COMPSCI 590N
Lecture 9: Sparse Matrices and Probability 1

Roy J. Adams

College of Information and Computer Sciences
University of Massachusetts Amherst

Outline

Sparsity

- A **sparse matrix** is an matrix in which most of the entries are zero.

Sparsity

- A **sparse matrix** is an matrix in which most of the entries are zero.
- This is in contrast to a **dense matrix**.

Sparsity

- A **sparse matrix** is an matrix in which most of the entries are zero.
- This is in contrast to a **dense matrix**.
- Sparse matrices naturally appear in many applications:

Sparsity

- A **sparse matrix** is an matrix in which most of the entries are zero.
- This is in contrast to a **dense matrix**.
- Sparse matrices naturally appear in many applications:
  - Network adjacency matrices are typically very sparse. For example, you are only Facebook friends with a small percentage of the total Facebook population.

Sparsity

- A **sparse matrix** is an matrix in which most of the entries are zero.
- This is in contrast to a **dense matrix**.
- Sparse matrices naturally appear in many applications:
  - Network adjacency matrices are typically very sparse. For example, you are only Facebook friends with a small percentage of the total Facebook population.
  - In recommender systems, ratings are often arranged as a sparse matrix.

Sparsity

- A **sparse matrix** is an matrix in which most of the entries are zero.
- This is in contrast to a **dense matrix**.
- Sparse matrices naturally appear in many applications:
  - Network adjacency matrices are typically very sparse. For example, you are only Facebook friends with a small percentage of the total Facebook population.
  - In recommender systems, ratings are often arranged as a sparse matrix.
  - In NLP, the matrix of word counts in a set of documents is typically sparse.

Sparsity

- A **sparse matrix** is an matrix in which most of the entries are zero.
- This is in contrast to a **dense matrix**.
- Sparse matrices naturally appear in many applications:
  - Network adjacency matrices are typically very sparse. For example, you are only Facebook friends with a small percentage of the total Facebook population.
  - In recommender systems, ratings are often arranged as a sparse matrix.
  - In NLP, the matrix of word counts in a set of documents is typically sparse.
  - Multiple parallel event sequences can be arranged as a matrix. If the events are uncommon, then the matrix is sparse.

Sparsity

- A **sparse matrix** is an matrix in which most of the entries are zero.
- This is in contrast to a **dense matrix**.
- Sparse matrices naturally appear in many applications:
  - Network adjacency matrices are typically very sparse. For example, you are only Facebook friends with a small percentage of the total Facebook population.
  - In recommender systems, ratings are often arranged as a sparse matrix.
  - In NLP, the matrix of word counts in a set of documents is typically sparse.
  - Multiple parallel event sequences can be arranged as a matrix. If the events are uncommon, then the matrix is sparse.
- If we know an matrix is sparse, we can take advantage of this structure to speed up computations on the matrix.

Sparsity

- Consider taking the inner product of two length n vectors, x and y.

Sparsity

- Consider taking the inner product of two length n vectors, x and y.
- In general, how many multiplications must we perform?

Sparsity

- Consider taking the inner product of two length n vectors, x and y.
- In general, how many multiplications must we perform?
- What if we know that only 10% of the entries in x are non-zero and we know where the are, how many multiplications do we need to perform?

Sparse Representations

There are two main strategies for storing sparse matrices:

1. Formats that support **efficient modifications** include Dictionary of Keys (DOK), List of Lists (LIL), and Coordinate list (COO) formats.

Sparse Representations

There are two main strategies for storing sparse matrices:

1. Formats that support **efficient modifications** include Dictionary of Keys (DOK), List of Lists (LIL), and Coordinate list (COO) formats.
2. Formats that support **efficient access and operations** include Compressed Sparse Row (CSR) and Compressed Sparse Column (CSC) formats.

Dictionary of Keys Format

- The DOK format is perhaps the simplest of the formats for efficient modification.

Dictionary of Keys Format

- The DOK format is perhaps the simplest of the formats for efficient modification.
- The DOK format stores the matrix as a dictionary with row/column tuples as keys and one key/value pair per non-zero entry.

Dictionary of Keys Format

- The DOK format is perhaps the simplest of the formats for efficient modification.
- The DOK format stores the matrix as a dictionary with row/column tuples as keys and one key/value pair per non-zero entry.
- Adding or removing an entry can be done in constant time.

