Lecture 14

RNA-seq: Quantification

Count-based model

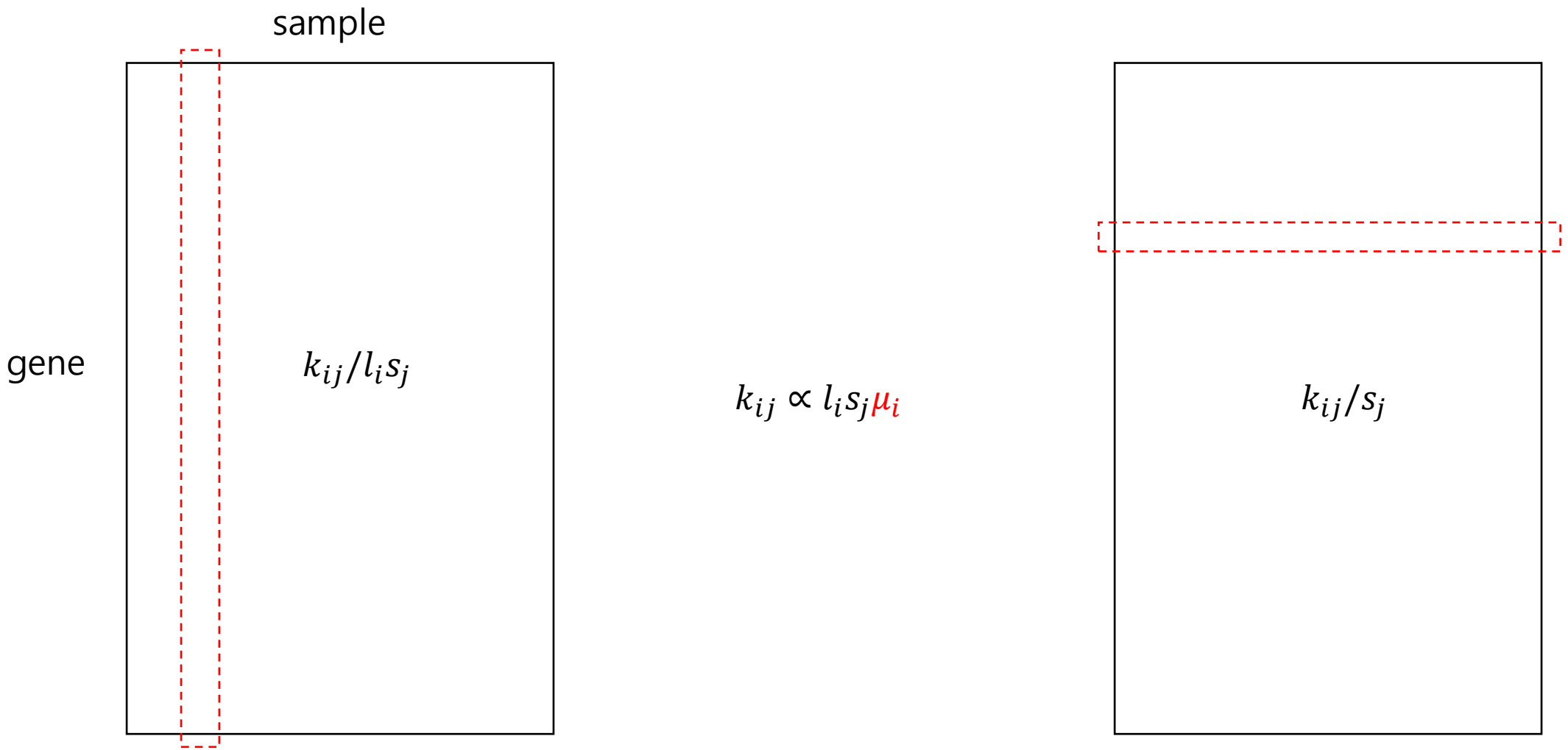
- We want to quantify the relative transcript abundance from mapped read counts.

$$k_{ij} \propto l_i s_j \mu_i$$

- Count-based model: the total number of reads mapping to a region, normalized by length and total number of reads in the experiment, is used as a proxy for abundance.

- This intuitive measure is based on an underlying model that is multinomial, and the normalized counts can be understood as the maximum likelihood estimate of parameters corresponding to relative transcript abundances based on the model.

