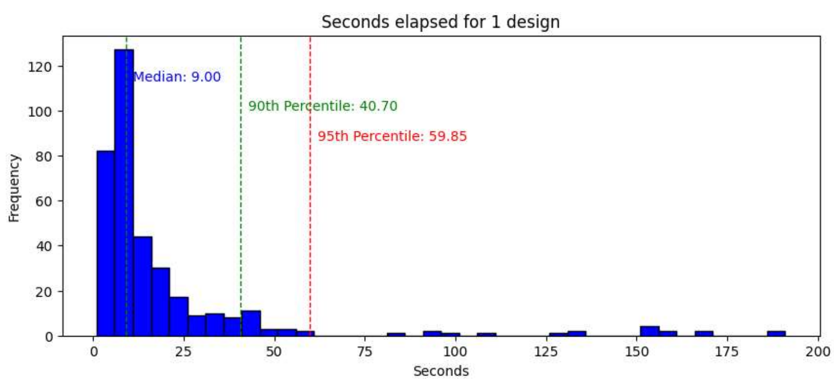
Plotting a histogram for the seconds elapsed for 1 design

```
import matplotlib.pyplot as plt
import numpy as np
plt.figure(figsize=(10,4))

seconds = [1, 1, 1, 1, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5

bin_width = 5 # Define the bin width (5 seconds in this case)

print(len(seconds))

# Calculate the number of bins
num_bins = int((max(seconds) - min(seconds)) / bin_width) + 1

# Create a histogram with custom bins
plt.hist(seconds, bins=num_bins, range=(min(seconds), max(seconds) + bin_width), color='blue', edgecolor='black')
# Calculate percentiles
median = np.percentile(seconds, 50)
percentile_90 = np.percentile(seconds, 90)
percentile_95 = np.percentile(seconds, 95)

# Add vertical markers for the 90th and 95th percentiles
plt.axvline(median, color='green', linestyle='dashed', linewidth=1, label='median')
plt.axvline(percentile_90, color='green', linestyle='dashed', linewidth=1, label='90th Percentile')
plt.axvline(percentile_95, color='red', linestyle='dashed', linewidth=1, label='95th Percentile')
# Add text labels for the percentile values
plt.text(median + 2, plt.ylim()[1] * 0.85, f'Median: {median:.2f}', color='blue', fontsize=10)
plt.text(percentile_90 + 2, plt.ylim()[1] * 0.75, f'90th Percentile: {percentile_90:.2f}', color='green', fontsize=10)
plt.text(percentile_95 + 2, plt.ylim()[1] * 0.65, f'95th Percentile: {percentile_95:.2f}', color='red', fontsize=10)

# Customize the plot
plt.title('Seconds elapsed for 1 design');
plt.xlabel('Seconds'); plt.ylabel('Frequency')

plt.show()# Show the plot
```

```
364
```

