# Tutorial

This package can be used for estimating viral transmission bottleneck sizes using different methods.

## 1. Install package and dataset

Use the following commands to install the package "ViralBottleneck" and download the dataset in test_dataset folder

```
library(devtools)
install_github("BowenArchaman/ViralBottleneck", build_vignettes = TRUE)
library(ViralBottleneck)
```

### 1.1 Example data

Some datasets are associated with the R package and can be imported directly using:

```
ViralBottleneck::
```

A window would open with a list of all the objects available in the package including the functions and example datasets. The names of the example datasets start with `Example_`.

### 1.2. Test dataset

Files can also be imported from external sources by providing the path to the file. We provide some test dataset (`test_dataset`) in the Github repository (https://github.com/BowenArchaman/ViralBottleneck/). The folder contains two datasets: H1N1 dataset which is a realistic dataset [PROVIDE THE CITATION] and a simulated dataset which was created as part of the study. In the simulated dataset folder, `Publicated_simulated_dataset` contains all the datasets that can help the user reproduce the published results. `Example_dataset` is used for this tutorial. The `H1N1_dataset` will be used to illustrate an example on how the user can apply the package to their own dataset.

## 2. Create transmission object

[PROVIDE MORE INFO ON HOW TO USE download-directory.github.io] Download the `test_dataset/Simulated_dataset`
It could be download using "https://download-directory.github.io". The transmission object needs to be created before the bottleneck size estimation. To create the transmission object, the working directory requires two inputs: the transmission pairs table and a folder of sample files. This package will extract sample files according to the transmission pairs table in the user's input. The sample files for this tutorial are in the folder `Example_dataset`.

The transmission pairs are in a table which contains the names of donors in the first column and recipients in the second column.You can see the example via following code:

```
ViralBottleneck::Example_TansmissionPairs
```

To view the table:

| donor | recipient |
|---|---|
| donor_3000 | 50_0_All_r1 |
| donor_3000 | 50_3_All_r1 |
| donor_3000 | 50_6_All_r1 |
| donor_3000 | 50_9_All_r1 |
| donor_3000 | 50_12_All_r1 |

Note: Do not put the "-" in name of the sample.

After making sure the sample files all exist according to the transmission pairs, you can create the transmission object. Here, we directly import data of transmission pair from the package using `ViralBottleneck::`

```
Sim_trans = ViralBottleneck::Example_TansmissionPairs
Sim_ob = CreateTransmissionObject(Sim_trans)
```

The transmission object is an R object class which contains the transmission pair ID that is created by linking the donor and recipient sample names with a "-" character, and two "sample" R object classes: donor and recipient. The "sample" data structure stores the sample ID and the variant sites table containing the following information in columns: genome position, viral genome segment name, frequencies of the four bases (A, C, G, T), and whether the allele of the variant site are synonymous or non-synonymous mutations. You can see the example in the following code:

```
ViralBottleneck::Example_ob
```

### 2.1 Subset transmission object

The transmission object can be used as a list, thus enabling to subset the top three transmission pairs:

```
# Get first 3 transmission object

Sim_ob_subset = Sim_ob[1:2]
```

## 3. Summary transmission object

After creating the transmission object, the `Summary_ob` function will provide the information of the shared sites (sites that are sequenced both in donor and recipient). Example code:

```
Summary_Sim = Summary_ob(Sim_ob)
```

The result (which is also stored in Example data using `ViralBottleneck::Example_summaryOutput`) can be vieweed with the following code:

| Donors | Recipients | number.of.shared.sites |
|---|---|---|
| donor_3000 | 50_0_All_r1 | 13158 |
| donor_3000 | 50_3_All_r1 | 13158 |
| donor_3000 | 50_6_All_r1 | 13158 |
| donor_3000 | 50_9_All_r1 | 13158 |

| | | |
|---|---|---|
| donor_3000 | 50_12_All_r1 | 13158 |

## 4. Transmission bottleneck size estimation

We can now calculate the transmission bottleneck size using the transmission object. There are currently six methods provided in `ViralBottleneck`, including: `KL` method (Emmett et al., 2015), `Presence-Absence` method (Sacristán et al., 2011), `Binomial` method (Leonard et al., 2017), `Beta_binomial_Approximate` method (Leonard et al., 2017) and `Beta_binomial_Exact` method (Leonard et al., 2017) and `Wight-Fisher` method (Poon et al., 2016). In the future, more methods will be integrated into the package. (Note: if you want to access the original publication for each method, you could click the *Publication link* after each methods)

### 4.1 Output of `Bottleneck_size_Calculation` function

Calculation using the Beta-binomial method approximate version as an example:

```
BB_App_output =
  Bottleneck_size_Calculation(
            transmission_ob = Sim_ob,
            method ="Beta_binomial_Approximate",
            variant_calling = 0.03,
            error_filtering = 0
            Nbmin = 1,
            Nbmax = 200,
            donor_depth_threshold = 0,
            recipient_depth_threshold = 0
            )
```

