# Task 2: Fitting and Comparing Distributions

**COURSE NAME**

CSDS 413 Introduction to Data Analysis


**Authors:**

Wiam Skakri
Jacob Anderson


September 22, 2025

# Contents

## Context

In this task, you will explore how different types of real-world datasets may follow different distributions. You will need to develop a set of hypotheses and perform experiments to validate your own hypotheses.

# 1 Normal Distribution Dataset

## 1.1 Part A: Developing Hypotheses

Identify and collect a real-world dataset that you hypothesize follows a Normal distribution. Please be clear about the reasoning behind your hypothesis and be specific about the source of the dataset.

—

We have collected the Iris dataset for this example. Specifically, we will use the sepal widths of the Setosa species in the dataset. We hypothesize this data to follow a Normal distribution because the sepal widths of Setosas ought to be influenced by many underlying environmental factors. These factors could be anything from the composition of the soil, the level of shade, the composition of the surrounding ecosystem, the agricultural practices of surrounding human civilizations, etc. More generally, notions of performance in a homogenous population often approximate a Gaussian, as most observations will tend to distribute about the mean, symmetrically deviating from the general behavior on each side of the spectrum, becoming less frequent at extreme conditions. The Iris dataset was originally produced by Ronald Fisher in 1936 and was sourced from this page on Kaggle.

As our hypothesize relates to the Gaussian distribution PDF:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

The model defines a mean parameter $\mu$ that the distribution mass is centered around, reflecting the idea that the sepal widths are going to average to some value over the observations. The exponential term $e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$ defines the decay behavior for the Gaussian, dictated by the standard deviation parameter $\sigma$. The difference between the observation $x$ and $\mu$ is squared so that deviations from the mean decay exponentially and symmetrically about the center, which are then scaled relative to the variance $\sigma^2$, i.e. the general proclivity of a population to deviate from its mean.

In other words, if the distribution of mass for some population tends to deviate from the mean naturally (higher $\sigma^2$ ), then the decay for observations far from the mean is much slower. This captures the natural variance of the observations in our data, and the bell shape reflects the fact that many Setosas ought to have generally similar sepal widths, and extreme cases ought to be less and less common the more extreme the observations get.

### 1.2 Part B: Fitting Distributions

For this exercise, we will call each of the four different theoretical distributions (normal, uniform, power law, exponential) a "model". Fit the dataset (i.e., estimate the model parameters) against each model (not just the one you hypothesized) using maximum likelihood estimation (or using any technique you think is appropriate; make sure to comment on the validity of your approach). This should result in a total of **4 parameter sets**. Report the estimated parameters in the following tabular format:

| | | ***Model*** | | | |
|---|---|---|---|---|---|
| ***Dataset*** | ***# Observations*** | **Normal** | **Uniform** | **Power law** | **Exponential** |
| **Dataset 1** | $n_1$ | $\mu_1, \sigma_1$ | $a_1, b_1$ | $\alpha_1, x_{\min_1}$ | $\lambda_1$ |

Be sure to show the code you used to arrive at your final estimates clearly.

—

Below is the code that produced the parameter estimates and the tabulated estimates for this dataset (for the full tabulation described in the original assignment TeX, see Appendix):

```
def dists_fit(input_csv: str) -> tuple:
    """
    Fits the obs dataset to each model using MLE.

    :param input_csv: Path to input data to fit paramater(s) to
    :type input_csv: str
    """
    obs = pd.read_csv(input_csv).iloc[:, 0].to_numpy()

    mu = np.mean(obs)
    std = np.sqrt(np.sum((obs - mu) ** 2) / len(obs))

    a, b = obs.min(), obs.max()

    alpha = 1 + len(obs) / np.sum(np.log(obs / a))

    lamb = 1 / np.mean(obs)

    return (mu, std, a, b, alpha, lamb)
```

**Figure 1:** Parameter estimation function for the four models.

| | | ***Model*** | | | |
|---|---|---|---|---|---|
| ***Dataset*** | ***# Observations*** | **Normal** | **Uniform** | **Power law** | **Exponential** |
| **Iris Sepal Widths (Setosa)** | 50 | 3.4, 0.4 | 2.3, 4.4 | 3.6, 2.3 | 0.3 |

**Figure 2:** Parameter estimates for each model on Iris Sepal Widths (Setosa).

## 1.3 Part C: Comparing Real and Synthetic Data

For each fitted distribution (there will be 4 of them for this dataset, each corresponding to a different model), generate a synthetic sample of data points equal to the sample size of the real dataset using the respective model parameters you inferred from the real dataset.

Compare the real vs. synthetic data distributions using methods you think are the most appropriate, including visualizations. So, for this dataset, we compare the original dataset to four synthetic datasets, all with equal number of observations, but each synthetic dataset is generated using a different model.

For this dataset, identify the synthetic dataset (which corresponds to a model) that is most similar to the original data in terms of its distribution.

Now revisit your initial hypothesis. For this dataset: Did the dataset behave as expected, or was another model (assumed distribution) a better fit to the dataset? Reflect on why the observed results may differ from your expectations.