Normalization



RPKM/FPKM/TPM

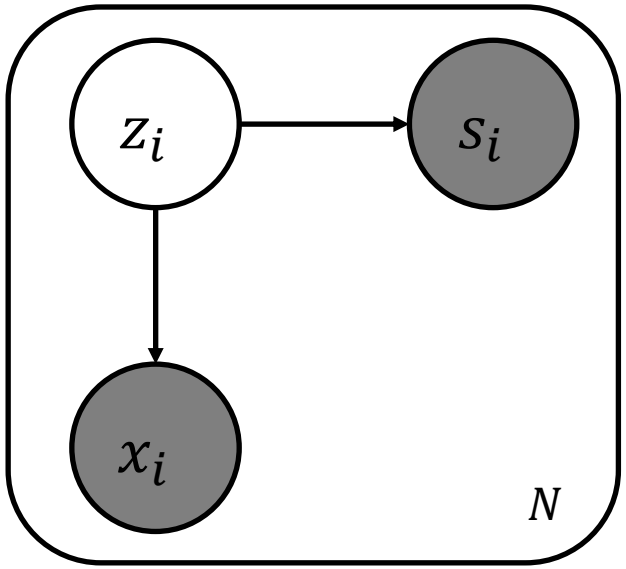
- Reads/Fragments per kilobase of transcript per millions of read mapped

$$\frac{10^9 k_i}{l_i \sum_j k_j}$$

- TPM: Transcripts per million

$$\frac{10^6 k_i / l_i}{\sum_j k_j / l_j}$$

A generative model



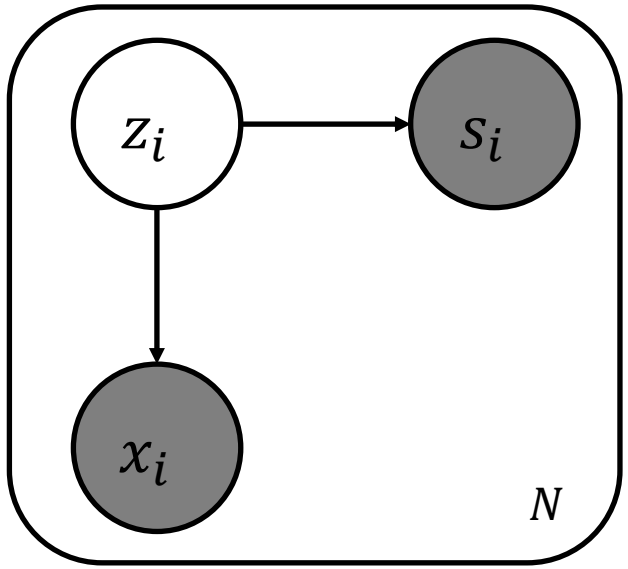
Relative abundance of transcript: $\mu = (\mu_1, \ldots, \mu_K)^T$, and the parameters μ_k are constrained to satisfy $\mu_k \geq 0$ and $\sum_{k=1}^{K} \mu_k = 1$.

$z_i \in [1, K]$: the transcript from which read i is derived.

s_i: read start position

$x_i \in [1, K]$: the transcript to which read i is uniquely mapped.

A generative model



$$p(z_{ik} = 1|\mu) = \frac{\mu_k \tilde{l}_k}{\sum_{k'=1}^{K} \mu_{k'} \tilde{l}_{k'}} = \alpha_k$$

$\tilde{l}_k = l_k - m + 1$

l_k is the length of transcript k

m is the length of reads

$$p(s_i = j|z_{ik} = 1) = \frac{1}{\tilde{l}_k}$$

$$p(x_i|z_i) = 1(x_i = z_i)$$

A generative model

$p(x_i|z_i)$ by conditional independence

- $p(x_i, s_i|\mu)$:

$$p(x_i, s_i) = \sum_{z_i} p(x_i|z_i, s_i)p(s_i|z_i)p(z_i)$$

$$= \sum_{z_i} p(x_i|z_i)p(s_i|z_i)p(z_i)$$

$$= \sum_{z_i} 1(x_i = z_i) \prod_{k'} \frac{\alpha_{k'}^{z_{ik'}}}{\tilde{l}_{k'}^{z_{ik'}}} = \prod_{k'} \frac{\alpha_{k'}^{x_{ik'}}}{\tilde{l}_{k'}^{x_{ik'}}}$$

Likelihood function

- The likelihood of observing the reads is

$$L(\mu) = \prod_{i=1}^{N} p(x_i, s_i|\mu) = \prod_{i=1}^{N} \prod_{k=1}^{K} \frac{\alpha_k^{x_{ik}}}{\tilde{l}_k^{x_{ik}}} = \prod_{k=1}^{K} \left(\frac{\alpha_k}{\tilde{l}_k}\right)^{x_k}$$

$$x_k = \sum_{i=1}^{N} x_{ik}$$

Maximum likelihood estimate

- We need to maximize $\ln p(X, S|\alpha)$ with respect to α_k taking account of the constraint that the α_k must sum to 1.

- This can be achieved using a Lagrange multiplier λ and maximizing

$$\sum_{k=1}^{K} x_k \log \alpha_k + \lambda \left(\sum_{k=1}^{K} \alpha_k - 1 \right)$$

Maximum likelihood estimate

- Setting the derivative of the Lagrangian function with respect to α_k to zero, we obtain

$$\alpha_k = -\frac{x_k}{\lambda}$$

- We can solve for the Lagrange multiplier λ by substituting $\alpha_k = -\frac{x_k}{\lambda}$ into the constraint $\sum_{k=1}^{K} \alpha_k = 1$ to give $\lambda = -N$.