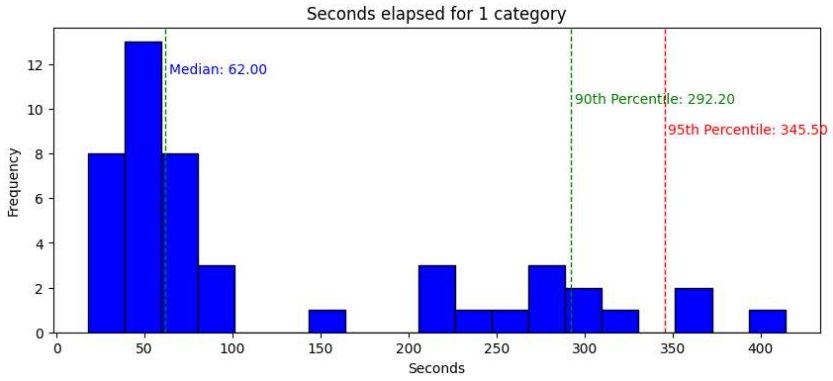


Plotting a histogram for the seconds elapsed for 1 category

```
import matplotlib.pyplot as plt
import numpy as np
plt.figure(figsize=(10,4))

seconds = [18, 19, 20, 21, 21, 22, 24, 24, 43, 44, 44, 46, 46, 47, 48, 49, 51, 55, 56, 56, 59, 61, 61, 62, 66, 66, 76, 78, 79, 93, 94, 95, 15

bin_width = 20 # Define the bin width (5 seconds in this case)

print(len(seconds))

# Calculate the number of bins
```

```
num_bins = int((max(seconds) - min(seconds)) / bin_width) + 1

# Create a histogram with custom bins
plt.hist(seconds, bins=num_bins, range=(min(seconds), max(seconds) + bin_width), color='blue', edgecolor='black')
# Calculate percentiles
median = np.percentile(seconds, 50)
percentile_90 = np.percentile(seconds, 90)
percentile_95 = np.percentile(seconds, 95)

# Add vertical markers for the 90th and 95th percentiles
plt.axvline(median, color='green', linestyle='dashed', linewidth=1, label='median')
plt.axvline(percentile_90, color='green', linestyle='dashed', linewidth=1, label='90th Percentile')
plt.axvline(percentile_95, color='red', linestyle='dashed', linewidth=1, label='95th Percentile')
# Add text labels for the percentile values
plt.text(median + 2, plt.ylim()[1] * 0.85, f'Median: {median:.2f}', color='blue', fontsize=10)
plt.text(percentile_90 + 2, plt.ylim()[1] * 0.75, f'90th Percentile: {percentile_90:.2f}', color='green', fontsize=10)
plt.text(percentile_95 + 2, plt.ylim()[1] * 0.65, f'95th Percentile: {percentile_95:.2f}', color='red', fontsize=10)

# Customize the plot
plt.title('Seconds elapsed for 1 category');
plt.xlabel('Seconds'); plt.ylabel('Frequency')

plt.show()# Show the plot
```

```
47
```

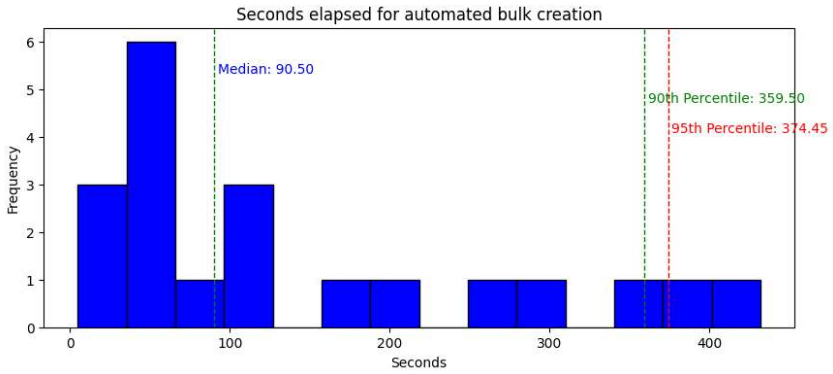


Plotting a histogram for the seconds elapsed for automated mug creation

```
import matplotlib.pyplot as plt
import numpy as np
plt.figure(figsize=(10,4))

seconds = [5, 25, 30, 51, 56, 60, 62, 63, 64, 77, 104, 105, 106, 185, 189, 277, 303, 358, 373, 402]

bin_width = 30 # Define the bin width (5 seconds in this case)

print(len(seconds))

# Calculate the number of bins
num_bins = int((max(seconds) - min(seconds)) / bin_width) + 1

# Create a histogram with custom bins
plt.hist(seconds, bins=num_bins, range=(min(seconds), max(seconds) + bin_width), color='blue', edgecolor='black')
# Calculate percentiles
median = np.percentile(seconds, 50)
percentile_90 = np.percentile(seconds, 90)
percentile_95 = np.percentile(seconds, 95)

# Add vertical markers for the 90th and 95th percentiles
plt.axvline(median, color='green', linestyle='dashed', linewidth=1, label='median')
plt.axvline(percentile_90, color='green', linestyle='dashed', linewidth=1, label='90th Percentile')
plt.axvline(percentile_95, color='red', linestyle='dashed', linewidth=1, label='95th Percentile')
# Add text labels for the percentile values
plt.text(median + 2, plt.ylim()[1] * 0.85, f'Median: {median:.2f}', color='blue', fontsize=10)
plt.text(percentile_90 + 2, plt.ylim()[1] * 0.75, f'90th Percentile: {percentile_90:.2f}', color='green', fontsize=10)
```