—

Below is the code that produced the synthetic datasets for each fitted distribution:

```
def dists_generate(input_csv: str, params: tuple, dist_type: str):
    """
    Generates N synthetic examples

    :param input_csv: Path to input dataset
    :type input_csv: str
    :param params: tuple of parameters returned by dists_fit (mu, std, a, b, alpha, lamb)
    :type params: tuple
    :param dist_type: name of the folder correponsding to a type of distribution
    :type dist_type: str
    """
    obs_df = pd.read_csv(input_csv)
    obs_name = obs_df.columns[0]
    obs = obs_df.iloc[:, 0].to_numpy()
    n = len(obs)

    mu, std, a, b, alpha, lamb = params

    gaussian_samples = int(np.random.normal(loc=mu, scale=std, size=n))
    uniform_samples = int(np.random.uniform(low=a, high=b, size=n))
    powerlaw_samples = int((a * (1 - np.random.uniform(0, 1, n)) ** (-1 / (alpha - 1))))
    exponential_samples = int(np.random.exponential(scale=1/lamb, size=n))

    gaussian_df = pd.DataFrame({f'{obs_name}_gaussian': gaussian_samples})
    uniform_df = pd.DataFrame({f'{obs_name}_uniform': uniform_samples})
    powerlaw_df = pd.DataFrame({f'{obs_name}_powerlaw': powerlaw_samples})
    exponential_df = pd.DataFrame({f'{obs_name}_exponential': exponential_samples})

    gaussian_df.to_csv(f'../datasets/{dist_type}/synth/{obs_name}_gaussian.csv', index=False)
    uniform_df.to_csv(f'../datasets/{dist_type}/synth/{obs_name}_uniform.csv', index=False)
    powerlaw_df.to_csv(f'../datasets/{dist_type}/synth/{obs_name}_powerlaw.csv', index=False)
    exponential_df.to_csv(f'../datasets/{dist_type}/synth/{obs_name}_exponential.csv', index=False)
```

**Figure 3:** Sampling function for fitted distributions.

We believe an appropriate test for these synthetic distributions would be the K-S statistic because we can equip ourselves with a notion of disagreement between the distributions in terms of their maximum difference between the cumulative probability structure of the actual dataset. The K-S test implemented in SciPy also provides the p-value for our comparisons, so before evening looking at the distributions we can gauge the disagreement between synthetic and real and the extent to which that disagreement is structurally significant or if it's just a product of random noise.

We can see in the table below that immediately the Power Law and Exponential models stand out tremendously with K-S statistics above and around 0.5, both of which are disagreements that are very structural in nature. On the other hand, the Uniform and Normal models show much less disagreement in their CDFs relative to the real sample, however the difference in p-value is very telling of which is actually representative of the data. The 0.04 p-value for the Uniform distribution tells us immediately that there is almost no chance, a 3.9% probability, that such as disagreement could come from two samples in the same distribution, whereas the p-value for the Normal suggests there you could see this K-S statistic with about a 96.7% chance if the two samples were pulled from the same distribution.

Without even observing the other two visualizations we prepared, we can make quite strong claims about which models do well to represent that Sepal Widths dataset and which do not, though it is always helpful to pair these statistics with a visualization that makes the comparison much more readily obvious. In Fig 5, we observe that the Power Law and Exponential models at least begin to converge, climbing a little bit of the probability mass of the real data, but the Uniform and Normal models clearly outmatch them with central tendencies much closer to the real sample.

| **Distribution** | **K-S Statistic** | **p-value** | **Significant** |
|---|---|---|---|
| Normal | 0.10 | 0.967 | No |
| Uniform | 0.28 | 0.039 | Yes |
| Power Law | 0.48 | $1.39 \times 10^{-5}$ | Yes |
| Exponential | 0.64 | $6.08 \times 10^{-10}$ | Yes |

**Figure 4:** Kolmogorov-Smirnov test results between Sepal Widths dataset and synthetic samples.

The histogram also makes it readily obvious the disparity between the Uniform and Normal. While the Uniform performs in a different class than the previous two models, it remains apparent from the mass distribution that the Uniform distribution is more right-skewed and not as centered about its mean as the real sample.