- We obtain the maximum likelihood solution in the form

$$\hat{\alpha}_k = \frac{x_k}{N}$$

From α_k to μ_k

$$\alpha_k = \frac{\mu_k \tilde{l}_k}{\sum_{k'=1}^{K} \mu_{k'} \tilde{l}_{k'}}$$

$$\sum_{k=1}^{K} \frac{\alpha_k}{\tilde{l}_k} = \sum_{k=1}^{K} \frac{\mu_k}{\sum_{k'=1}^{K} \mu_{k'} \tilde{l}_{k'}} = \frac{1}{\mu_k \tilde{l}_k / \alpha_k} = \frac{\alpha_k}{\mu_k \tilde{l}_k}$$

$$\mu_k = \frac{\frac{\alpha_k}{\tilde{l}_k}}{\sum_{k'=1}^{K} \frac{\alpha_{k'}}{\tilde{l}_{k'}}}$$

$\hat{\mu}_k$

$$\hat{\mu}_k = \frac{\frac{\hat{\alpha}_k}{\tilde{l}_k}}{\sum_{k'=1}^{K} \frac{\hat{\alpha}_{k'}}{\tilde{l}_{k'}}} = \frac{x_k}{N\tilde{l}_k} \frac{1}{\sum_{k'=1}^{K} \frac{x_{k'}}{N\tilde{l}_{k'}}} = \frac{x_k}{\tilde{l}_k} \frac{1}{\sum_{k'=1}^{K} \frac{x_{k'}}{\tilde{l}_{k'}}}$$

TPM and $\hat{\mu}_k$

- TPM:

$$10^6 \frac{x_k}{\tilde{l}_k} \frac{1}{\sum_{k'=1}^{K} \frac{x_{k'}}{\tilde{l}_{k'}}}$$

- TPM $= 10^6 \hat{\mu}_k$

RPKM and $\hat{\mu}_k$

- RPKM

$$\frac{x_k}{\frac{N}{10^6}\frac{\tilde{l}_k}{10^3}} = \frac{x_k}{N\tilde{l}_k}10^9$$

- RPKM is not properly normalized.

RPKM and $\hat{\mu}_k$

- The normalization factor of TPM

$$\frac{1}{\sum_{k'=1}^{K}\frac{x_{k'}}{\tilde{l}_{k'}}} = \frac{\hat{\mu}_k}{\frac{x_k}{\tilde{l}_k}} = \frac{\hat{\mu}_k}{\frac{N\hat{\alpha}_k}{\tilde{l}_k}} = \frac{\hat{\mu}_k\tilde{l}_k}{N\hat{\alpha}_k} = \frac{\hat{\mu}_k\tilde{l}_k}{\frac{N\hat{\mu}_k\tilde{l}_k}{\sum_{k'=1}^{K}\hat{\mu}_{k'}\tilde{l}_{k'}}} = \frac{1}{N}\sum_{k'=1}^{K}\hat{\mu}_{k'}\tilde{l}_{k'}$$

is proportional to the mean length of transcripts in the transcriptome.

- Because the mean expressed transcript length may vary between samples, we prefer TPM over RPKM measures.

Statistical inference

The rules of probability

$$\textbf{sum rule} \qquad p(X) = \sum_{Y} p(X, Y)$$

$$\textbf{product rule} \qquad p(X, Y) = p(Y|X)p(X).$$

$p(X, Y)$: joint probability, "the probability of X and Y"

$p(Y|X)$: conditional probability, "the probability of Y given X"

$p(X)$: marginal probability, "the probability of X"

Bayes' theorem

$$p(Y|X) = \frac{p(X|Y)p(Y)}{p(X)}$$

$$p(X) = \sum_{Y} p(X|Y)p(Y)$$

Bayes' theorem

$$p(\mathbf{w}|\mathcal{D}) = \frac{p(\mathcal{D}|\mathbf{w})p(\mathbf{w})}{p(\mathcal{D})}$$

$p(\boldsymbol{w})$: prior probability

$p(D|\boldsymbol{w})$: likelihood function. It expresses how probable the observed data set is for different settings of the parameter vector $\boldsymbol{w}$. The likelihood is not a probability distribution over $\boldsymbol{w}$.

Bayes' theorem allows us to evaluate the uncertainty in $\boldsymbol{w}$ after we have observed D in the form of the posterior probability $p(\boldsymbol{w}|D)$.

$$\text{posterior} \propto \text{likelihood} \times \text{prior}$$

Bayesian and frequentist approaches

- In a frequentist setting:

$\boldsymbol{w}$ is considered to be a fixed parameter, whose value is determined by some form of estimator, and error bars on this estimate are obtained by considering the distribution of possible data sets D.

