Generative Models for Discrete Data

Brandon Kozak

15/09/2019

```r
library(tidyverse)
```

```
## -- Attaching packages ------------------------- tidyverse 1.2.1 --

## v ggplot2 3.2.1     v purrr   0.3.2
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

## -- Conflicts ---------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(BiocManager)
library(Biostrings)
```

```
## Loading required package: BiocGenerics

## Loading required package: parallel

##
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB

## The following objects are masked from 'package:dplyr':
##
##     combine, intersect, setdiff, union

## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs

## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, basename, cbind,
##     colnames, dirname, do.call, duplicated, eval, evalq, Filter,
##     Find, get, grep, grepl, intersect, is.unsorted, lapply, Map,
##     mapply, match, mget, order, paste, pmax, pmax.int, pmin,
##     pmin.int, Position, rank, rbind, Reduce, rownames, sapply,
##     setdiff, sort, table, tapply, union, unique, unsplit, which,
##     which.max, which.min
## Loading required package: S4Vectors

## Loading required package: stats4

##
## Attaching package: 'S4Vectors'

## The following objects are masked from 'package:dplyr':
##
##     first, rename

## The following object is masked from 'package:tidyr':
##
##     expand

## The following object is masked from 'package:base':
##
##     expand.grid

## Loading required package: IRanges

##
## Attaching package: 'IRanges'

## The following objects are masked from 'package:dplyr':
##
##     collapse, desc, slice

## The following object is masked from 'package:purrr':
##
##     reduce

## The following object is masked from 'package:grDevices':
##
##     windows

## Loading required package: XVector

##
## Attaching package: 'XVector'

## The following object is masked from 'package:purrr':
##
##     compact

##
## Attaching package: 'Biostrings'

## The following object is masked from 'package:base':
##
##     strsplit
```

```r
library(BSgenome.Celegans.UCSC.ce2)
```

```
## Loading required package: BSgenome

## Loading required package: GenomeInfoDb

## Loading required package: GenomicRanges

## Loading required package: rtracklayer
```

Exercises

1.1

Geometric Distribution:

Given a probability p, how many failures will it take to see the first success?

```r
# A random sample of size 5 from a geometric distribution with p=.25
rgeom(5, .25)
```

```
## [1]  0 10  1  5  0
```

```r
# What is the probability that we will see 4 failures before the first success?
dgeom(4, .25)
```

```
## [1] 0.07910156
```

```r
# What is the probability that we will see no more than 3 failures before the first success?
pgeom(3, .25)
```

```
## [1] 0.6835938
```

Hypergeometric Distribution:

Given a population of size N where K of the N objects are "success states." How many success state objects will I obtain (k) from drawing a sample of size n without replacement?

```r
# A random sample of size 5 from a hyper geometric distribution with a population of N=25, K=5 success
# given a sample of n=10
rhyper(5, 5, 20, 10)
```

```
## [1] 4 1 3 1 0
```

```r
# What is the probability that we will see 5 success state objects?
dhyper(5, 5, 20, 10)
```

```
## [1] 0.004743083
```

```r
# What is the probability that we will see at least 1 success state object?
phyper(0, 5, 20, 10, lower.tail = F)
```

```
## [1] 0.9434783
```

1.2

P(X = 2 | X ~ Bin(10, .3))

```r
dbinom(x = 2, size = 10, p = .3)
```

```
## [1] 0.2334744
```

P(X <= 2 | X ~ Bin(10, .3))

```r
# Using only dbinom()


dbinom(x = 0, size = 10, p = .3) + dbinom(x = 1, size = 10, p = .3) + dbinom(x = 2, size = 10, p = .3)
```

```
## [1] 0.3827828
```

```r
# Using pbinom()
pbinom(q = 2, size = 10, p = .3)
```

```
## [1] 0.3827828
```

1.3

```r
pois_max = function(n, max, lamda) {
  # First calculate P(X >= max) = 1 - P(X <= max - 1)
  prob = ppois(max-1, lamda)
  # Then, as we showed before using order statistics, calculate P(X(n) >= max) = P(X(n) <= max-1)
  prob_max = 1 - prob^n
  return(prob_max)
}
```

