Code ▾

Statistical methods for de novo mutation analysis

This note describes some approaches for gene-level de novo mutation (DNM) analysis. We will use the dataset from [1]. To demonstrate these methods, only loss-of-function DNMs (dn.LoF) are used.

This is a draft, there would be errors/typos inside the note. We will update this note, and will also add current methods.

We will summarize two tests: the Poisson test, and a Bayesian test (a Mixture-model based approach).

Dataset

We will use the dataset from De Rubeis et al., (2015) [1]

[1]. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4402723/ (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4402723/)

- Read the data frame into `R`.

Hide

```
x <- read.table("test_Data_from_ASD.txt", header = TRUE, as.is = TRUE)
##https://www.nature.com/articles/nature13772#Sec9
Ntrio = 3871
```

- Print some lines of this dataset

Hide

```
head(x)
```

	Gene	mut.rate	dn.LoF
	<chr>	<dbl>	<int>
1	SCN2A	0.00007400	4
2	SYNGAP1	0.00006600	5
3	CHD8	0.00009200	3
4	ARID1B	0.00009000	4
5	ANK2	0.00014923	3
6	SUV420H1	0.00003500	3

6 rows

Statistical analysis

Poisson test

Let q, μ be the gene-level DNM count and the mutation rate of the tested gene, and let $Ntrio$ be the number of trios/families. Let X be a random variable denoting DNM counts. As described in [1], X follows a Poisson distribution with mean = $2 * Ntrio * \mu$, and the p-value is calculated as $P(X \geq q)$.

Usually, there are only some DNMs for a gene, we can write a simple function to calculate the p-value by using $P(X \geq q) = 1 - P(X < q)$. However, the function $ppois$ in $\mathbb{R}$ can be used to calculate the p-values.

[1] Ware JS, Samocha KE, Homsy J, Daly MJ. Interpreting de novo Variation in Human Disease Using denovolyzeR. Curr Protoc Hum Genet. 2015 Oct 6;87:7.25.1-7.25.15. doi: 10.1002/0471142905.hg0725s87. PMID: 26439716; PMCID: PMC4606471. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4606471/ (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4606471/).

Hide

```
x.poisson <- ppois(q = x$dn.LoF - 1, ##observed number - 1 (X - 1)
                    lambda = 2*Ntrio*x$mut.rate, ##expected
                    lower.tail = FALSE)
```

Take a look at the result.

Hide

```
x1 <- data.frame(x, x.poisson = x.poisson)
head(x1[x.poisson < 0.05, ])
```

	Gene <chr>	mut.rate <dbl>	dn.LoF <int>	x.poisson <dbl>
1	SCN2A	7.4e-05	4	0.0028513034
2	SYNGAP1	6.6e-05	5	0.0001901230
3	CHD8	9.2e-05	3	0.0356506317
4	ARID1B	9.0e-05	4	0.0056625306
6	SUV420H1	3.5e-05	3	0.0027099340
7	DYRK1A	3.1e-05	4	0.0001141829

6 rows

For this test, we can remove genes with DNM counts = 0 (p-value = 1).

Mixture model

Another approach is to use a mixture model. Based on how many hypotheses (n) you plan to test, you can build a mixture model of n distributions. There are some published methods for this approach, we will review a simple model and go through some of these methods.

Simple mixture model.

We can compare two hypotheses: the gene is a risk gene (H_1) and the gene is not a risk gene (H_0)[2]. Let M_1 and M_0 be the two models for the two hypotheses. We assume that X follows two Poisson distributions for the two models (*If we can find better distributions, we can use those distributions.*).

[2] *Statistical approaches only help to prioritize genes. To better understand the prioritized genes, other approaches (e.g., wet-lab/dry-lab approaches + results from independent datasets) should be used in downstream analyses.*

For example, we can model X as follows:

Hypothesis	Model	Distribution
H_1	M_1	$X \sim Poisson(2 * Ntrio * \mu * \gamma)$
H_0	M_0	$X \sim Poisson(2 * Ntrio * \mu)$
$-$	$-$	$-$

The parameter for M_0 is the same as the Poisson test above ($2 * Ntrio * \mu$). We assume another parameter for M_1: $2 * Ntrio * \mu * \gamma$. **The question here is how to find γ to compare two hypotheses**.

If you have any reliable information for γ, you can use that information. However, a simple way is to find γ from the tested dataset by using a likelihood function.