- In a Bayesian setting:

There is only a single data set D (the one that is actually observed), and the uncerainty in $\boldsymbol{w}$ is expressed through $p(\boldsymbol{w})$.

Maximum likelihood

- A widely used frequentist estimator is maximum likelihood, in which $\boldsymbol{w}$ is set to the value that maximizes the likelihood function $p(D|\boldsymbol{w})$. This corresponds to choosing the value of $\boldsymbol{w}$ for which the probability of the observed data set is maximized.

- In the machine learning literature, the negative log of the likelihood function is called an error function.

Parameter estimation

- Parametric distributions: governed by a small number of adaptive parameters.

- Parameter estimation: a procedure for determining suitable values for the parameters given an observed data set.
  - In a frequentist treatment: we choose specific values for the parameters by optimizing some criterion, such as the likelihood function.
  - In a Bayesian treatment: we introduce prior distributions over the parameters and then use Bayes' theorem to compute the corresponding posterior distribution given the observed data.

Conjugate prior

- Conjugate priors: lead to posterior distributions having the same functional form as the prior, and that therefore lead to a greatly simplified Bayesian analysis.

- For example, the conjugate prior for the parameters of the multinomial distribution is called the Dirichlet distribution.

Binary variables

- Consider a single binary random variable $x \in \{0,1\}$. The probability of $x = 1$ will be denoted by the parameter μ so that

$$p(x = 1|\mu) = \mu$$

where $0 \leq \mu \leq 1$, from which it follows that $p(x = 0|\mu) = 1 - \mu$.

- The probability distribution over x can therefore be written in the form

$$Bern(x|\mu) = \mu^x(1 - \mu)^{1-x}$$

which is known as the Bernoulli distribution.

Binary variables

- The mean and variance are given by

$$E[x] = \mu, Var[x] = \mu(1-\mu)$$

- Suppose we have a data set $D = \{x_1, \dots, x_N\}$ of observed values of x. We can construct the likelihood function, which is a function of μ, on the assumption that the observations are drawn independently from $p(x|\mu)$, so that

$$p(\mathcal{D}|\mu) = \prod_{n=1}^{N} p(x_n|\mu) = \prod_{n=1}^{N} \mu^{x_n}(1-\mu)^{1-x_n}$$

Binary variables

- In a frequentist setting, we can estimate a value for μ by maximizing the likelihood function, or equivalently by maximizing the logarithm of the likelihood. The log likelihood function is given by

$$\ln p(\mathcal{D}|\mu) = \sum_{n=1}^{N} \ln p(x_n|\mu) = \sum_{n=1}^{N} \{x_n \ln \mu + (1 - x_n) \ln(1 - \mu)\}$$

- If we set the derivative of the log likelihood function with respect to μ equal to zero, we obtain the maximum likelihood estimator:

$$\mu_{\mathrm{ML}} = \frac{1}{N} \sum_{n=1}^{N} x_n$$

Multinomial variables

- Binary variables can be used to describe quantities that can take one of two possible values. Often we encounter discrete variables that can take on one of K possible mutually exclusive states.

- Although there are various alternative ways to express such variables, we shall see that a particularly convenient representation is the 1-of-K scheme in which the variable is represented by a K-dimensional vector x in which one of the elements x_k equals 1, and all remaining elements equal 0.

Multinomial variables

- For instance if we have a variable that can take $K = 6$ states and a particular observation of the variable happens to correspond to the state where $x_3 = 1$, then $\boldsymbol{x}$ will be represented by

$$\mathbf{x} = (0, 0, 1, 0, 0, 0)$$

- Note that such vectors satisfy $\sum_{k=1}^{K} x_k = 1$. If we denote the probability of $x_k = 1$ by the parameter μ_k, then the distribution of $\boldsymbol{x}$ is given

$$p(\mathbf{x}|\boldsymbol{\mu}) = \prod_{k=1}^{K} \mu_k^{x_k}$$

where $\mu = (\mu_1, \ldots, \mu_K)^T$, and the parameters μ_k are constrained to satisfy $\mu_k \geq 0$ and $\sum_{k=1}^{K} \mu_k = 1$.

Multinomial variables

- It is easily seen that the distribution is normalized:

$$\sum_{\mathbf{x}} p(\mathbf{x}|\boldsymbol{\mu}) = \sum_{k=1}^{K} \mu_k = 1$$

$$\mathbb{E}[\mathbf{x}|\boldsymbol{\mu}] = \sum_{\mathbf{x}} p(\mathbf{x}|\boldsymbol{\mu})\mathbf{x} = (\mu_1, \ldots, \mu_M)^{\mathrm{T}} = \boldsymbol{\mu}$$

Multinomial variables

- Consider a data set D of N independent observations $x_1, \ldots, x_N$. The corresponding likelihood function takes the form:

$$p(\mathcal{D}|\boldsymbol{\mu}) = \prod_{n=1}^{N}\prod_{k=1}^{K}\mu_k^{x_{nk}} = \prod_{k=1}^{K}\mu_k^{(\sum_n x_{nk})} = \prod_{k=1}^{K}\mu_k^{m_k}$$

$$m_k = \sum_n x_{nk}$$

Lagrange multipliers

- Lagrange multipliers are used to find the stationary points of a function of several variables subject to one or more constraints.