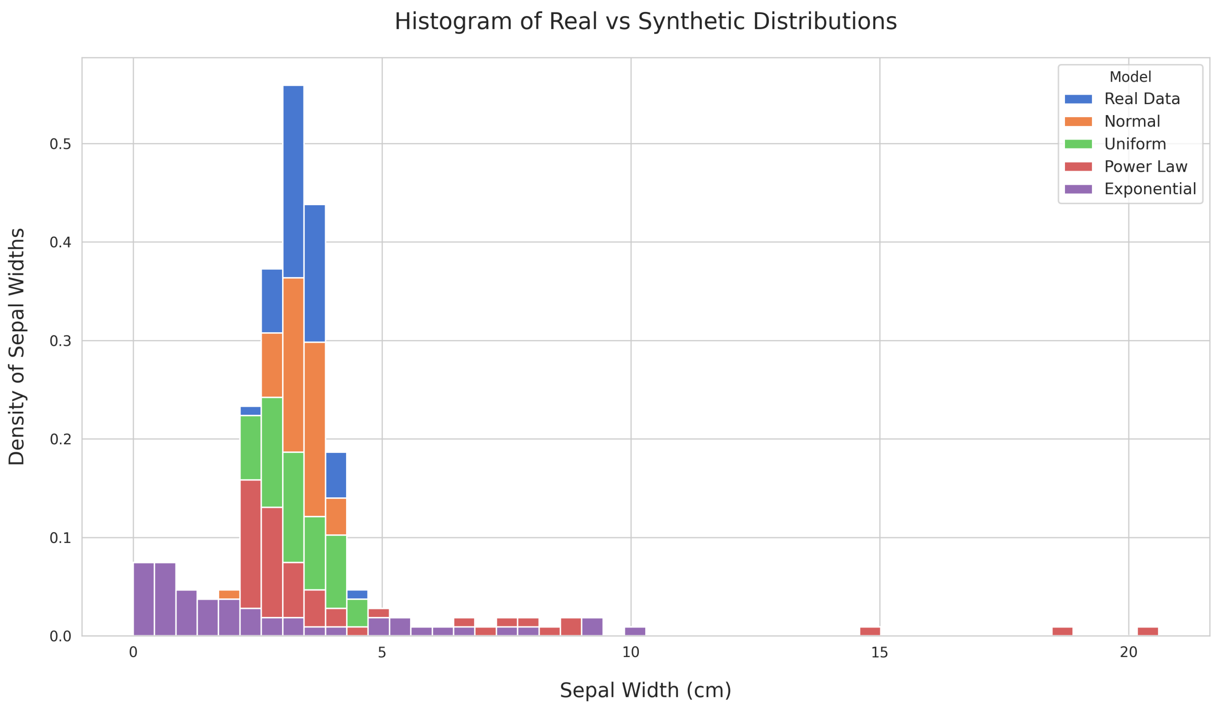


**Figure 5:** Histogram of Real vs Synthetic Distributions on Sepal Widths dataset.

To better confirm the comparison between the Uniform and Normal Distribution, we also curated a Q-QPlot of the data over 25 quantiles to gauge the agreement between the real and synthetic samples across the entire distribution. What we are left with is a general agreement of each model with the real sample's quantiles for average-case behavior, but the Uniform shows an obvious "snaking" pattern about the agreement line that worsens toward the extremes of the distributions.

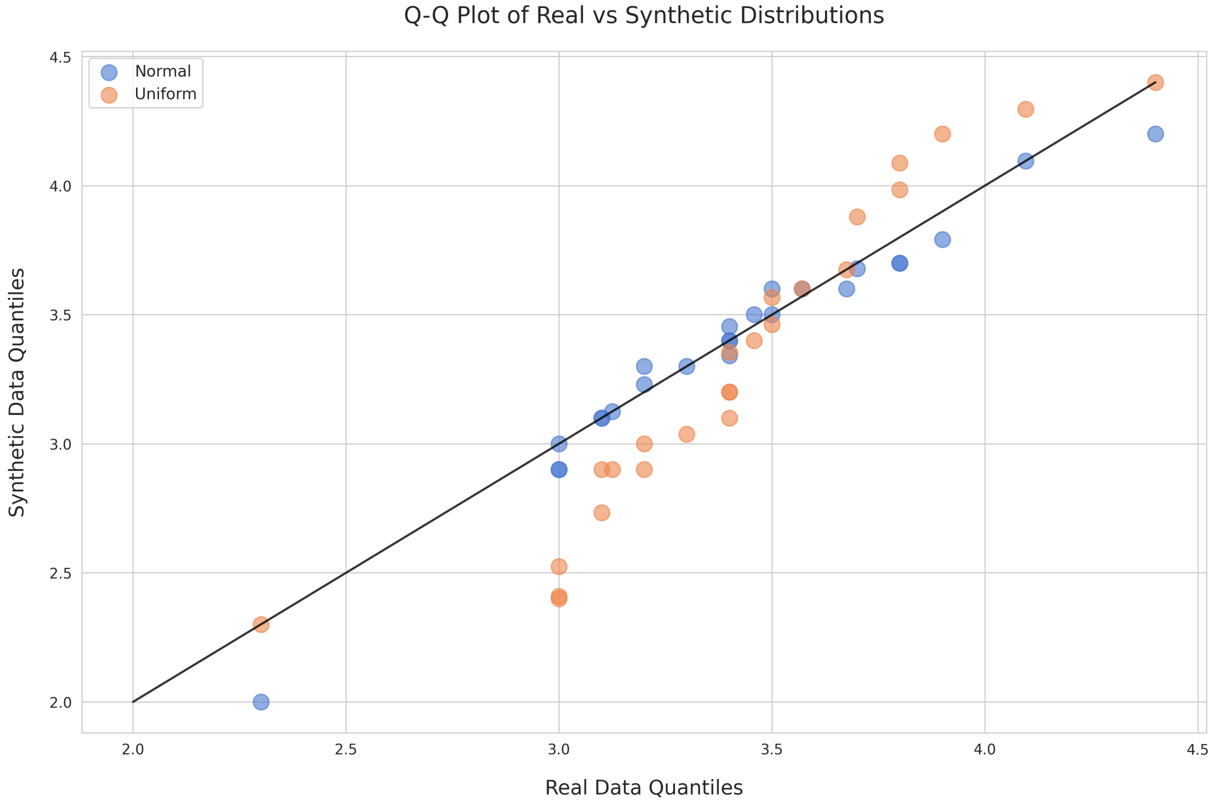


**Figure 6:** Q-Q Plot comparing Real vs Synthetic quantiles on Sepal Widths dataset.

Given these comparisons, we can make a strong claim that the sepal_widths_gaussian synthetic dataset is most similar to the real data in terms of its distribution. This aligns with our original claim that the Sepal Widths dataset contains notions of central tendency and a spread about its mean that decays toward extreme values.

# 2 Uniform Distribution Dataset

## 2.1 Part A: Developing Hypotheses

Identify and collect a real-world dataset that you hypothesize follows a Uniform distribution. Please be clear about the reasoning behind your hypothesis and be specific about the source of the dataset.