1.4

```r
pois_max = function(n = 100, max = 0, lamda = 1) {
  # First calculate P(X >= max) = 1 - P(X <= max - 1)
  prob = ppois(max-1, lamda)
  # Then, as we showed before using order statistics, calculate P(X(n) >= max) = P(X(n) <= max-1)
  prob_max = 1 - prob^n
  return(prob_max)
}
```

1.5

```r
# Real answer
pois_max(100, 9, .5)
```

```
## [1] 3.43549e-07
```

```r
# Simulation

pois_has_max = function(n, max, lamda) {

result_vector = rpois(n, lamda)

return(max(result_vector >= max))
}

a = replicate(1e5, pois_has_max(100, 9, .5))
mean(a)
```

```
## [1] 0
```

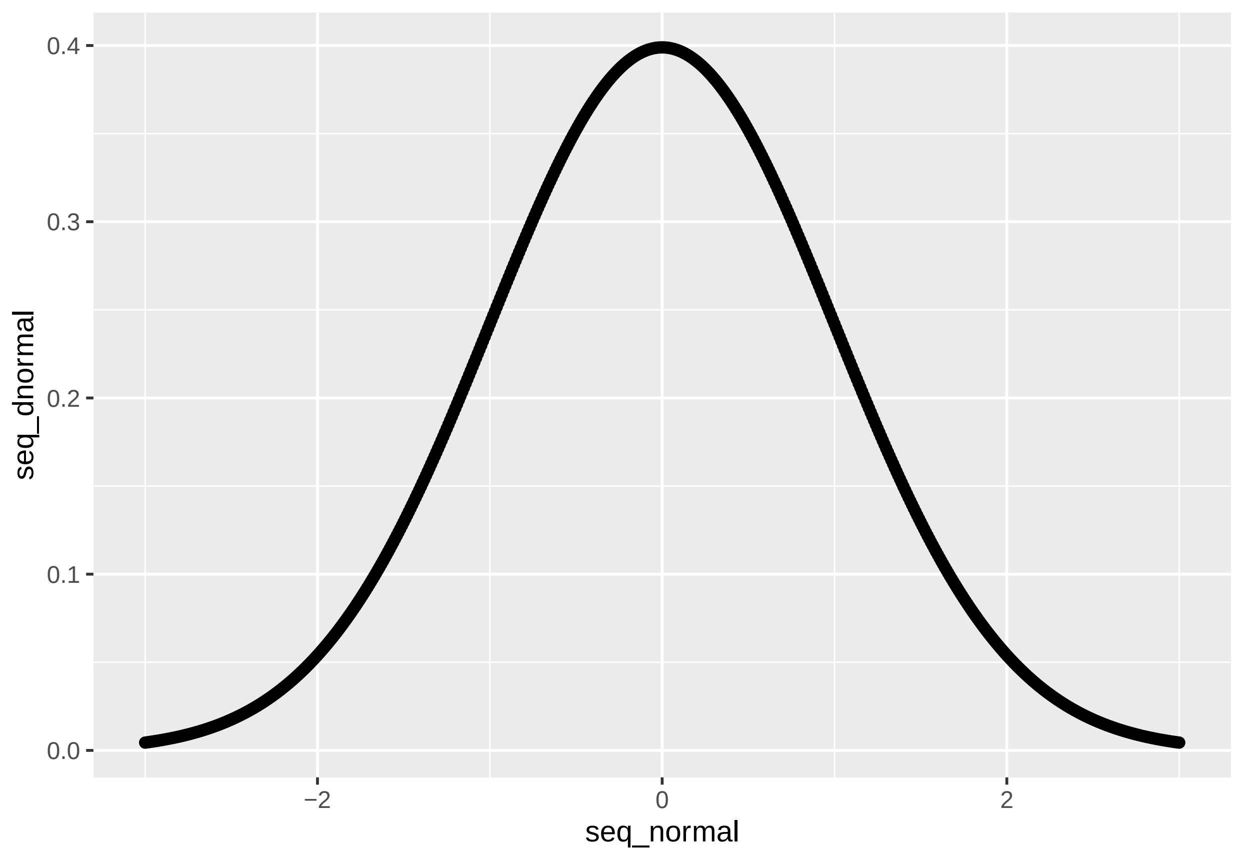
1.6

```r
# Standard normal

seq_normal = seq(-3, 3, .01)

seq_dnormal = dnorm(seq_normal, 0, 1)

qplot(x = seq_normal, y = seq_dnormal)
```