- Consider the problem of finding the maximum of a function $f(\boldsymbol{x})$ subject to a constraint $g(\boldsymbol{x}) = 0$.

- It is convenient to introduce a Lagrangian function defined by

$$L(x, \lambda) \stackrel{\text{def}}{=} f(x) + \lambda g(\boldsymbol{x})$$

Lagrange multipliers

- To find the maximum of a function $f(\boldsymbol{x})$ subject to the constraint $g(\boldsymbol{x}) = 0$, we define the Lagrangian function and we then find the stationary point of $L(\boldsymbol{x}, \lambda)$ with respect to both $\boldsymbol{x}$ and λ.

- The stationary point $\boldsymbol{x}^*$ is obtained by setting $\nabla_x L = 0$. The condition $\frac{\partial L}{\partial \lambda} = 0$ leads to the constraint equation $g(\boldsymbol{x}) = 0$.

- If we are only interested in $\boldsymbol{x}^*$, then we can eliminate λ from the stationary equations without needing to find its value.

Multinomial variables

- To find the maximum likelihood solution for μ, we need to maximize $\ln p(D|\mu)$ with respect to μ_k taking account of the constraint that the μ_k must sum to 1.

- This can be achieved using a Lagrange multiplier λ and maximizing

$$\sum_{k=1}^{K} m_k \ln \mu_k + \lambda \left(\sum_{k=1}^{K} \mu_k - 1 \right)$$

Multinomial variables

- Setting the derivative of the Lagrangian function with respect to μ_k to zero, we obtain

$$\mu_k = -\frac{m_k}{\lambda}$$

- We can solve for the Lagrange multiplier λ by substituting $\mu_k = -\frac{m_k}{\lambda}$ into the constraint $\sum_{k=1}^{K} \mu_k = 1$ to give $\lambda = -N$.

- We obtain the maximum likelihood solution in the form

$$\mu_k^{ML} = \frac{m_k}{N}$$

Conditional independence

- An important concept for probability distributions over multiple variables is that of conditional independence.

- Consider three variables a, b, and c, and suppose that the conditional distributions of a, given b and c, is such that it does not depend on the values of b, so that

$$p(a|b,c) = p(a|c)$$

- We say that a is conditionally independent of b given c.

Conditional independence

- This can be expressed in a slightly different way if we consider the joint distribution of a and b conditioned on c.

$$\begin{aligned} p(a, b|c) &= p(a|b, c)p(b|c) \\ &= p(a|c)p(b|c). \end{aligned}$$

- We see that, conditioned on c, the joint distribution of a and b factorizes into the product of the marginal distribution of a and the marginal distribution of b. This says that the variables a and b are statistically independent given c.

Conditional independence

- We use a shorthand notation for conditional independence in which

$$a \perp\!\!\!\perp b \mid c$$

denotes that a is conditionally independent of b given c.

Graphical models

- Probabilistic graphical models (PGMs), diagrammatic representations of probability distributions, offer several useful properties:
  - They provide a simple way to visualize the structure of a probabilistic model and can be used to design and motivate new models.
  - Insights into the properties of the model, including conditional independence properties, can be obtained by inspection of the graph.
  - Complex computations, required to perform inference and learning in sophisticated models, can be expressed in terms of graphical manipulations, in which underlying mathematical expressions are carried along implicitly.

Graphical models

- A graph comprises nodes (also called vertices) connected by links (also known as edges).
- In a PGM, each node represents a random variable, and the links express probabilistic relationships between these variables.
- The graph captures the way in which the joint distribution over all of the random variables can be decomposed into a product of factors each depending only on a subset of the variables.

Graphical models

- Directed graphical models (also known as Bayesian networks): the links of the graphs have a particular directionality indicated by arrows. Useful for expressing causal relationships between random variables.

- Undirected graphical models (also known as Markov random fields): the links do not carry arrows and have no directional significance. Better suited to expressing soft constraints between random variables.

Bayesian networks

- Consider the joint distribution over K variables given by $p(x_1, \ldots, x_K)$. By repeated application of the product rule of probability, this joint distribution can be written as a product of conditional distributions:

$$p(x_1, \ldots, x_K) = p(x_K|x_1, \ldots, x_{K-1}) \ldots p(x_2|x_1)p(x_1)$$

- We can represent this as a directed graph having K nodes, one for each conditional distribution, with each node having incoming links from all lower numbered nodes. We say this graph is fully connected because there is a link between every pair of nodes.

Bayesian networks

- It is the absence of links in the graph that conveys interesting information about the properties of the class of distributions that the graph represents.