Dictionary of Keys Format

- The DOK format is perhaps the simplest of the formats for efficient modification.
- The DOK format stores the matrix as a dictionary with row/column tuples as keys and one key/value pair per non-zero entry.
- Adding or removing an entry can be done in constant time.
- What is the complexity of row or column slicing?

Compressed Sparse Row Format

- The CSR format stores an $m \times n$ matrix as three one dimensional arrays `indices`, `indptr`, and `data`.

Compressed Sparse Row Format

- The CSR format stores an $m \times n$ matrix as three one dimensional arrays `indices`, `indptr`, and `data`.
  1. The `data` array stores the non-zero entries of the matrix in a left-to-right top-to-bottom order (row major order).

Compressed Sparse Row Format

- The CSR format stores an $m \times n$ matrix as three one dimensional arrays `indices`, `indptr`, and `data`.
  1. The `data` array stores the non-zero entries of the matrix in a left-to-right top-to-bottom order (row major order).
  2. The `indices` array has the same length as the `data` array and stores the column of each entry.

Compressed Sparse Row Format

- The CSR format stores an $m \times n$ matrix as three one dimensional arrays `indices`, `indptr`, and `data`.
  1. The `data` array stores the non-zero entries of the matrix in a left-to-right top-to-bottom order (row major order).
  2. The `indices` array has the same length as the `data` array and stores the column of each entry.
  3. The `indptr` array is a length m array. `indptr[i]` stores the index in `data` and `indices` of the first non-zero entry in the ith row.

Compressed Sparse Row Format

- The CSR format stores an $m \times n$ matrix as three one dimensional arrays `indices`, `indptr`, and `data`.
  1. The `data` array stores the non-zero entries of the matrix in a left-to-right top-to-bottom order (row major order).
  2. The `indices` array has the same length as the `data` array and stores the column of each entry.
  3. The `indptr` array is a length m array. `indptr[i]` stores the index in `data` and `indices` of the first non-zero entry in the ith row.
- The CSC format is the same as CSR, but `data` is stored in column major order and `indices` stores the row of each entry.

Compressed Sparse Row Format: Example

$$\begin{bmatrix} 1 & 0 & 2 \\ 0 & 0 & 3 \\ 4 & 5 & 0 \end{bmatrix}$$

```
indptr:
  [0,2,3,5]
indices:
  [0,2,2,0,1]
data:
  [1,2,3,4,5]
```

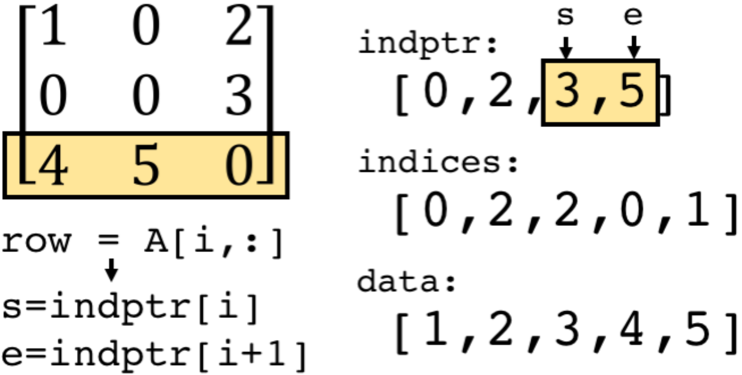
Compressed Sparse Row Format: Example

$$\begin{bmatrix} 1 & 0 & 2 \\ 0 & 0 & 3 \\ 4 & 5 & 0 \end{bmatrix}$$

```
row = A[i,:]
```

```
indptr:
  [0,2,3,5]
indices:
  [0,2,2,0,1]
data:
  [1,2,3,4,5]
```

Compressed Sparse Row Format: Example



Compressed Sparse Row Format: Example

$$\begin{bmatrix}1 & 0 & 2\\ 0 & 0 & 3\\ 4 & 5 & 0\end{bmatrix}$$

```
row = A[i,:]
s=indptr[i]
e=indptr[i+1]
row=data[s:e]
```

```
indptr:    s   e
  [0,2,3,5]
indices:
  [0,2,2,0,1]
data:
  [1,2,3,4,5]
```

Compressed Sparse Row Format: Example

$$\begin{bmatrix} 1 & 0 & 2 \\ 0 & 0 & 3 \\ \boxed{4 \quad 5 \quad 0} \end{bmatrix}$$

```
row = A[i,:]
s=indptr[i]
e=indptr[i+1]
row=data[s:e]
```

indptr:

$[0,2,\boxed{\overset{s}{3},\overset{e}{5}}]$

indices:

$[0,2,2,0,1]$

data:

$[1,2,3,\boxed{4,5}]$

Scipy Sparse Matrices

The module `scipy.sparse` implements each of these sparse formats.

```
>>> import scipy.sparse as sps
>>> import numpy as np

>>> A = np.eye(5)
>>> identity = np.eye(5)
>>> sparse_identity = sps.csr_matrix(identity)

>>> sparse_identity.indptr
array([0, 1, 2, 3, 4, 5], dtype=int32)
>>> sparse_identity.indices
array([0, 1, 2, 3, 4], dtype=int32)
>>> sparse_identity.data
array([ 1.,  1.,  1.,  1.,  1.])
```

Interactive Demo

- What linear algebra operations are most sped up by using sparse matrices?

Outline

Random Variables

- Probability and statistics play a central role in data analysis, modeling, and numerical algorithms.

Random Variables

- Probability and statistics play a central role in data analysis, modeling, and numerical algorithms.
- But first, a quick review.

Random Variables

- Probability and statistics play a central role in data analysis, modeling, and numerical algorithms.
- But first, a quick review.

Random Variables

- Probability and statistics play a central role in data analysis, modeling, and numerical algorithms.
- But first, a quick review.

Random Variables

A random variable, X, is a quantity that can take any value from a set of possible values, Ω, according to a set of probabilities.

Random Variables

For example: Imagine we are flipping a coin.

Random Variables

For example: Imagine we are flipping a coin.

- What is the random variable?

Random Variables

For example: Imagine we are flipping a coin.

- What is the random variable?
  - The random variable is the outcome of the coin flip.

Random Variables

For example: Imagine we are flipping a coin.

- What is the random variable?
    - The random variable is the outcome of the coin flip.
- What is the set of possible outcomes?

Random Variables

For example: Imagine we are flipping a coin.

- What is the random variable?
  - The random variable is the outcome of the coin flip.
- What is the set of possible outcomes?
  - $\Omega = \{H, T\}$

Discrete Random Variables

- A discrete random variable may take its value from a finite or countably infinite set.

Discrete Random Variables

- A discrete random variable may take its value from a finite or countably infinite set.
- Examples include:

Discrete Random Variables

- A discrete random variable may take its value from a finite or countably infinite set.
- Examples include:
    - The outcome of a coin flip can take one of two possible values.

Discrete Random Variables

- A discrete random variable may take its value from a finite or countably infinite set.
- Examples include:
  - The outcome of a coin flip can take one of two possible values.
  - A randomly dealt five card poker hand can take one of $\approx$2.6 possible million values.

Discrete Random Variables

- A discrete random variable may take its value from a finite or countably infinite set.
- Examples include:
  - The outcome of a coin flip can take one of two possible values.
  - A randomly dealt five card poker hand can take one of $\approx$2.6 possible million values.
  - The number of people who log in to Netflix between 1pm and 2pm can be any non-negative integer.

Discrete Random Variables

- A discrete random variable may take its value from a finite or countably infinite set.
- Examples include:
  - The outcome of a coin flip can take one of two possible values.
  - A randomly dealt five card poker hand can take one of $\approx$2.6 possible million values.
  - The number of people who log in to Netflix between 1pm and 2pm can be any non-negative integer.
  - The end of season goal differential for a soccer team can be any integer (positive or negative).

Probability Mass Functions

The probability of each possible outcome is defined by a Probability Mass Function.

Probability Mass Function

For a discrete random variable X with support Ω, a Probability Mass Function (PMF) $P : \Omega \to [0, 1]$ maps possible outcomes to probabilities. A PMF must satisfy two conditions:

1. Probability of any single outcome must be between zero and one (i.e. $P(x) \in [0, 1]$ for all $x \in \Omega$).

Probability Mass Functions

The probability of each possible outcome is defined by a Probability Mass Function.

Probability Mass Function

For a discrete random variable X with support Ω, a Probability Mass Function (PMF) $P : \Omega \to [0, 1]$ maps possible outcomes to probabilities. A PMF must satisfy two conditions:

1. Probability of any single outcome must be between zero and one (i.e. $P(x) \in [0, 1]$ for all $x \in \Omega$).
2. The probabilities of all possible outcomes must sum to one (i.e. $\sum_x P(x) = 1$).