—

For this example, we sourced a dataset of 20-sided (d20) die rolls from Kaggle. The dataset contains the outcomes of 145 rolls of a single, unweighted d20. We hypothesize that this data follows a Uniform distribution because an unweighted die ought to have an equal probability of landing on any of its faces.

That would make repeated d20 rolls a clear example of a discrete uniform distribution, where all possible values in its range are equally massed. The probability mass function for a discrete uniform distribution from $a$ to $b$ (inclusive) is given by:

$$P(X = x) = \frac{1}{b - a + 1}, \quad \text{for } x \in \{a, a + 1, \ldots, b\}$$

where $a$ is 1 and $b$ is 20, so each face of the die ought to have a probability of around:

$$P(X = x) = \frac{1}{20} = 0.05$$

Because the d20 has a finite outcome space and possesses a notion of fairness, we expect the Uniform distribution to be the most appropriate model for representing this sample. It specifies exactly the parameters necessary to characterize this environment; nothing more imposed. The die doesn't not possess notions of general behavior or extreme values nor does it possess a unique spread of mass within different ranges of outcomes; it simply traverses the outcome space freely, bounded only by the values it can take on.

## 2.2 Part B: Fitting Distributions

For this exercise, we will call each of the four different theoretical distributions (normal, uniform, power law, exponential) a “model". Fit the dataset (i.e., estimate the model parameters) against each model (not just the one you hypothesized) using maximum likelihood estimation (or using any technique you think is appropriate; make sure to comment on the validity of your approach). This should result in a total of **4 parameter sets**. Report the estimated parameters in the following tabular format:

| | | ***Model*** | | | |
|---|---|---|---|---|---|
| ***Dataset*** | ***# Observations*** | **Normal** | **Uniform** | **Power law** | **Exponential** |
| **Dataset 2** | $n_2$ | $\mu_2, \sigma_2$ | $a_2, b_2$ | $\alpha_2, x_{\min_2}$ | $\lambda_2$ |

Be sure to show the code you used to arrive at your final estimates clearly.

—

Below are the tabulated parameter estimates for this dataset (for the full tabulation described in the original assignment TeX, see Appendix). The code in Fig. 1 was also used to arrive at our final parameter estimates for this dataset, here is the same implementation below for convenience:

```
def dists_fit(input_csv: str) -> tuple:
    """
    Fits the obs dataset to each model using MLE.

    :param input_csv: Path to input data to fit paramater(s) to
    :type input_csv: str
    """
    obs = pd.read_csv(input_csv).iloc[:, 0].to_numpy()

    mu = np.mean(obs)
    std = np.sqrt(np.sum((obs - mu) ** 2) / len(obs))

    a, b = obs.min(), obs.max()

    alpha = 1 + len(obs) / np.sum(np.log(obs / a))

    lamb = 1 / np.mean(obs)

    return (mu, std, a, b, alpha, lamb)
```

| | | ***Model*** | | | |
|---|---|---|---|---|---|
| ***Dataset*** | ***# Observations*** | **Normal** | **Uniform** | **Power law** | **Exponential** |
| **D20 Rolls** | 145 | 11.117, 6.055 | 1, 20 | 1.459, 1 | 0.090 |

**Figure 7:** Parameter estimates for each model on D20 Rolls dataset.

## 2.3 Part C: Comparing Real and Synthetic Data

For each fitted distribution (there will be 4 of them for this dataset, each corresponding to a different model), generate a synthetic sample of data points equal to the sample size of the real dataset using the respective model parameters you inferred from the real dataset.

Compare the real vs. synthetic data distributions using methods you think are the most appropriate, including visualizations. So, for this dataset, we compare the original dataset to four synthetic datasets, all with equal number of observations, but each synthetic dataset is generated using a different model.

For this dataset, identify the synthetic dataset (which corresponds to a model) that is most similar to the original data in terms of its distribution.

Now revisit your initial hypothesis. For this dataset: Did the dataset behave as expected, or was another model (assumed distribution) a better fit to the dataset? Reflect on why the observed results may differ from your expectations.

—–

To assess the synthetic data versus the real sample given our Uniform hypothesis, we need an understanding of how evenly distributed the probability mass is of the real sample and each of the model samples. That makes a histogram plot sound very appealing, however some of the other models, namely the exponential model and especially the power law model, fit toward very infrequent extreme values, which make it rather hard to visualize bins of die roll values in this way. Additionally, the Normal model possesses the unique quality of fitting toward a symmetric distribution, allowing for negative roles at the extremes of its distribution. That being said, we first take a look at their K-S statistics tabulated below:

| Distribution | K-S Statistic | p-value | Significant |
|---|---|---|---|
| Normal | 0.11 | 0.341 | No |
| Uniform | 0.12 | 0.273 | No |
| Power Law | 0.30 | 5.00 $\times 10^{-6}$ | Yes |
| Exponential | 0.21 | 0.004 | Yes |

**Figure 8:** Kolmogorov-Smirnov test results between D20 Die Rolls dataset and synthetic samples.

Interestingly enough, the K-S statistic accompanied with p-values shows that the Power Law model and the Exponential model are unsuitable for representing the real data die rolls, showing relatively higher disagreement that is almost certainly significant in nature, however the representative ability of the Normal and Uniform distribution appear quite similar from this test despite the fact that the Gaussian can and will sometimes sample values outside of the outcome space.