- We now go from this graph to the corresponding representation of the joint probability distribution written in terms of the product of a set of conditional distributions, one for each node in the graph. Each such conditional distribution will be conditioned only on the parents of the corresponding node in the graph.

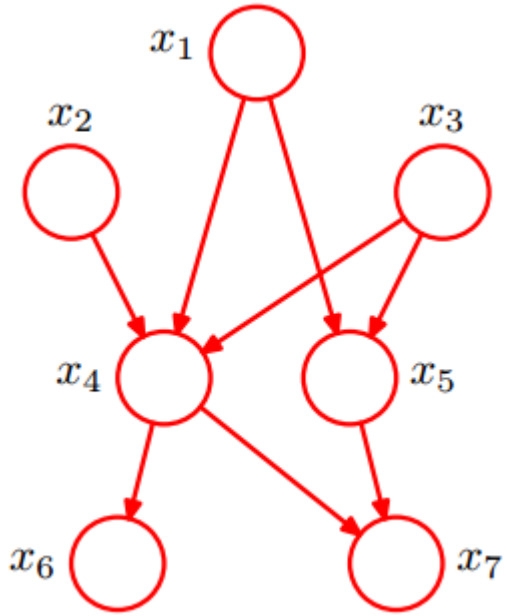
Bayesian networks

- The joint distribution defined by a graph is given by the product, over all of the nodes of the graph, of a conditional distribution for each node conditioned on the variables corresponding to the parents of that node in the graph.

- For a graph with K nodes, the joint distribution is given by

$$p(\mathbf{x}) = \prod_{k=1}^{K} p(x_k|\mathrm{pa}_k)$$

Bayesian networks



$$p(x_1)p(x_2)p(x_3)p(x_4|x_1,x_2,x_3)p(x_5|x_1,x_3)p(x_6|x_4)p(x_7|x_4,x_5)$$

Bayesian networks

- This key equation expresses the factorization properties of the joint distribution for a directed graphical model.

- It is easy to show that the representation on the right-hand side is always correctly normalized provided the individual conditional distributions are normalized.

- The directed graphs that we are considering are subject to an important restriction namely that there must be no directed cycles. Such graphs are called directed acyclic graphs, or DAGs.

D-separation

- Consider a general directed graph in which A, B, and C are arbitrary nonintersecting sets of nodes. We wish to ascertain whether a particular conditional independence statement $A \perp B|C$ is implied by a given DAG.

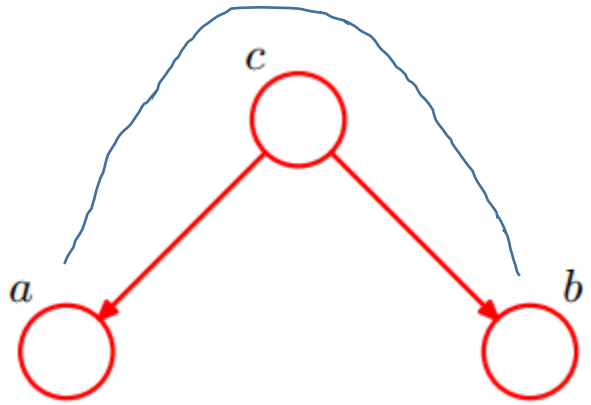
- To do so, we consider all possible paths from any node in A to any node in B. Any such path is said to be blocked if it includes a node such that either

(a) the arrows on the path meet either head-to-tail or tail-to-tail at the node, and the node is in the set C, or

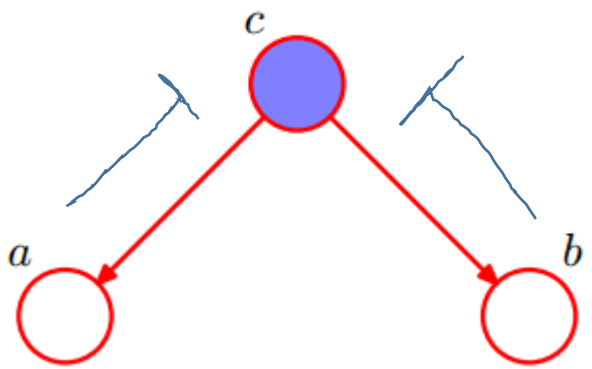
(b) the arrows meet head-to-head at the node, and neither the node, nor any of its descendants, is in the set C.

D-separation

- If all paths are blocked, then A is said to be d-separated from B by C, and the joint distribution over all of the variables in the graph will satisfy $A \perp B|C$.