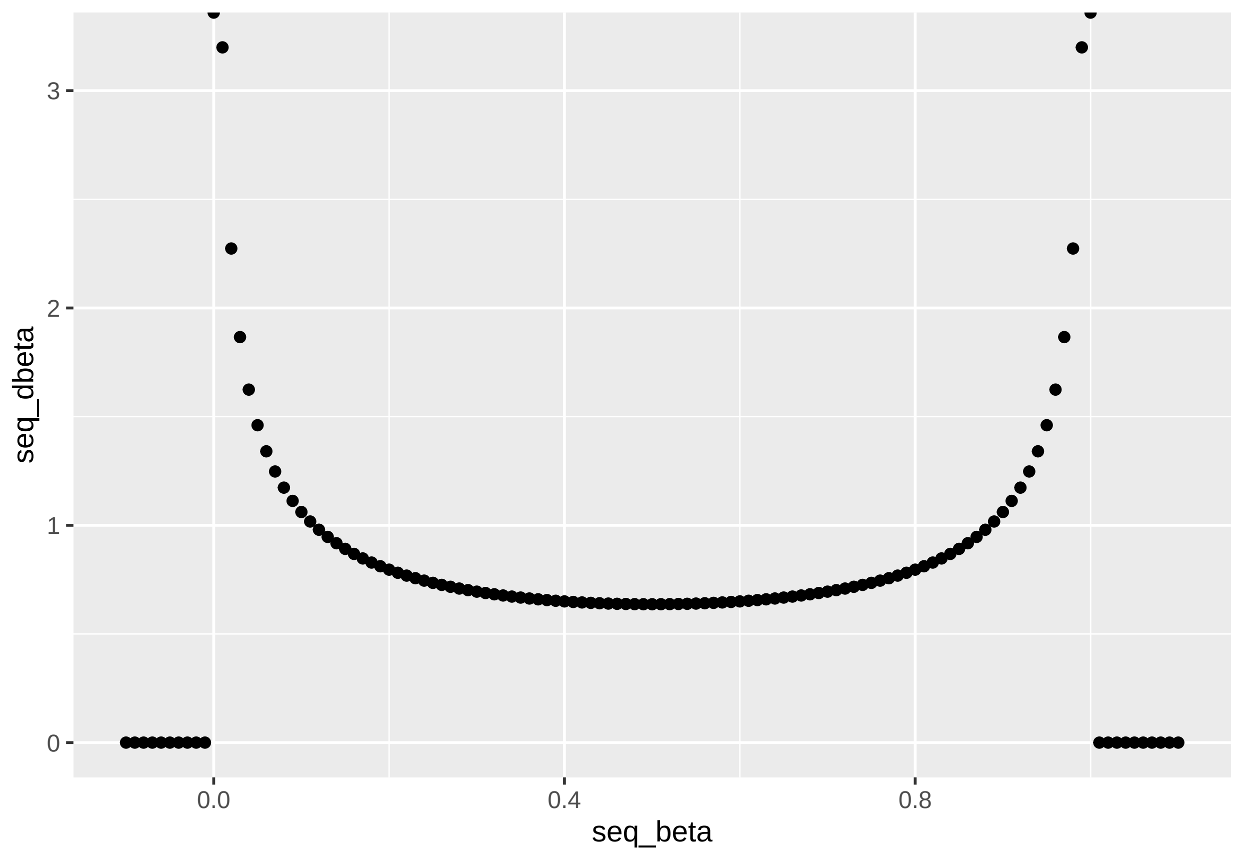


```
# Beta(.5,.5)

seq_beta = seq(-.1, 1.1, .01)

seq_dbeta = dbeta(seq_beta, .5, .5)

qplot(x = seq_beta, y = seq_dbeta)
```