We now plot for our two most representative models the aforementioned histogram, without any worries of single extremes obfuscating the entire visual:

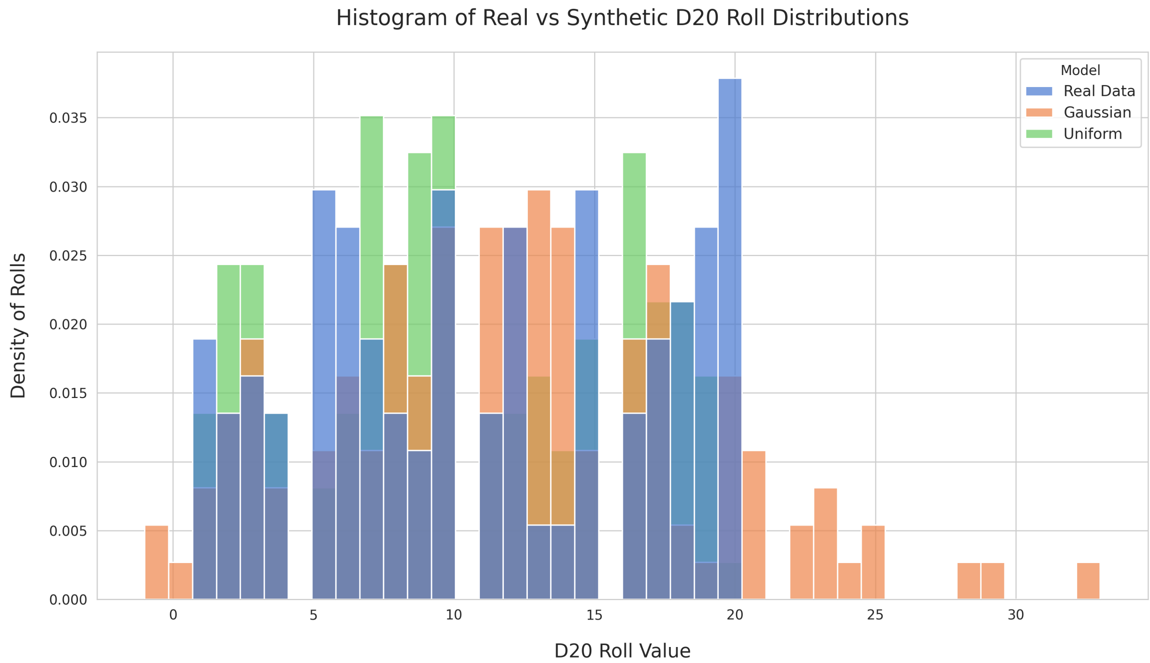

**Figure 9:** Histogram of Real vs Synthetic Distributions on D20 Die Rolls dataset.

It is immediately obvious from this plot that the Normal distribution is not as well-representative of the real sample toward extreme values, where the underlying assumptions of the uniform model regarding outcome space become relevant. The Uniform distribution mirrors the sporadic distribution of mass as the real sample set within the realms of possibilities for the die.

We can make a confident claim that the data sample produced by the Uniform model best represents the original data from its distribution. This confirms our hypothesis and the dataset behaved as we had expected; unweighted dice ought to be uniform in outcome.

# 3 Power Law Distribution Dataset

## 3.1 Part A: Developing Hypotheses

Identify and collect a real-world dataset that you hypothesize follows a Power Law distribution. Please be clear about the reasoning behind your hypothesis and be specific about the source of the dataset.
For the Power Law distribution, we decided to select a well-known example of power lar in the real-world: a dataset of US city populations, sourced from this Kaggle dataset. The Power Law distribution captures a very unique proclivity of certain phenomenon to have extreme values you can count on being present; a notion predictable frequency of extreme values in the distribution. It still decays and extreme values do become more and more scarce further from the mean, however the Power Law distribution maintains some amount of mass it relies on to capture a predicatable extreme behavior in the data. In this context, we hypothesize that the population of cities will follow this distribution because although it is true that very generally larger cities become less commonplace as that population grows, however there always will be a small yet reliable amount of cities that are extremely large in size.

This behavior of city population is reflected in the PDF for Power Law:

$$f(x) = \frac{\alpha - 1}{x_{\min}} \left( \frac{x}{x_{\min}} \right)^{-\alpha}$$

the chance of observing a city with some input size is proportional to $x^{-}a$, which dictates the natural inequality in the distribution between the average case behavior represented in the data and the extreme cases. This would allow the model to represent the proclivity of US cities to become mega-cities.

Regarding the data itself, it is a 2022 census of 19,268 US cities. For the purpose of fitting the Power Law model, it was necessary to remove 49 cities from the dataset that were made up of less than 10 people. This choice was made primarily because Power Law is designed to describe scaling behaviors about some minimum size, and in calculating logs of populations this small, we faced programmatic errors. Not to mention the census of cities this small could reflect some level of data collection error surely if the cities being characterized is the size of one large family. For this reason, and to preserve the underlying mechanism of the Power Law distribution, these "cities" were excluded.

### 3.2 Part B: Fitting Distributions

For this exercise, we will call each of the four different theoretical distributions (normal, uniform, power law, exponential) a “model". Fit the dataset (i.e., estimate the model parameters) against each model (not just the one you hypothesized) using maximum likelihood estimation (or using any technique you think is appropriate; make sure to comment on the validity of your approach). This should result in a total of **4 parameter sets**. Report the estimated parameters in the following tabular format:

| | | ***Model*** | | | |
|---|---|---|---|---|---|
| ***Dataset*** | ***# Observations*** | **Normal** | **Uniform** | **Power law** | **Exponential** |
| **Dataset 3** | $n_3$ | $\mu_3, \sigma_3$ | $a_3, b_3$ | $\alpha_3, x_{\min_3}$ | $\lambda_3$ |

Be sure to show the code you used to arrive at your final estimates clearly.