The output can be presented as a table using the following code (it is also stored in Example data using `ViralBottleneck::Example_output`):

| donor | recipient | transmission_bottleneck_size | CI_low | CI_high |
|---|---|---|---|---|
| donor_3000 | 50_0_All_r1 | 70 | 64 | 70 |
| donor_3000 | 50_3_All_r1 | 45 | 30 | 64 |
| donor_3000 | 50_6_All_r1 | 28 | 20 | 39 |
| donor_3000 | 50_9_All_r1 | 34 | 23 | 47 |
| donor_3000 | 50_12_All_r1 | 47 | 31 | 67 |

### 4.2 Specify transmission pairs during estimation

This package provide a chance that if user need to specify some transmission pairs for estimation. Here we used example data to import the data. [THE FOLLOWING EXAMPLE IS UNCLEAR]

```
subset_transmission_pairs = ViralBottleneck::
```

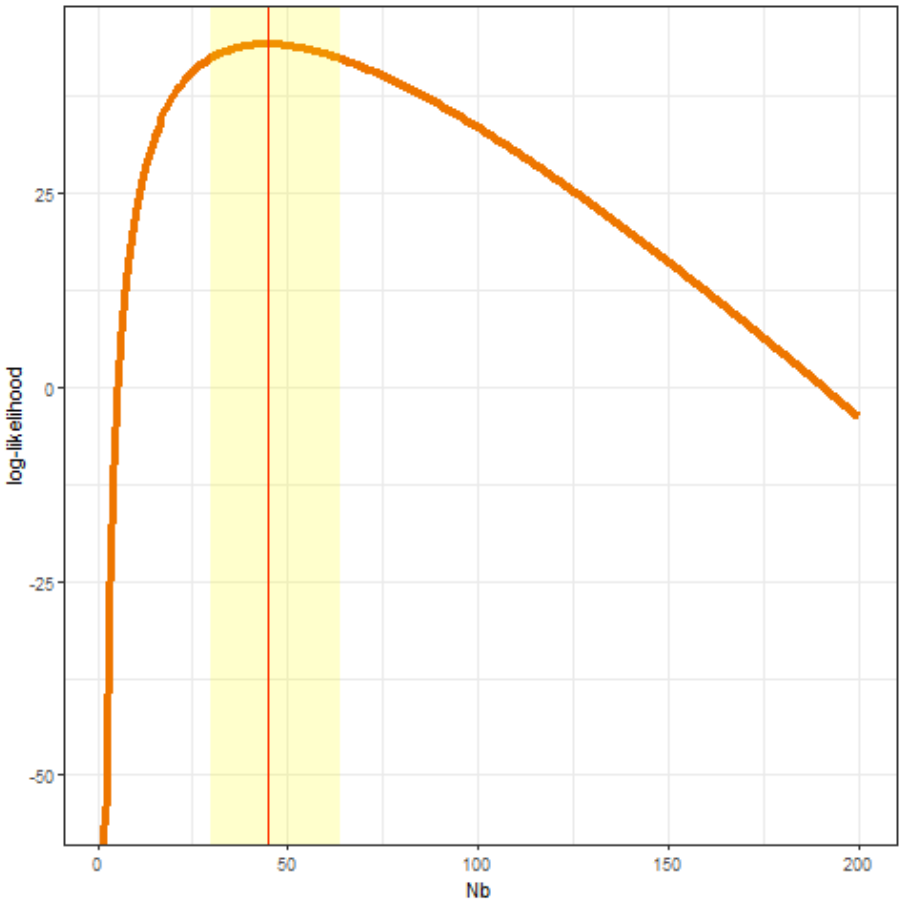
### 4.3 Plot

The `Bottleneck_size_Calculation` function can plot the likelihood curve for each transmission pairsand save the output as a csv file in the working directory. However, this argument only works for the methods using the maximum likelihoods estimation, including the `KL` method , the `Presence-Absence` method, the `Binomial` method, the `Beta_binomial_Approximate` method and the `Beta_binomial_Exact` method. Using `show_table` and `plot` options can help to save the output and obtain the plots of the likelihood curve for each transmission pairs.

The program would create individual folder for each transmission pair to store the plots. Example code for creating the plots:

```
BB_App_output_plot =
            Bottleneck_size_Calculation(
            transmission_ob = Sim_ob,
            method = "Beta_binomial_Approximate",
            variant_calling = 0.03,
            error_filtering = 0
            Nbmin = 1,
            Nbmax = 200,
            donor_depth_threshold = 0,
            recipient_depth_threshold = 0,
            show_table = FALSE,
            plot= TRUE
            )
```

The plot of the likelihood curve for one transmission pair (donor_3000-50_3_All_r1) is below:




### 4.4 Log file

`Bottleneck_size_Calculation` can create a log file containing number of variants used in calculation and the number of variants filtered before the calculation in the working directory.