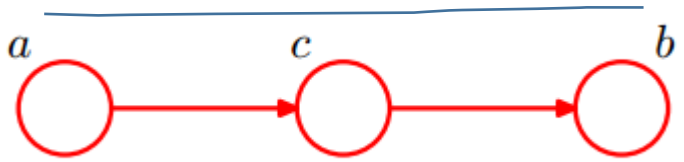


$a \not\perp\!\!\!\perp b \mid \emptyset$

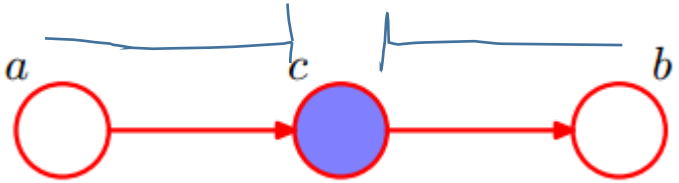


$a \perp\!\!\!\perp b \mid c.$

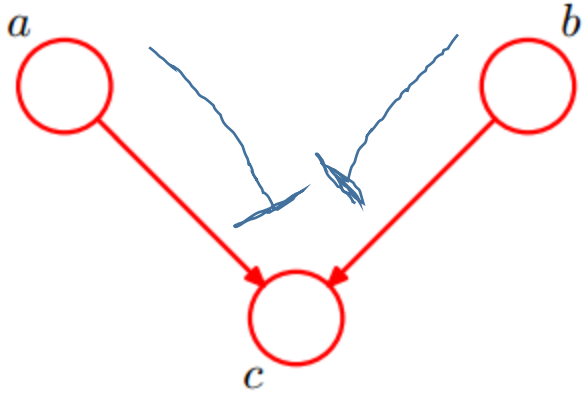
D-separation



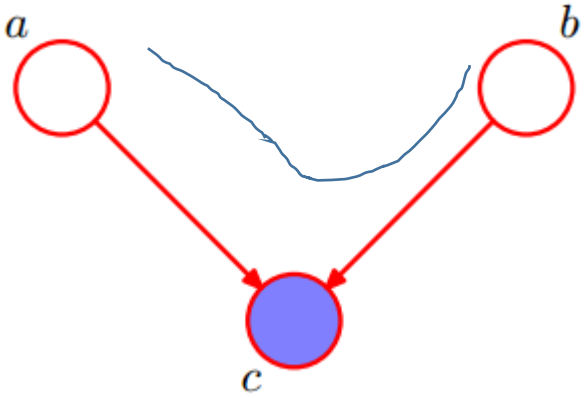
$a \not\perp\!\!\!\perp b \mid \emptyset$



$a \perp\!\!\!\perp b \mid c.$

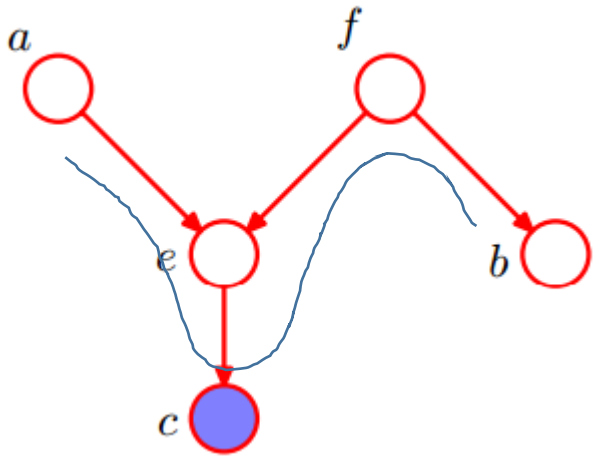


$a \perp\!\!\!\perp b \mid \emptyset.$



$a \not\perp\!\!\!\perp b \mid c.$

D-separation



$$a \not\perp\!\!\!\perp b \mid c.$$

Calculating read counts and TPM

HTSeq

BIOINFORMATICS ORIGINAL PAPER

Vol. 31 no. 2 2015, pages 166–169
doi:10.1093/bioinformatics/btu638

Genome analysis

Advance Access publication September 25, 2014

HTSeq—a Python framework to work with high-throughput sequencing data

Simon Anders*, Paul Theodor Pyl and Wolfgang Huber

Genome Biology Unit, European Molecular Biology Laboratory, 69111 Heidelberg, Germany

Associate Editor: Michael Brudno

htseq-count

```
conda activate python-3.6
```

htseq-count

```
#!/bin/bash
set -e
set -u
set -o pipefail

sample_list=('SRR3191542' 'SRR3191543' 'SRR3191544' 'SRR3191545' 'SRR3194428' 'SRR3194429' 'SRR3194430' 'SRR3194431')
num_sample=${#sample_list[@]}
for (( i=0;i<$num_sample;i++ )); do
        htseq-count -m union -s no -a 0 -i gene_id -f bam \
        /home/username/star/${sample_list[${i}]}_Aligned.out.bam \
        /home/reference/human/Homo_sapiens.GRCh38.108.gtf > /home/username/htseq/${sample_list[${i}]}.htseqcount.txt
done
```


Should make directory called “htseq”
in your home directory

featureCounts

BIOINFORMATICS ORIGINAL PAPER

Vol. 30 no. 7 2014, pages 923–930
doi:10.1093/bioinformatics/btt656

Sequence analysis

Advance Access publication November 13, 2013

featureCounts: an efficient general purpose program for assigning sequence reads to genomic features