—

Below are the tabulated parameter estimates for this dataset (for the full tabulation described in the original assignment TeX, see Appendix). The code in Fig. 1 was also used to arrive at our final parameter estimates for this dataset, here is the same implementation below for convenience:

```
def dists_fit(input_csv: str) -> tuple:
    """
    Fits the obs dataset to each model using MLE.

    :param input_csv: Path to input data to fit paramater(s) to
    :type input_csv: str
    """
    obs = pd.read_csv(input_csv).iloc[:, 0].to_numpy()

    mu = np.mean(obs)
    std = np.sqrt(np.sum((obs - mu) ** 2) / len(obs))

    a, b = obs.min(), obs.max()

    alpha = 1 + len(obs) / np.sum(np.log(obs / a))

    lamb = 1 / np.mean(obs)

    return (mu, std, a, b, alpha, lamb)
```

| | | ***Model*** | | | |
|---|---|---|---|---|---|
| ***Dataset*** | ***# Observations*** | **Normal** | **Uniform** | **Power law** | **Exponential** |
| **US City Populations** | 19268 | 10800, 83590 | 11, 8335897 | 1.206, 11 | 0.0000926 |

**Figure 10:** Parameter estimates for each model on US City Populations dataset.

### 3.3 Part C: Comparing Real and Synthetic Data

For each fitted distribution (there will be 4 of them for this dataset, each corresponding to a different model), generate a synthetic sample of data points equal to the sample size of the real dataset using the respective model parameters you inferred from the real dataset.

Compare the real vs. synthetic data distributions using methods you think are the most appropriate, including visualizations. So, for this dataset, we compare the original dataset to four synthetic datasets, all with equal number of observations, but each synthetic dataset is generated using a different model.

For this dataset, identify the synthetic dataset (which corresponds to a model) that is most similar to the original data in terms of its distribution.

Now revisit your initial hypothesis. For this dataset: Did the dataset behave as expected, or was another model (assumed distribution) a better fit to the dataset? Reflect on why the observed results may differ from your expectations.

---

We plotted a log-log Q-Q plot for the sample because of the magnitude of the sample ranges, which should in turn tell us which samples better resemble a Power Law distribution based on which are best aligned with the diagonal:

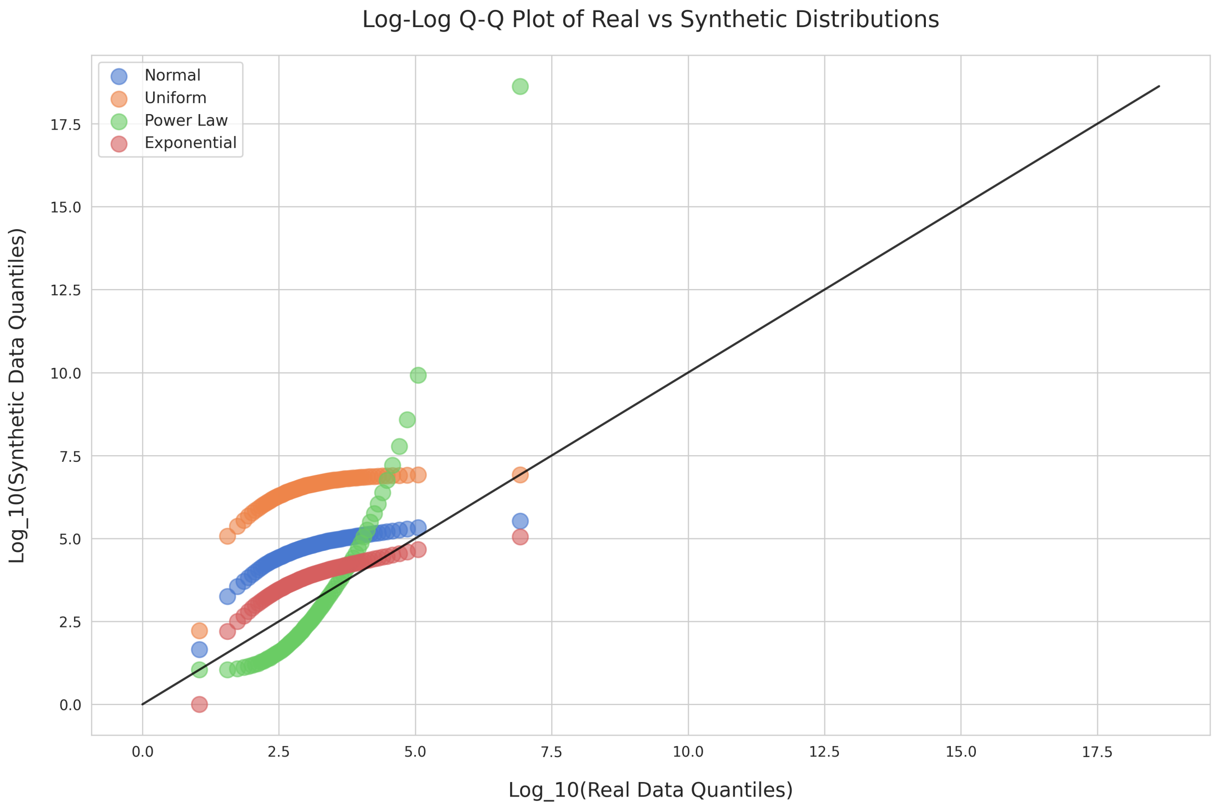


**Figure 11:** Q-Q Log-Log Plot comparing Real vs Synthetic quantiles on US City Populations dataset.

This visualization shows the models all struggled to produce a data sample resembling a Power Law distribution. We were surprised by this result until we doubled back and saw that the estimates of the Power Law parameters produce by the MLE step were quite poor, with an $\alpha$ of 1.206. Our data sample for our most promising distribution does not have a definable mean nor variance, and data values so extreme that plotting on visuals like a histogram would be infeasible for the entire range.