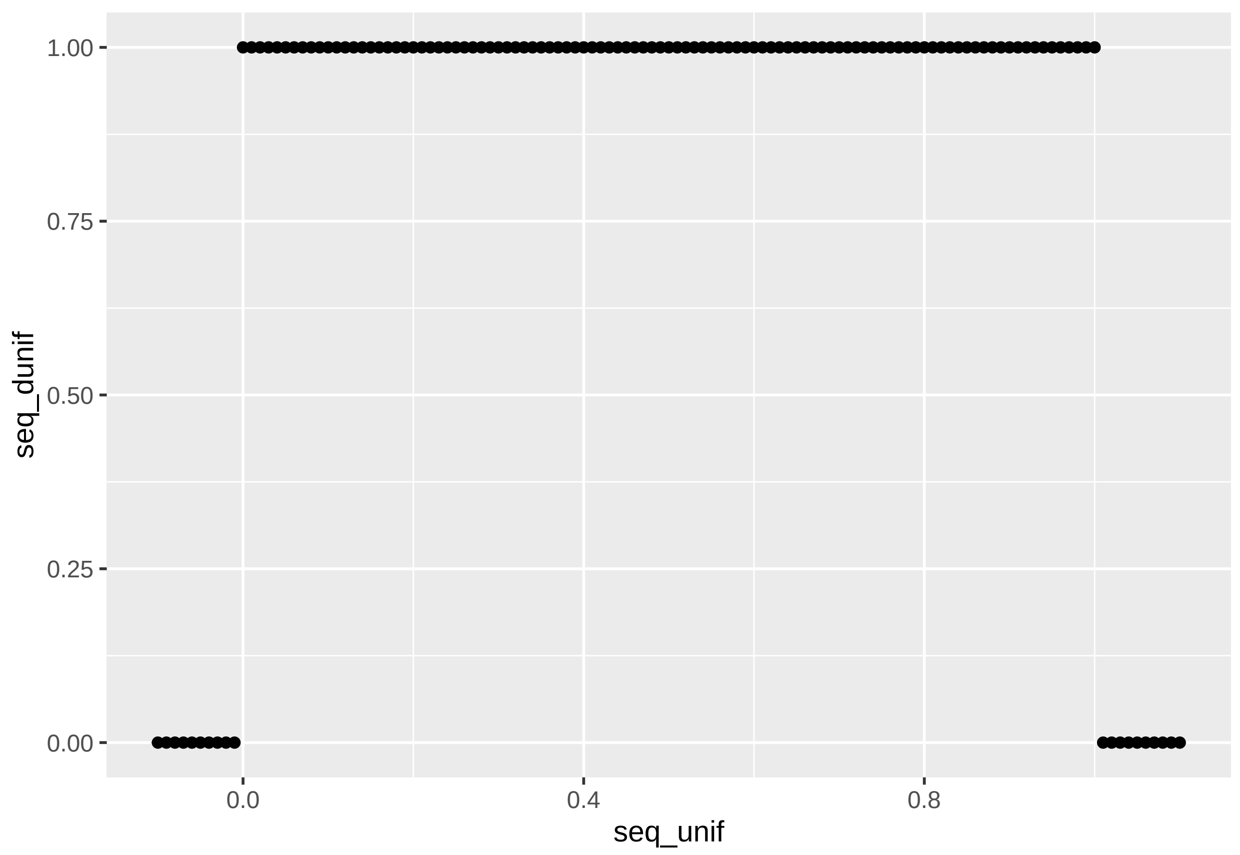


```
# Uniform (0,1)

seq_unif = seq(-.1, 1.1, .01)

seq_dunif = dunif(seq_unif, 0, 1)

qplot(x = seq_unif, y = seq_dunif)
```