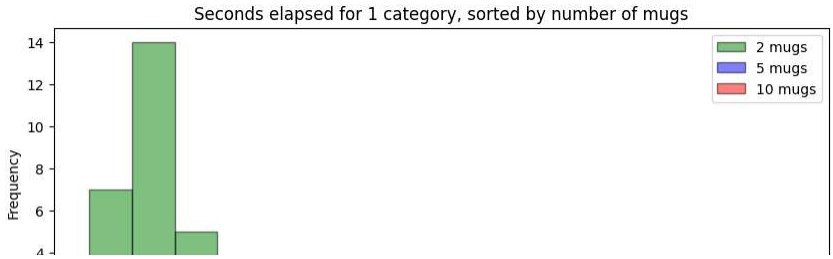
```
plt.text(percentile_95 + 2, plt.ylim()[1] * 0.65, f'95th Percentile: {percentile_95:.2f}', color='red', fontsize=10)

# Customize the plot
plt.title('Seconds elapsed for automated bulk creation');
plt.xlabel('Seconds'); plt.ylabel('Frequency')

plt.show()# Show the plot
```

```
20
```



Plotting a histogram for the seconds elapsed for 1 category, grouped by number of mugs per category

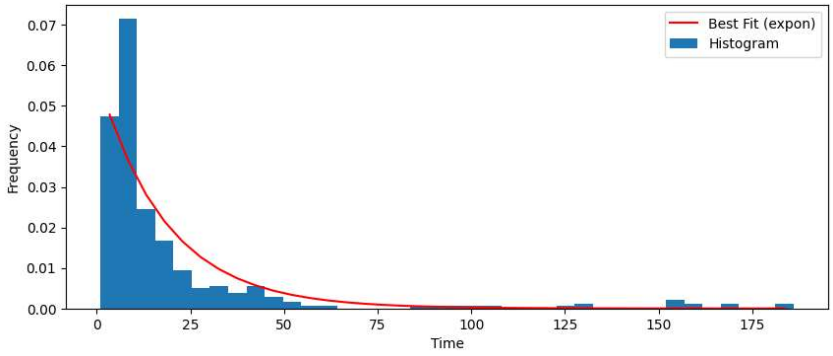
```
import matplotlib.pyplot as plt
import numpy as np

plt.figure(figsize=(10,4))
bin_width = 20 # Define the bin width (5 seconds in this case)

seconds1 = [18, 19, 20, 21, 21, 22, 24, 43, 44, 44, 46, 46, 47, 48, 49, 51, 55, 56, 61, 61, 62, 66, 66, 76, 78, 79, 93, 94]
# Calculate the number of bins
num_bins = int((max(seconds1) - min(seconds1)) / bin_width) + 1
# Create a histogram with custom bins
plt.hist(seconds1, bins=num_bins, range=(min(seconds1), max(seconds1) + bin_width), color='green', edgecolor='black', alpha=0.5, label='2 mug

seconds2 = [24, 56, 59, 95, 156, 222, 271, 275, 289, 328, 353, 394]

# Calculate the number of bins
num_bins = int((max(seconds2) - min(seconds2)) / bin_width) + 1
# Create a histogram with custom bins
plt.hist(seconds2, bins=num_bins, range=(min(seconds2), max(seconds2) + bin_width), color='blue', edgecolor='black', alpha=0.5, label='5 mugs


seconds3 = [212, 225, 228, 266, 273, 297, 363]
# Calculate the number of bins
num_bins = int((max(seconds3) - min(seconds3)) / bin_width) + 1
# Create a histogram with custom bins
plt.hist(seconds3, bins=num_bins, range=(min(seconds3), max(seconds3) + bin_width), color='red', edgecolor='black', alpha=0.5, label='10 mugs
# Customize the plot
plt.title('Seconds elapsed for 1 category, sorted by number of mugs');
plt.legend()
plt.xlabel('Seconds'); plt.ylabel('Frequency')

plt.show()# Show the plot
```