| Distribution | K-S Statistic | p-value | Significant |
|---|---|---|---|
| Normal | 0.46 | 0.000 | Yes |
| Uniform | 0.97 | 0.000 | Yes |
| Power Law | 0.31 | 0.000 | Yes |
| Exponential | 0.45 | 0.000 | Yes |

**Figure 12:** Kolmogorov-Smirnov test results between the US City Populations dataset and synthetic samples.

To corroborate this result for the data, we produce a K-S statistics table, showing incredibly low levels of uncertainty in the structural disagreements between the real data and each of these models. We can at least say though that the agreement of the Power Law model is still the best, but is a far cry from anything that we can speak definitely on.

That being said, we can only say with moderate levels of uncertainty that the Power Law model did perform best to mirror the real data from is distribution. The dataset certainly did not behave as expected, and perhaps we would benefit from pursuing other techniques with which to fit the data. Potential reasons why the reality was so different from our expectations could be attributed to the fact that there were many data points that perhaps were not filtered out that were too small to be reasonably subjected to Power Law scaling, making the challenge of estimation much more difficult. Not to mention the exponential distribution, the only distribution somewhat resembling the shape of the Power Law distribution struggled greatly as well, so there is an argument to be made that perhaps that the data may need to be dissected and cleaned much more heavily before attempting to fit a model to it using MLE.

# 4 Exponential Distribution Dataset

## 4.1 Part A: Developing Hypotheses

Identify and collect a real-world dataset that you hypothesize follows an Exponential distribution. Please be clear about the reasoning behind your hypothesis and be specific about the source of the dataset.
We sourced a dataset from Allen Downey, a principal data scientist at PyMC and a former professor at Olin college, from his Data Exploration repository on GitHub. This dataset possesses maternity hospital data, characterizing the intervals of births in minutes. It is a small set of timestamps through a single day, which we transformed into uni-variate observations of the inter-arrival times of consecutive births.

We hypothesize that birth intervals follow an Exponential distribution as this measure possess a notion of a central tendency, an average interval between births, and can at times extends to extreme values for a myriad of underlying reasons. What we see are a bunch of independent events naturally occurring a some rate throughout this day, and what is important to understand about these observations is that despite a central tendency, they are completely unrelated to the intervals between previous consecutive births. For this reason, our density function for the Exponential distribution:

$$f(x) = \lambda e^{-\lambda x}$$

makes a natural representation of independent events spaced out somewhat evenly. Additionally, this distribution has a decay property at further deviates from its mean in the positive direction, meaning that it can readily characterize perhaps the increasing rarity or longer intervals between births occurring, and it turn characterizes the increasing proclivity of births to occur at intervals surrounding the mean.

## 4.2 Part B: Fitting Distributions

For this exercise, we will call each of the four different theoretical distributions (normal, uniform, power law, exponential) a “model". Fit the dataset (i.e., estimate the model parameters) against each model (not just the one you hypothesized) using maximum likelihood estimation (or using any technique you think is appropriate; make sure to comment on the validity of your approach). This should result in a total of **4 parameter sets**. Report the estimated parameters in the following tabular format:

| | | ***Model*** | | | |
|---|---|---|---|---|---|
| ***Dataset*** | ***# Observations*** | **Normal** | **Uniform** | **Power law** | **Exponential** |
| **Dataset 4** | $n_4$ | $\mu_4, \sigma_4$ | $a_4, b_4$ | $\alpha_4, x_{\min_4}$ | $\lambda_4$ |

Be sure to show the code you used to arrive at your final estimates clearly.

—

Below are the tabulated parameter estimates for this dataset (for the full tabulation described in the original assignment TeX, see Appendix). The code in Fig. 1 was also used to arrive at our final parameter estimates for this dataset, here is the same implementation below for convenience:

```
def dists_fit(input_csv: str) -> tuple:
    """
    Fits the obs dataset to each model using MLE.

    :param input_csv: Path to input data to fit paramater(s) to
    :type input_csv: str
    """
    obs = pd.read_csv(input_csv).iloc[:, 0].to_numpy()

    mu = np.mean(obs)
    std = np.sqrt(np.sum((obs - mu) ** 2) / len(obs))

    a, b = obs.min(), obs.max()

    alpha = 1 + len(obs) / np.sum(np.log(obs / a))

    lamb = 1 / np.mean(obs)

    return (mu, std, a, b, alpha, lamb)
```

| | | ***Model*** | | | |
|---|---|---|---|---|---|
| ***Dataset*** | ***# Observations*** | **Normal** | **Uniform** | **Power law** | **Exponential** |
| **Brisbane Birth Intervals** | 43 | 33.3, 29.2 | 1, 157 | 1.32, 1 | 0.0301 |

**Figure 13:** Parameter estimates for each model on Brisbane Birth Intervals dataset.

### 4.3 Part C: Comparing Real and Synthetic Data

For each fitted distribution (there will be 4 of them for this dataset, each corresponding to a different model), generate a synthetic sample of data points equal to the sample size of the real dataset using the respective model parameters you inferred from the real dataset.