Probability Mass Functions: Examples

Let X represent the outcome of a six sided dice roll.

- The set of possible outcomes is $\Omega = \{1, 2, 3, 4, 5, 6\}$.

Probability Mass Functions: Examples

Let X represent the outcome of a six sided dice roll.

- The set of possible outcomes is $\Omega = \{1, 2, 3, 4, 5, 6\}$.
- Assuming the dice is fair, then the PMF may look like:

Probability Mass Functions: Examples

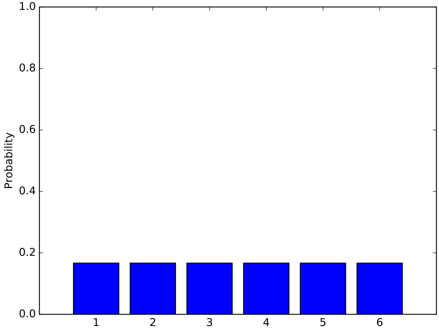
Let X represent the outcome of a six sided dice roll.

- The set of possible outcomes is $\Omega = \{1, 2, 3, 4, 5, 6\}$.
- Assuming the dice is fair, then the PMF may look like:

Probability Mass Functions: Examples

Let X represent the outcome of a six sided dice roll.

- The set of possible outcomes is $\Omega = \{1, 2, 3, 4, 5, 6\}$.
- Assuming the dice is fair, then the PMF may look like:



Probability Mass Functions: Examples

Let X represent the the number of people entering a certain bank between 1pm and 2pm.

- The set of possible outcomes is the set of all non-negative integers $\Omega = \mathbb{Z}_{\geq 0}$.

Probability Mass Functions: Examples

Let X represent the the number of people entering a certain bank between 1pm and 2pm.

- The set of possible outcomes is the set of all non-negative integers $\Omega = \mathbb{Z}_{\geq 0}$.
- This random variable is a canonical example of a Poisson distributed random variable which has the following PMF:

Probability Mass Functions: Examples

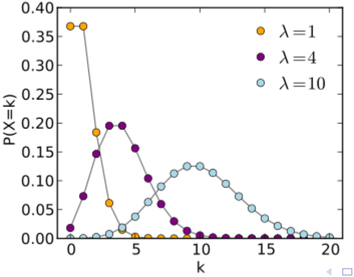
Let X represent the the number of people entering a certain bank between 1pm and 2pm.

- The set of possible outcomes is the set of all non-negative integers $\Omega = \mathbb{Z}_{\geq 0}$.
- This random variable is a canonical example of a Poisson distributed random variable which has the following PMF:

Probability Mass Functions: Examples

Let X represent the the number of people entering a certain bank between 1pm and 2pm.

- The set of possible outcomes is the set of all non-negative integers $\Omega = \mathbb{Z}_{\geq 0}$.
- This random variable is a canonical example of a Poisson distributed random variable which has the following PMF:



Parametric Distributions

The Poisson distribution is an example of a **parametric distribution**, that is, it requires a set of parameter values to fully specify the distribution.

Parametric Distributions

The Poisson distribution is an example of a **parametric distribution**, that is, it requires a set of parameter values to fully specify the distribution.

$$f(X = x; \lambda) = \frac{\lambda^x e^{-\lambda}}{x!}, \lambda > 0$$

Parametric Distributions

The Poisson distribution is an example of a **parametric distribution**, that is, it requires a set of parameter values to fully specify the distribution.

$$f(X = x; \lambda) = \frac{\lambda^x e^{-\lambda}}{x!}, \lambda > 0$$

- In this case the, the distribution has a single parameter λ that must be a positive real number.

Parametric Distributions

The Poisson distribution is an example of a **parametric distribution**, that is, it requires a set of parameter values to fully specify the distribution.

$$f(X = x; \lambda) = \frac{\lambda^x e^{-\lambda}}{x!}, \lambda > 0$$

- In this case the, the distribution has a single parameter λ that must be a positive real number.
- Much of statistics is concerned with inferring these parameters from data.

Continuous Random Variables

- A continuous random variable takes its value from an uncountably infinite set such as the set of real numbers.

Continuous Random Variables

- A continuous random variable takes its value from an uncountably infinite set such as the set of real numbers.
- Examples include:

Continuous Random Variables

- A continuous random variable takes its value from an uncountably infinite set such as the set of real numbers.
- Examples include:
  - The height of a randomly selected person.