Example code:

```
BB_App_output_log =
            Bottleneck_size_Calculation(
            transmission_ob = Sim_ob,
            method = "Beta_binomial_Approximate",
            variant_calling = 0.03,
            error_filtering = 0
            Nbmin = 1,
            Nbmax = 200,
            donor_depth_threshold = 0,
            recipient_depth_threshold = 0,
            log= TRUE
            )
```

Typical output of a `log` file:

| donor | recipient | donor_used | donor_unused | recipient_used | recipient_unused |
|---|---|---|---|---|---|
| donor_3000 | 50_0_All_r1 | 193 | 12965 | 193 | 12965 |
| donor_3000 | 50_3_All_r1 | 193 | 12965 | 193 | 12965 |
| donor_3000 | 50_6_All_r1 | 193 | 12965 | 193 | 12965 |
| donor_3000 | 50_9_All_r1 | 193 | 12965 | 193 | 12965 |
| donor_3000 | 50_12_All_r1 | 193 | 12965 | 193 | 12965 |

### 4.5 Methods comparison

Given that one major purpose of the package is to compare calculation of bottleneck sizes across methods on the same data set, it would be nice to illustrate this. For example, compare all methods (except Wright-Fisher, see below) on a single pair, Sim_ob[1]:

```
all_methods <-
  c("KL", "Presence-Absence", "Binomial", "Beta_binomial_Approximate", "Beta_binomial_Exact")

compare_methods <-
  t(sapply(all_methods, function(m){
    Bottleneck_size_Calculation(Sim_ob[1], method = m)
  }))

compare_methods
```

## 5.Example of using H1N1 dataset

An example using the realistic H1N1 dataset is in the folder `test_dataset`. After downloading the `H1N1_dataset` and setting up your working directory to the path to `H1N1_dataset`, the following code can be used. In this case, we import the information of the transmission pairs from the external csv file. It is important to set the correct working directory and to make sure that you have the transmission pairs file and related sample files in this directory. The code below can be applied to all the methods on one transmission pair:

```
library(ViralBottleneck)
# Set working directory and make sure you have
#               transmission pairs file and related host files in this directory.

setwd("/path/to/your/working/directory")
```

```
# Create transmission object.

transmission_pairs = read.csv("H1N1_transmission_pairs.csv", sep = ",")
ob_H1N1 = ViralBottleneck::CreateTransmissionObject(transmission_pairs)

# Applying all methods on one transmission pair.

all_methods <-
 c("KL", "Presence-Absence", "Binomial", "Beta_binomial_Approximate", "Beta_binomial_Exact")

compare_methods <-
  t(sapply(all_methods, function(m){
    Bottleneck_size_Calculation(ob_H1N1[1],
                                variant_calling = 0.03,
                                error_filtering = 0,
                                Nbmin = 1,Nbmax = 400,
                                donor_depth_threshold = 0,
                                recipient_depth_threshold = 0 ,
                                method = m)
  }))

# Save results as csv file.

write.csv(compare_methods,"compare_methods.csv")
```

A table to the results:

| method | donor | recipient | transmission_bottleneck_size | CI_low | CI_high |
|---|---|---|---|---|---|
| KL | 681_1_H1N1_donor | 681_1_H1N1_recipient | 21 | 14 | 30 |
| Presence-Absence | 681_1_H1N1_donor | 681_1_H1N1_recipient | 13 | 9 | 19 |
| Binomial | 681_1_H1N1_donor | 681_1_H1N1_recipient | 66 | 66 | 67 |
| Beta_binomial_Approximate | 681_1_H1N1_donor | 681_1_H1N1_recipient | 50 | 30 | 78 |
| Beta_binomial_Exact | 681_1_H1N1_donor | 681_1_H1N1_recipient | 49 | 30 | 78 |

## Reference:


Emmett, K. J., Lee, A., Khiabanian, H., & Rabadan, R. (2015) High-resolution genomic surveillance of 2014 Ebolavirus using shared subclonal variants. PLOS Currents Outbreaks 7, ecurrents.outbreaks.

Sacristán, S., Malpica, J. M., Fraile, A., & García-Arenal, F. (2003) Estimation of population bottlenecks during systemic movement of tobacco mosaic virus in tobacco plants. Journal of Virology77(18), 9906–9911.

Poon, L. L. M., Song, T., Rosenfeld, R., Lin, X., Rogers, M. B., Zhou, B., Sebra, R., Halpin, R.,Guan, Y., Twaddle, A., DePasse, J., Stockwell, T., Wentworth, D., Holmes, E., Greenbaum, B.,Peiris, J. S. M., Cowling, B. J., & Ghedin, E. (2016) Quantifying influenza virus diversity andtransmission in humans. Nature Genetics 48(2), 195–200.

Sobel Leonard, A., Weissman, D. B., Greenbaum, B., Ghedin, E., & Koelle, K. (2017) Transmission bottleneck size estimation from pathogen deep-sequencing data, with an application to human influenza A virus. Journal of Virology 91(14), e00171-17.