Plotting a histogram for the seconds elapsed for 1 design against the Exponential PDF

```
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats
from collections import Counter

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

# Your list of time values
seconds = [1, 1, 1, 1, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5

# Create a histogram of the data
bin_width = 5 # Define the bin width (5 seconds in this case)
num_bins = int((max(data) - min(data)) / bin_width) + 1
hist, bins = np.histogram(seconds, bins=num_bins, density = True)

# Find the bin width
bin_width = bins[1] - bins[0]

# Calculate the bin centers
bin_centers = (bins[:-1] + bins[1:]) / 2

# Plot the histogram
plt.bar(bin_centers, hist, width=bin_width, label="Histogram")

# Try fitting different distributions
distributions = [stats.norm, stats.expon, stats.gamma, stats.weibull_min]

best_fit = None
best_params = None
best_sse = float('inf')

for distribution in distributions:
    print("Testing the " + distribution.name + " distribution")
    # Fit the distribution to the data
    params = distribution.fit(seconds)

    # Generate the PDF (Probability Density Function) using the fitted parameters
    pdf = distribution.pdf(bin_centers, *params)

    # Calculate the Sum of Squared Errors (SSE) between the histogram and PDF
    sse = np.sum((hist - pdf)**2)

    # Check if this distribution has a lower SSE
    if sse < best_sse:
        best_fit = distribution
        best_params = params
        best_sse = sse

# Plot the best fit distribution
pdf = best_fit.pdf(bin_centers, *best_params)
plt.plot(bin_centers, pdf, label="Best Fit (" + best_fit.name + ")", color='red')

# Show the plot
plt.xlabel("Time")
plt.ylabel("Frequency")
plt.legend()
plt.show()

# Print the best fit distribution and its parameters
print("Best fit distribution:", best_fit.name)
print("Best fit parameters:", best_params)
```

```
Testing the norm distribution
Testing the expon distribution
Testing the gamma distribution
Testing the weibull_min distribution
```




```
Best fit distribution: expon
Best fit parameters: (1.0, 18.296195652173914)
```

Fitting the data with a Poisson Distribution

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit
from scipy.special import factorial
from scipy.stats import poisson

# get poisson deviated random numbers
data = [1, 1, 1, 1, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5

# the bins should be of integer width, because poisson is an integer distribution
bins = np.arange(190) - 0.5
entries, bin_edges, patches = plt.hist(data, bins=bins, density=True, label='Data')

# calculate bin centers
bin_centers = 0.5 * (bin_edges[1:] + bin_edges[:-1])


def fit_function(k, lamb):
    '''poisson function, parameter lamb is the fit parameter'''
    return poisson.pmf(k, lamb)


# fit with curve_fit
parameters, cov_matrix = curve_fit(fit_function, bin_centers, entries)

print("Parameter : ", parameters)

# plot poisson-deviation with fitted parameter
x_plot = np.arange(0, 190)

plt.plot(
    x_plot,
    fit_function(x_plot, *parameters),
    marker='o', linestyle='',
    label='Fit result',
)
plt.legend()
plt.show()
```

```
Parameter :  [6.85728612]
```

Plotting a histogram for the seconds spend in a session

```
import matplotlib.pyplot as plt
import numpy as np
plt.figure(figsize=(10,4))

seconds = [124, 1086, 15, 258, 117, 84, 37, 214, 15, 163, 20, 671, 1570, 439, 1031, 586, 69, 82, 11, 1269, 378, 2469, 240, 85, 660, 38, 1251,

bin_width = 60 # Define the bin width (60 seconds in this case)

print(len(seconds), "datapoints")
print("average of ", sum(seconds)/len(seconds))
# Calculate the number of bins
num_bins = int((max(seconds) - min(seconds)) / bin_width) + 1

# Create a histogram with custom bins
plt.hist(seconds, bins=num_bins, range=(min(seconds), max(seconds) + bin_width), color='blue', edgecolor='black')
# Calculate percentiles
median = np.percentile(seconds, 50)
percentile_90 = np.percentile(seconds, 90)
percentile_95 = np.percentile(seconds, 95)

# Add vertical markers for the 90th and 95th percentiles
plt.axvline(median, color='green', linestyle='dashed', linewidth=1, label='median')
plt.axvline(percentile_90, color='green', linestyle='dashed', linewidth=1, label='90th Percentile')
plt.axvline(percentile_95, color='red', linestyle='dashed', linewidth=1, label='95th Percentile')
# Add text labels for the percentile values
plt.text(median + 2, plt.ylim()[1] * 0.85, f'Median: {median:.2f}', color='blue', fontsize=10)
plt.text(percentile_90 + 2, plt.ylim()[1] * 0.75, f'90th Percentile: {percentile_90:.2f}', color='green', fontsize=10)
plt.text(percentile_95 + 2, plt.ylim()[1] * 0.65, f'95th Percentile: {percentile_95:.2f}', color='red', fontsize=10)

# Customize the plot
plt.title('Seconds elapsed for 1 design');
plt.xlabel('Seconds'); plt.ylabel('Frequency')

plt.show()# Show the plot
```

```
46 datapoints
average of  597.6739130434783
```