Continuous Random Variables

- A continuous random variable takes its value from an uncountably infinite set such as the set of real numbers.
- Examples include:
  - The height of a randomly selected person.
  - Income of a randomly selected household.

Continuous Random Variables

- A continuous random variable takes its value from an uncountably infinite set such as the set of real numbers.
- Examples include:
  - The height of a randomly selected person.
  - Income of a randomly selected household.
  - The amount of time between hard drive failures in a server.

Probability Density Functions

The distribution over possible values of a continuous random variable is given by a **probability density function**.

Probability Density Function

For a continuous random variable X, the probability density function (PDF) $P(x)$ describes the relative likelihood of a continuous random variable taking a given value. A PDF must satisfy the following two conditions:

1. All values must be non-negative. That is, $P(x) \geq 0$ for all $x \in \Omega$.

Probability Density Functions

The distribution over possible values of a continuous random variable is given by a **probability density function**.

Probability Density Function

For a continuous random variable X, the probability density function (PDF) $P(x)$ describes the relative likelihood of a continuous random variable taking a given value. A PDF must satisfy the following two conditions:

1. All values must be non-negative. That is, $P(x) \geq 0$ for all $x \in \Omega$.
2. The area under the PDF must equal one. That is, $\int_x P(x)dx = 1$.

Probability Density Functions

The distribution over possible values of a continuous random variable is given by a **probability density function**.

Probability Density Function

For a continuous random variable X, the probability density function (PDF) $P(x)$ describes the relative likelihood of a continuous random variable taking a given value. A PDF must satisfy the following two conditions:

1. All values must be non-negative. That is, $P(x) \geq 0$ for all $x \in \Omega$.
2. The area under the PDF must equal one. That is, $\int_x P(x)dx = 1$.

Probability Density Functions

The distribution over possible values of a continuous random variable is given by a **probability density function**.

Probability Density Function

For a continuous random variable X, the probability density function (PDF) $P(x)$ describes the relative likelihood of a continuous random variable taking a given value. A PDF must satisfy the following two conditions:

1. All values must be non-negative. That is, $P(x) \geq 0$ for all $x \in \Omega$.
2. The area under the PDF must equal one. That is, $\int_x P(x)dx = 1$.

The probability of X falling between a and b is given by the integral:

$$P(a < x < b) = \int_x p(x)dx$$

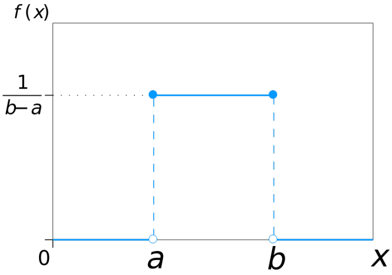
Probability Density Functions: Examples

- Given some range $[a, b]$, the **uniform distribution** places equal likelihood on all values in the range.

Probability Density Functions: Examples

- Given some range $[a, b]$, the **uniform distribution** places equal likelihood on all values in the range.

$$p(x; a, b) = \frac{1}{b - a}$$



Probability Density Functions: Examples

- Perhaps the most common distribution in statistics is the **Normal** distribution.

Probability Density Functions: Examples

- Perhaps the most common distribution in statistics is the **Normal** distribution.
- The Normal distribution takes two parameters: a mean parameter μ and a variance parameter σ^2.

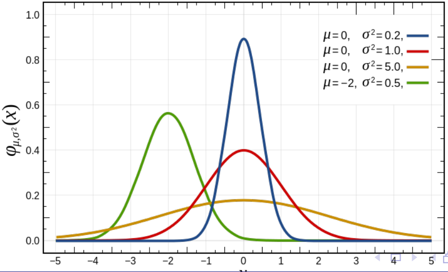
Probability Density Functions: Examples

- Perhaps the most common distribution in statistics is the **Normal** distribution.
- The Normal distribution takes two parameters: a mean parameter μ and a variance parameter σ^2.

Probability Density Functions: Examples

- Perhaps the most common distribution in statistics is the **Normal** distribution.
- The Normal distribution takes two parameters: a mean parameter μ and a variance parameter σ^2.

$$p(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$



Evaluating PMFs, PDFs, and CMFs

The most fundamental computation necessary when working with probability distributions is evaluating the distribution at different values. This computation can be difficult or costly for a number of reasons:

Evaluating PMFs, PDFs, and CMFs

The most fundamental computation necessary when working with probability distributions is evaluating the distribution at different values. This computation can be difficult or costly for a number of reasons:

- The PDF, PMF, or CMF involves a difficult to compute special function.