Yang Liao[1,2], Gordon K. Smyth[1,3] and Wei Shi[1,2,*]

[1]Bioinformatics Division, The Walter and Eliza Hall Institute of Medical Research, 1G Royal Parade, Parkville, VIC 3052, [2]Department of Computing and Information Systems and [3]Department of Mathematics and Statistics, The University of Melbourne, Parkville, VIC 3010, Australia

Associate Editor: Martin Bishop

http://bioinf.wehi.edu.au/subread-package/SubreadUsersGuide.pdf

Deactivate virtual environment to base

```
conda deactivate
```

featureCounts: single-end reads

```
#!/bin/bash
set -e
set -u
set -o pipefail

sample_list=('SRR3194428' 'SRR3194429' 'SRR3194430' 'SRR3194431')
num_sample=${#sample_list[@]}

for (( i=0;i<$num_sample;i++ ));
do
        featureCounts -T 5 -t exon -g gene_id -a /home/reference/human/Homo_sapiens.GRCh38.108.gtf \
        -o /home/username/ftCounts/ftCounts_${sample_list[${i}]} /home/username/star/${sample_list[${i}]}_Aligned.out.bam
done
```


Should make directory called "ftCounts"
in your home directory

featureCounts: paired-end reads

```
#!/bin/bash
set -e
set -u
set -o pipefail

sample_list=('SRR3191542' 'SRR3191543' 'SRR3191544' 'SRR3191545')
num_sample=${#sample_list[@]}

for (( i=0;i<$num_sample;i++ ));
do
        featureCounts -p -T 5 -t exon -g gene_id -a /home/reference/human/Homo_sapiens.GRCh38.108.gtf \
        -o /home/username/ftCounts/ftCounts_${sample_list[${i}]} /home/username/star/${sample_list[${i}]}_Aligned.out.bam
done
```

TPMCalculator

Bioinformatics, 35(11), 2019, 1960–1962
doi: 10.1093/bioinformatics/bty896
Advance Access Publication Date: 31 October 2018
Applications Note

Genome analysis

TPMCalculator: one-step software to quantify mRNA abundance of genomic features

Roberto Vera Alvarez[1,*], Lorinc Sandor Pongor[1,2], Leonardo Mariño-Ramírez[1] and David Landsman[1]

[1]Computational Biology Branch, National Center for Biotechnology Information, National Library of Medicine, National Institutes of Health, Bethesda, MD 20892, USA and [2]2nd Department of Pediatrics, Semmelweis University, Budapest 1094, Hungary

*To whom correspondence should be addressed.

Associate Editor: Bonnie Berger

Received on March 13, 2018; revised on October 2, 2018; editorial decision on October 22, 2018; accepted on October 30, 2018

TPMCalculator Usage

Usage: ./bin/TPMCalculator -g GTF_file [-d BAM_files_directory|-b BAM_file]

```
./bin/TPMCalculator options:

    -v    Print info
    -h    Display this usage information.
    -g    GTF file
    -d    Directory with the BAM files
    -b    BAM file
    -k    Gene key to use from GTF file. Default: gene_id
    -t    Transcript key to use from GTF file. Default: transcript_id
    -c    Smaller size allowed for an intron created for genes. Default: 16. We recommend to use the reads length
    -p    Use only properly paired reads. Default: No. Recommended for paired-end reads.
    -q    Minimum MAPQ value to filter out reads. Default: 0. This value depends on the aligner MAPQ value.
    -o    Minimum overlap between a reads and a feature. Default: 8.
    -e    Extended output. This will include transcript level TPM values. Default: No.
    -a    Print out all features with read counts equal to zero. Default: No.
```

TPMCalculator: single-end reads

*Should make a directory for saving TPMCalculator output"
in your home directory & run this bash file in that directory

```
#!/bin/bash
set -e
set -u
set -o pipefail

sample_list=('SRR3194428' 'SRR3194429' 'SRR3194430' 'SRR3194431')
num_sample=${#sample_list[@]}

for (( i=0;i<$num_sample;i++ ));
do
        TPMCalculator -b /home/username/star/${sample_list[${i}]}_Aligned.out.bam \
         -g /home/reference/human/Homo_sapiens.GRCh38.108.gtf -a
done
```

TPMCalculator: paired-end reads

```
#!/bin/bash
set -e
set -u
set -o pipefail

sample_list=('SRR3191542' 'SRR3191543' 'SRR3191544' 'SRR3191545')
num_sample=${#sample_list[@]}

for (( i=0;i<$num_sample;i++ ));
do
        TPMCalculator -p -b /home/username/star/${sample_list[${i}]}_Aligned.out.bam \
         -g /home/reference/human/Homo_sapiens.GRCh38.108.gtf -a
done
```