We will describe more for this part. However, you can write your own approach by reading mixture-model lectures.

`TADA` uses a similar approach, but add a prior for γ.

TADA method

We will calculate posterior probabilities (PPs) for genes from this method, and then compare with the Poisson test above.

Hide

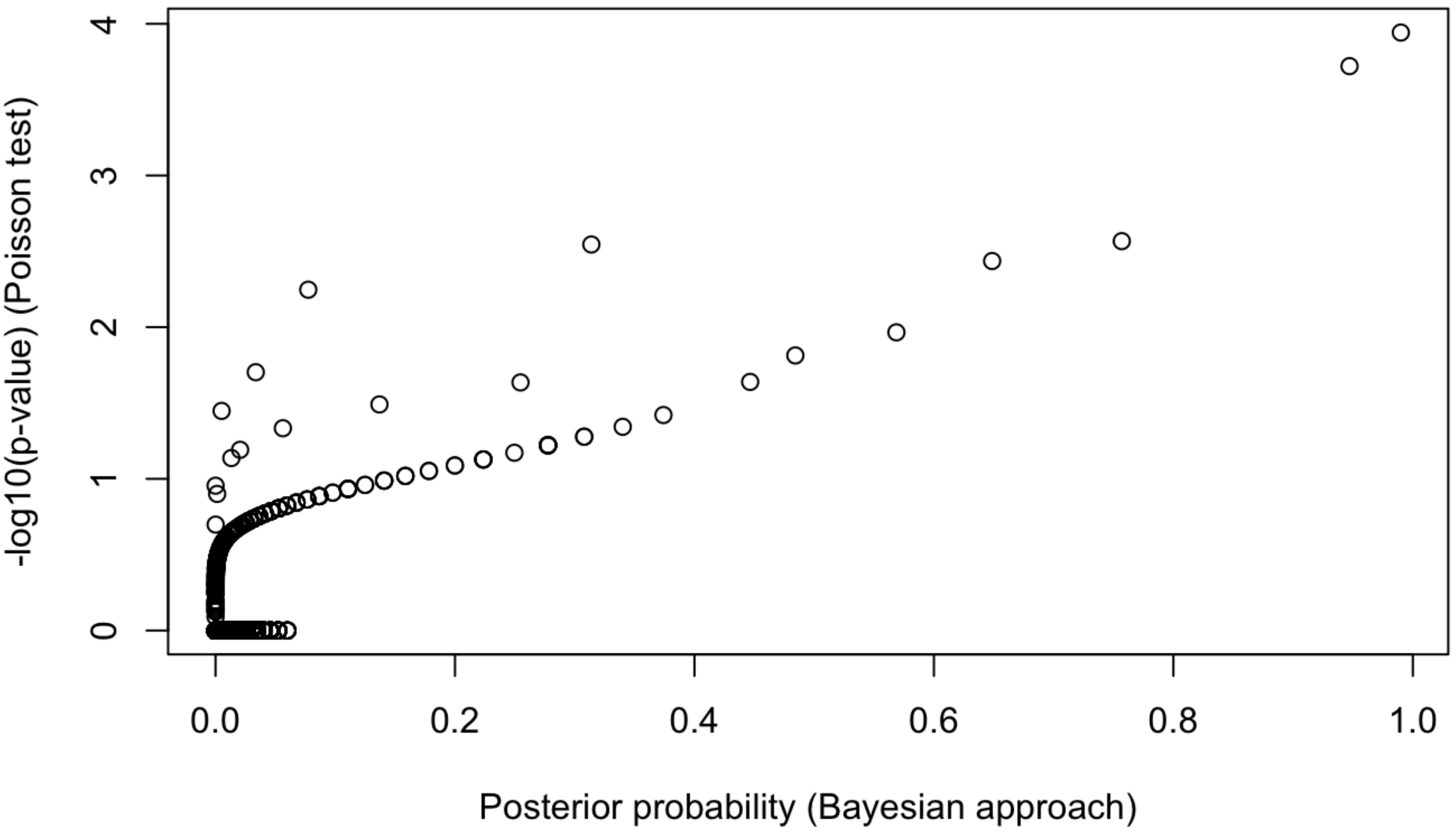
```
source("TADA.R")
gamma0 = 20
beta0 = 1
pi0 = 0.06
x.bf = bayes.factor.denovo(x$dn.LoF, N = Ntrio,
                           mu = x$mut.rate,
                           gamma.mean = gamma0, beta = beta0)
##Convert Bayes Factors to PPs

x.pp <- pi0*x.bf/(1 - pi0 + pi0*x.bf)
```

Compare results from a Poisson test and a Bayesian approach

Hide

```
plot(x.pp, -log(x.poisson, base = 10), xlab = 'Posterior probability (Bayesian approach)',
     ylab = '-log10(p-value) (Poisson test)')
```



Questions and answers

Do the Poisson test and a Bayesian approach generate different results?