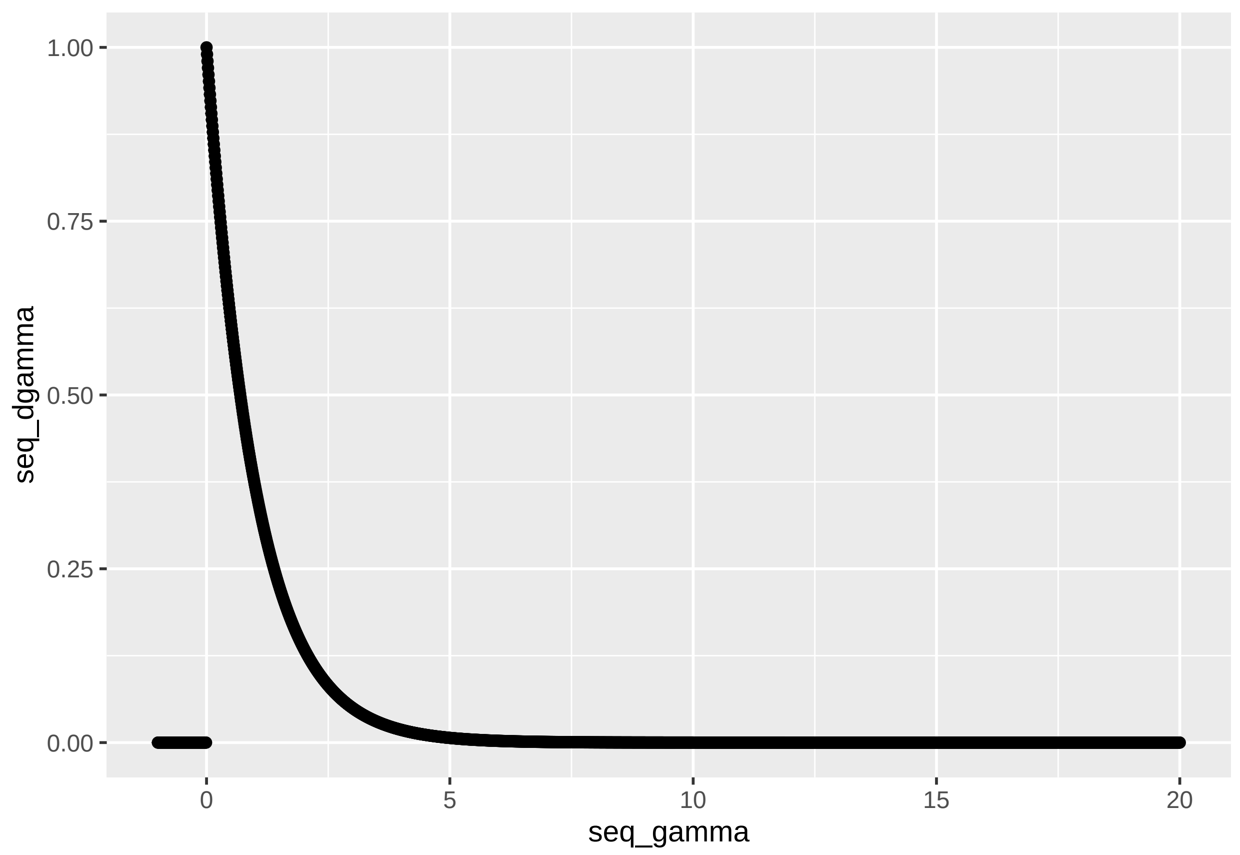


```
# Gamma(1,1)

seq_gamma = seq(-1, 20, .01)

seq_dgamma = dgamma(seq_gamma, 1, 1)

qplot(x = seq_gamma, y = seq_dgamma)
```

