. (4 points) Simulate 1,000 days of stock returns using the lognormal distribution. Plot the histogram of daily returns and calculate the proportion of days where the return is positive. How does this relate to typical stock market behavior?

[3]:
```
# Parameters for the lognormal distribution
mu = 0.001 # mean of the lognormal distribution
sigma = 0.02 # standard deviation of the lognormal distribution
num_days = 1000 # number of days to simulate

# Simulate stock returns
stock_returns = np.random.lognormal(mean=mu, sigma=sigma, size=num_days)

# Plot the histogram of daily returns
```

```python
plt.hist(stock_returns, bins=30, edgecolor='black')
plt.title('Histogram of Daily Stock Returns')
plt.xlabel('Daily Return')
plt.ylabel('Frequency')
plt.show()

# Calculate the proportion of days where the return is positive
positive_returns = np.sum(stock_returns > 1)
proportion_positive = (positive_returns / num_days) * 100

# Print the proportion of positive returns
print(f'Proportion of days with positive returns: {proportion_positive:.2f}%')
```




```
Proportion of days with positive returns: 52.10%
```

In typical stock market behavior, it is common to observe that the majority of days have small positive returns, with occasional larger positive or negative returns. This simulation helps to understand the variability and skewness in stock returns, which are important factors for investors when assessing risk and potential returns.