A Bayesian method's results rely on its priors. Frequently, results are not much different if you use prior information from the tested data.

We would suggest that you take a look at a Poisson test described in [1] to better understand this test.

[1] Ware JS, Samocha KE, Homsy J, Daly MJ. Interpreting de novo Variation in Human Disease Using denovolyzeR. Curr Protoc Hum Genet. 2015 Oct 6;87:7.25.1-7.25.15. doi: 10.1002/0471142905.hg0725s87. PMID: 26439716; PMCID: PMC4606471 https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4606471/ (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4606471/).

For the TADA approach, can we use a different distribution?

Yes, of course. The Gamma distribution is a simple prior which can help us analytically calculate the marginal probability of the data. For example, in `TADA`, an analytic approach is used. However, you can use numerical integration to calculate Bayes Factors.

Example

In `TADA`, the marginal distribution is a negative binomial distribution. **If you are not familiar with *Calculus 1***, you can use **numerical integration** in `R` or other programming languages.

Note: Numerical integration is an approximation, and might not be the same as analytical integration.

In the `TADA` model, $X \sim Poissson(2 * Ntrio * \mu * \gamma)$ and $\gamma \sim Gamma(\bar{\gamma} * \beta, \beta)$, we can create a $Gamma$ function in `R`.

Below is an approach using numerical integration. We will also compare this approach with the negative binomial distribution based approach of `TADA`.

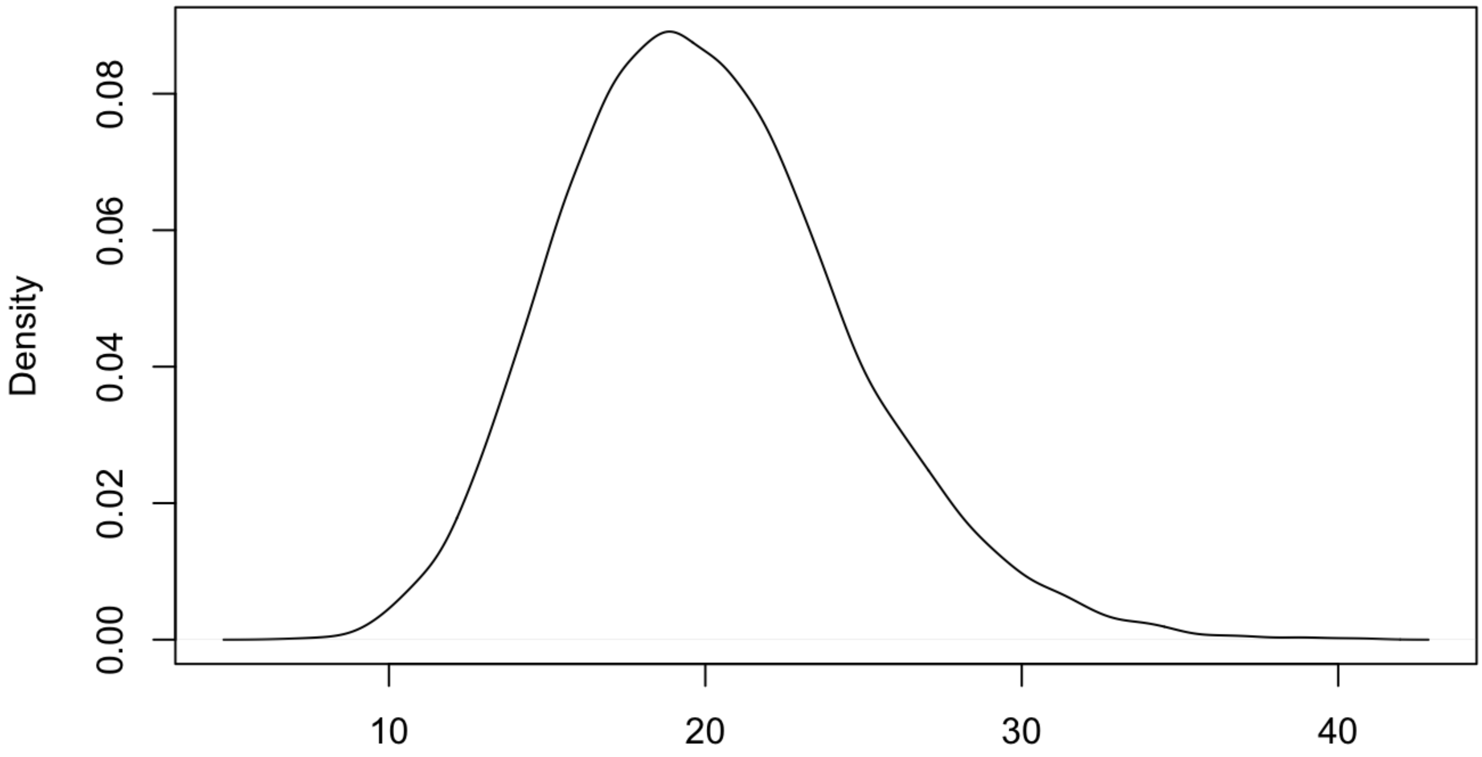
Hide

```
gamma0 = 20
beta0 = 1
f1 <- function(x.gamma){dgamma(x.gamma, shape = gamma0*beta0, rate = beta0)}
```