Evaluating PMFs, PDFs, and CMFs

The most fundamental computation necessary when working with probability distributions is evaluating the distribution at different values. This computation can be difficult or costly for a number of reasons:

- The PDF, PMF, or CMF involves a difficult to compute special function.
- Normalizing the distribution (ensuring the PDF/PMF integrates to 1) requires a difficult to compute sum or integral.

Evaluating PMFs, PDFs, and CMFs

The most fundamental computation necessary when working with probability distributions is evaluating the distribution at different values. This computation can be difficult or costly for a number of reasons:

- The PDF, PMF, or CMF involves a difficult to compute special function.
- Normalizing the distribution (ensuring the PDF/PMF integrates to 1) requires a difficult to compute sum or integral.
- Probabilities near zero or near one can cause numerical errors.

Calculating PDFs: The Gamma Function

Evaluating PDFs and PMFs, even common ones, often requires evaluating difficult to compute special functions. A particularly common function for continuous distributions is the Gamma function.

Calculating PDFs: The Gamma Function

Evaluating PDFs and PMFs, even common ones, often requires evaluating difficult to compute special functions. A particularly common function for continuous distributions is the Gamma function.

Gamma Function

The gamma function is defined as

$$\Gamma(t) = \int_0^\infty x^{t-1} e^{-x} dx$$

Calculating PDFs: The Gamma Function

Evaluating PDFs and PMFs, even common ones, often requires evaluating difficult to compute special functions. A particularly common function for continuous distributions is the Gamma function.

Gamma Function

The gamma function is defined as

$$\Gamma(t) = \int_0^\infty x^{t-1} e^{-x} dx$$

The Gamma function and related Incomplete Gamma and Incomplete Beta functions are necessary/useful for working with the following common distributions (among others).

- Gamma, Z, t, χ^2, F, binomial, Poisson

Calculating PDFs: The Gamma Function

Algorithms for computing the Gamma function have been heavily studied. The most commonly used algorithm is uses the Lanczos approximation.

Calculating PDFs: The Gamma Function

Algorithms for computing the Gamma function have been heavily studied. The most commonly used algorithm is uses the Lanczos approximation.

$$\Gamma(z+1) = \sqrt{2\pi}\left(z+g+\frac{1}{2}\right)^{z+1/2} e^{-(z+g+1/2)} A_g(z)$$

$$A_g(z) = c_0 + \frac{c_1}{z+1} + \frac{c_2}{z+2} + \frac{c_3}{z+3} + \ldots$$

Calculating PDFs: The Gamma Function

Algorithms for computing the Gamma function have been heavily studied. The most commonly used algorithm is uses the Lanczos approximation.

$$\Gamma(z+1) = \sqrt{2\pi}\left(z+g+\frac{1}{2}\right)^{z+1/2} e^{-(z+g+1/2)} A_g(z)$$

$$A_g(z) = c_0 + \frac{c_1}{z+1} + \frac{c_2}{z+2} + \frac{c_3}{z+3} + \ldots$$

Importantly, the user can choose g and pre-calculate the constants c_i.

Calculating PMFs: Using Recursion

- Often, when multiple values of a PMF are desired, we can take advantage of recurrence relations to avoid calculating costly special functions.

Calculating PMFs: Using Recursion

- Often, when multiple values of a PMF are desired, we can take advantage of recurrence relations to avoid calculating costly special functions.
- The binomial distribution is the distribution for the number of successes in a sequence of n yes/no experiments (e.g. coin flips) with success probability p. The binomial PMF is,

Calculating PMFs: Using Recursion

- Often, when multiple values of a PMF are desired, we can take advantage of recurrence relations to avoid calculating costly special functions.
- The binomial distribution is the distribution for the number of successes in a sequence of n yes/no experiments (e.g. coin flips) with success probability p. The binomial PMF is,