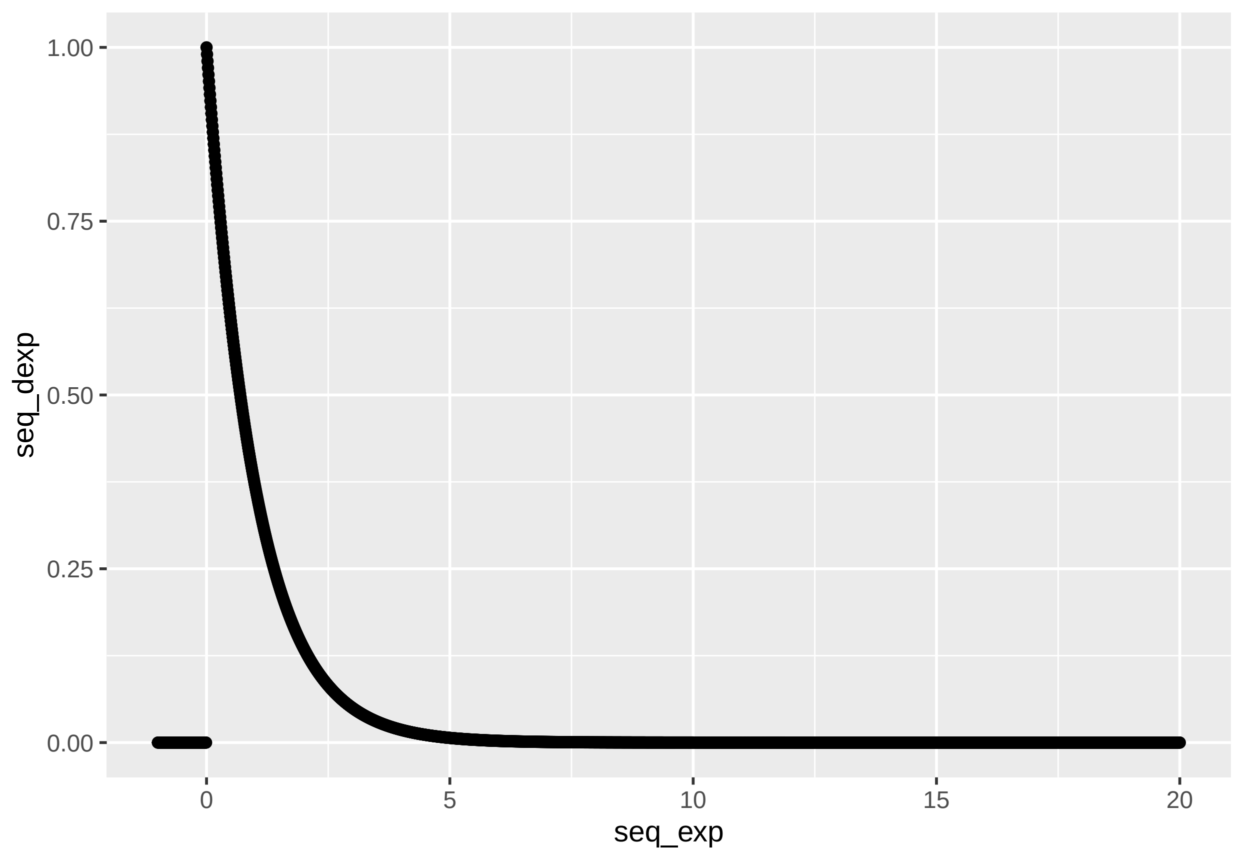


```
# Exponential(1)

seq_exp = seq(-1, 20, .01)

seq_dexp = dexp(seq_exp, 1)

qplot(x = seq_exp, y = seq_dexp)
```



1.7

Note that the mean of a pois(3) is 3, and the variance is also 3.

```
poisson_rv = rpois(100,3)

mean(poisson_rv)
```

```
## [1] 2.9
```

```
var(poisson_rv)
```

```
## [1] 4.090909
```

1.8

```
cel = BSgenome.Celegans.UCSC.ce2

dna_seq = cel$chrM
```