and have a look at its density distribution with parameters from the paper.

Hide

```
xGamma <- rgamma(10000, shape = gamma0*beta0, rate = beta0)
plot(density(xGamma), main = '', xlab = '')
```



Hide

```
gammaMin = min(xGamma)
gammaMax = max(xGamma)
```

Now, we can calculate Bayes Factors $\frac{P(X|H_1)}{P(X|H_0)}$ in which $P(X|H_1 = \int Poisson(X|2 * Ntrio * \mu * \gamma) Gamma(\gamma|\bar{\gamma} * \beta, \beta) d\gamma$ and $P(X|H_0) = Poissson(X|2 * Ntrio * \mu)$.

- We test for $X = 0$.

Hide

```
xdn = 0
(dpois(xdn, lambda = 2*Ntrio*10^-6*gamma0)*integrate(f1, gammaMin, gammaMax, subdivisions = 200L)$value)/dpois(xdn, lambda = 2*Ntrio*10^-6)
```

```
[1] 0.8630973
```

Hide

```
x.bf.test = bayes.factor.denovo(xdn, N = Ntrio,
                           mu = 10^-6,
                           gamma.mean = gamma0, beta = beta0)
x.bf.test
```

```
[1] 0.8637243
```

- We test for $X = 2$.

Hide

```
xdn = 2
(dpois(xdn, lambda = 2*Ntrio*10^-6*gamma0)*integrate(f1, gammaMin, gammaMax, subdivisions = 1000L)$value)/dpois(xdn, lambda = 2*Ntrio*10^-6)
```

```
[1] 345.2389
```

Hide

```
x.bf.test = bayes.factor.denovo(xdn, N = Ntrio,
                           mu = 10^-6,
                           gamma.mean = gamma0, beta = beta0)
x.bf.test
```

```
[1] 357.2117
```

- We test for all genes from the dataset in [1].