Calculating PMFs: Using Recursion

- Often, when multiple values of a PMF are desired, we can take advantage of recurrence relations to avoid calculating costly special functions.
- The binomial distribution is the distribution for the number of successes in a sequence of n yes/no experiments (e.g. coin flips) with success probability p. The binomial PMF is,

$$P(X = x; n, p) = \binom{n}{x} p^x (1-p)^{n-x}$$

Calculating PMFs: Using Recursion

Evaluating the binomial PMF involves evaluating factorials (a special case of the Gamma function); however, if multiple values are desired, we can take advantage of the following recurrence relation:

$$
\begin{aligned}
P(X = x) &= \binom{n}{x} p^x (1-p)^{n-x} \\
&= \frac{n-x}{x+1} \frac{p}{1-p} \left[\binom{n}{x-1} p^{x-1} (1-p)^{n-(x-1)} \right] \\
&= P(X = x-1) \frac{n-x}{x+1} \frac{p}{1-p}
\end{aligned}
$$

Calculating PMFs: Using Recursion

- Does utilizing this recursion improve the complexity or the constant?

Calculating PMFs: Using Recursion

- Does utilizing this recursion improve the complexity or the constant?
- The complexity of calculating the first k values of the binomial PMF is $\mathcal{O}(k)$ in both cases; however, we are replacing a special function with arithmetic operations only.

Working in log-space and the Log-Sum-Exp trick

- As we discussed in a previous lecture, exponentiating even moderate values can result in numerical overflow or underflow.

Working in log-space and the Log-Sum-Exp trick

- As we discussed in a previous lecture, exponentiating even moderate values can result in numerical overflow or underflow.
- Many distributions require exponentiation of intermediate values.

Working in log-space and the Log-Sum-Exp trick

- As we discussed in a previous lecture, exponentiating even moderate values can result in numerical overflow or underflow.
- Many distributions require exponentiation of intermediate values.
- One common solution is to work in **log-space**.

Working in log-space and the Log-Sum-Exp trick

The multinomial distribution is the standard distribution for finite sets. Consider the following multinomial distribution over K discrete values parameterized by a length K vector of weights α.

$$P(X = i; \alpha) = \frac{e^{\alpha_i}}{\sum_i e^{\alpha_i}}, \ i = 1, ..., K$$

Working in log-space and the Log-Sum-Exp trick

The multinomial distribution is the standard distribution for finite sets. Consider the following multinomial distribution over K discrete values parameterized by a length K vector of weights α.

$$P(X = i; \alpha) = \frac{e^{\alpha_i}}{\sum_i e^{\alpha_i}},\ i = 1, ..., K$$

Rather than evaluate this function directly, risking over/underflow, we can evaluate the log PMF,

$$\log P(X = i; \alpha) = \alpha_i - \log \sum_i e^{\alpha_i}$$

Working in log-space and the Log-Sum-Exp trick

The multinomial distribution is the standard distribution for finite sets. Consider the following multinomial distribution over K discrete values parameterized by a length K vector of weights α.

$$P(X = i; \alpha) = \frac{e^{\alpha_i}}{\sum_i e^{\alpha_i}}, \; i = 1, ..., K$$

Rather than evaluate this function directly, risking over/underflow, we can evaluate the log PMF,

$$\log P(X = i; \alpha) = \alpha_i - \log \sum_i e^{\alpha_i}$$

Why does this not completely solve our problem?

Working in log-space and the Log-Sum-Exp trick

Calculating the second term $\log \sum_i e^{\alpha_i}$ (also known as the cumulant function) still requires exponentiating α. Instead, let $\alpha^* = \max_i \alpha_i$. Then, we can use the following trick,

$$
\begin{aligned}
\log \sum_i e^{\alpha_i} &= \log \frac{e^{\alpha^*}}{e^{\alpha^*}} \sum_i e^{\alpha_i} \\
&= \log \sum_i e^{\alpha_i - \alpha^*} + \log e^{\alpha^*} \\
&= \log \sum_i e^{\alpha_i - \alpha^*} + \alpha^*
\end{aligned}
$$

Working in log-space and the Log-Sum-Exp trick

- Now we are guaranteed that the largest term in the sum equals one.

Working in log-space and the Log-Sum-Exp trick

- Now we are guaranteed that the largest term in the sum equals one.
- There can be no overflow and, while individual terms in the sum may underflow, this results only in round-off error in the final sum.

Working in log-space and the Log-Sum-Exp trick

- Now we are guaranteed that the largest term in the sum equals one.
- There can be no overflow and, while individual terms in the sum may underflow, this results only in round-off error in the final sum.
- Log-sum-exp is implemented in `scipy.misc.logsumexp`.