Compare the real vs. synthetic data distributions using methods you think are the most appropriate, including visualizations. So, for this dataset, we compare the original dataset to four synthetic datasets, all with equal number of observations, but each synthetic dataset is generated using a different model.

For this dataset, identify the synthetic dataset (which corresponds to a model) that is most similar to the original data in terms of its distribution.

Now revisit your initial hypothesis. For this dataset: Did the dataset behave as expected, or was another model (assumed distribution) a better fit to the dataset? Reflect on why the observed results may differ from your expectations.

We first ought to compare the samples by how well-they reflect the decaying nature of an Exponential at greater interval ranges, for which we can use a histogram:

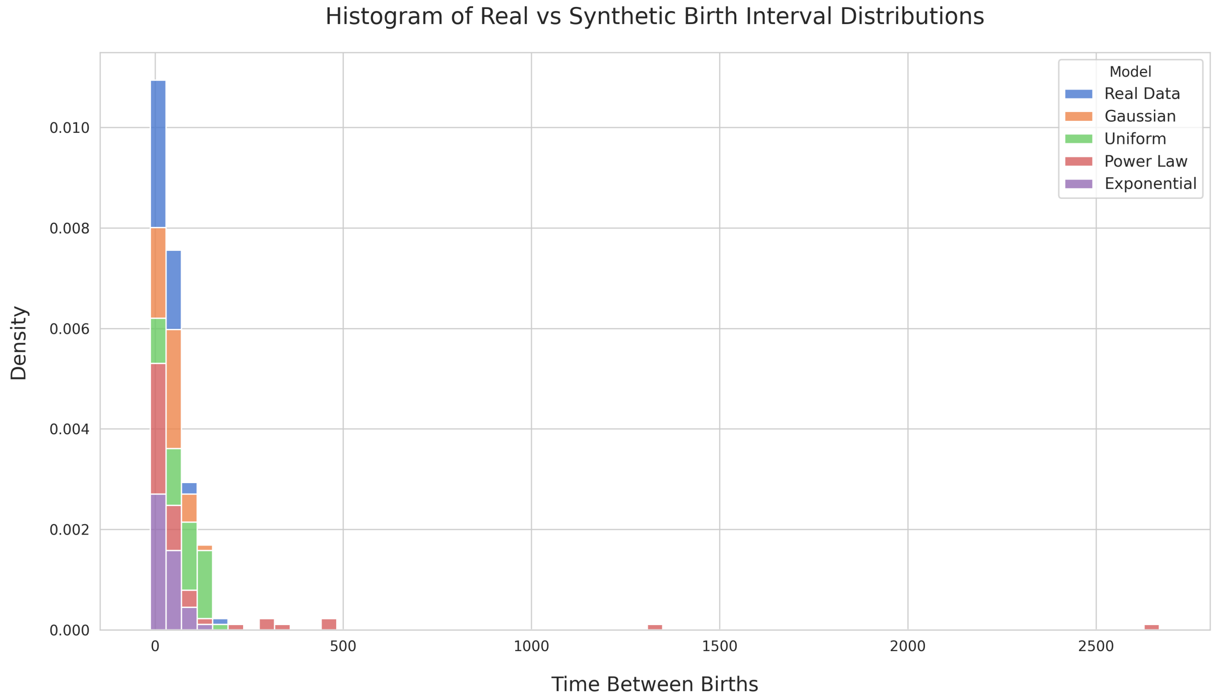


**Figure 14:** Histogram of Real vs Synthetic Distributions on Birth Intervals dataset.

Viewing how the mass is distributed was challenging for overlapping distributions, so we chose to stack them, and it is clear visually that for where the mass is concentrated in the real dataset, we see more mass distributed in the same spot for the Power Law Distribution and the Exponential distribution. It is also apparent that the Power Law distribution is still trying to fit to extreme values in the data, suggesting that while a strong choice, it may not be the best option for representing a decay pattern at extremes. The other two datasets are quite unsuitable in this case and are showcase their own idiosyncrasies while somewhat adhere to the structure of the Birth Intervals dataset.

We produced a K-S test below to gain a more concrete understanding of how the Power Law distribution and the Exponential distribution compare in this context:

| Distribution | K-S Statistic | p-value | Significant |
|---|---|---|---|
| Gaussian | 0.28 | 0.070 | No |
| Uniform | 0.51 | $1.73 \times 10^{-5}$ | Yes |
| Power Law | 0.30 | 0.039 | Yes |
| Exponential | 0.12 | 0.938 | No |

**Figure 15:** Kolmogorov-Smirnov test results between the Birth Intervals dataset and synthetic samples.

Surprisingly, the K-S test told us that there is actually much higher disagreement than we could discern from the histogram alone between the real data and the Power Law model with a significant p-value of 0.039, and even more surprisingly, the Gaussian, while not as in agreement with the data as the Exponential, agrees more with the real data than the Power Law distribution, which perhaps makes sense when you reduce them both to distributions that prioritize decay away from the center.

From these visualizations, we can make a confident claim that the dataset produce by the Exponential distribution is most similar to the real dataset. This was the expected behavior for this data, and in fact the fit and visualization showed with much greater clarity how important the general notions of a distribution are to how it is characterized, and specifically how it is differentiated from others. Two distributions like Power Law and Exponential can look very similar, but there is nuance that distances them greater than may be readily apparent.