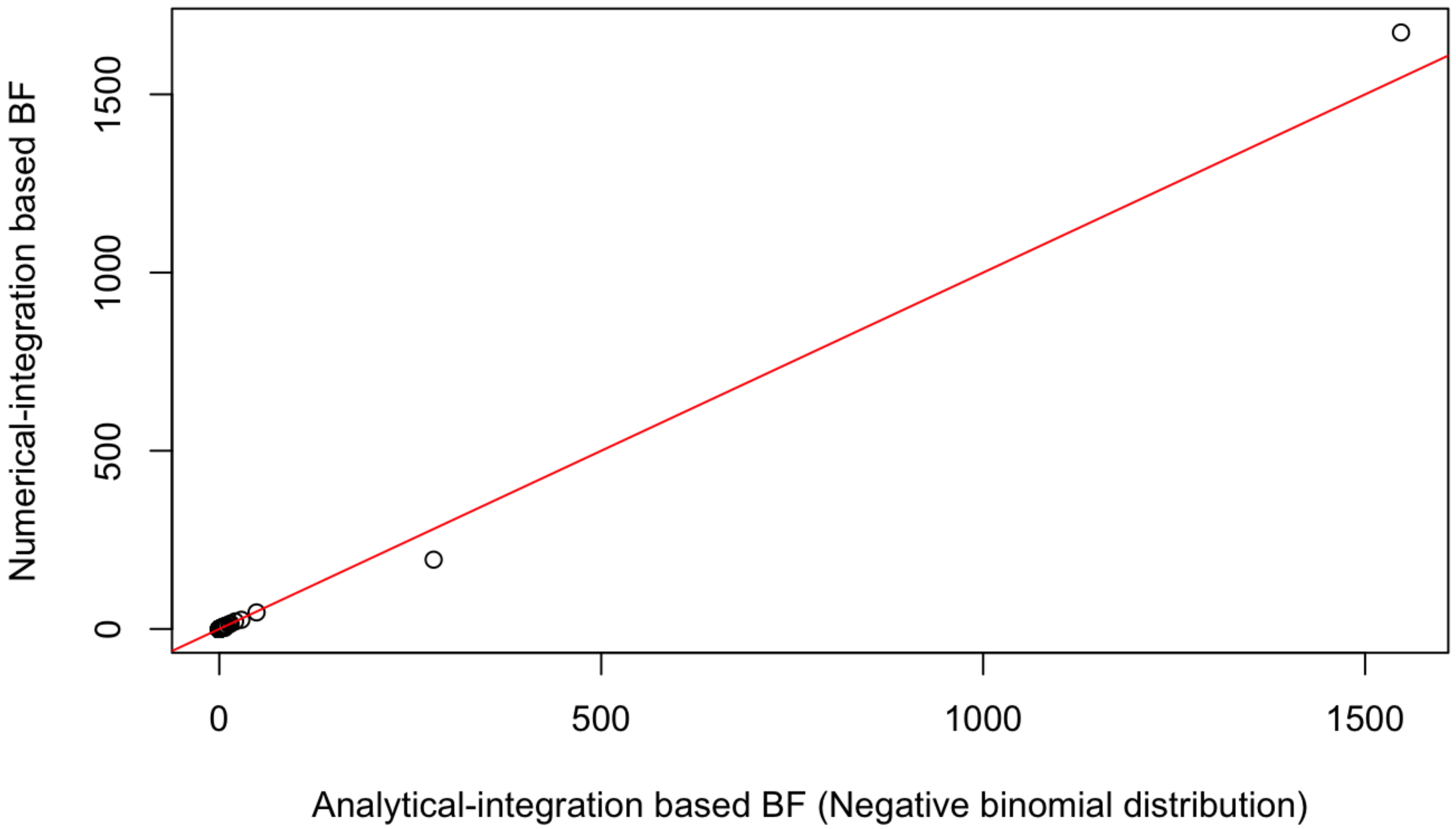
Hide

```
x.bf.numericalIn <- apply(data.frame(x$mut.rate, x$dn.LoF), 1, function(y){
  (dpois(y[2], lambda = 2*Ntrio*y[1]*gamma0)*integrate(f1, gammaMin, gammaMax, subdivisions = 200L)$value)/dpois(y[2], lambda = 2*Ntrio*y[1])
})
```

- Compare between the two methods.

Hide

```
plot(x.bf, x.bf.numericalIn, xlab = 'Analytical-integration based BF (Negative binomial distribution)', ylab = 'Numerical-integration based BF')
abline(a = 0, b = 1, col = 'red')
```



Hide

```
x[x.bf > 250, ]
```

	Gene <chr>	mut.rate <dbl>	dn.LoF <int>
2	SYNGAP1	6.6e-05	5
7	DYRK1A	3.1e-05	4

2 rows

Hide

```
NA
```

Hide

```
pT = 0.001
x[x.poisson < pT, ]
```

	Gene <chr>	mut.rate <dbl>	dn.LoF <int>
2	SYNGAP1	6.6e-05	5
7	DYRK1A	3.1e-05	4

2 rows

Hide

```
x.poisson[x.poisson < pT]
```

```
[1] 0.0001901230 0.0001141829
```