Beginner's guide to **rSalvador** *for Windows and Linux users*

This guide helps you get started with rSalvador. For more advanced usage, you should consult the accompanying technical manual or use the inbuilt help facility. To get inbuilt help for a function, e.g., `newton.LD`, type the R command `?newton.LD`.

1 Installing rSalvador on the Windows Platform

- Download and install R 3.0.1 or higher version on your computer.
- Make a new folder as your working folder. We recommend naming your working folder `c:/rsalvador`. Copy the rSalvador distribution file `rsalvador_1.1.zip` into that new folder.
- Invoke R. rSalvador requires the R package `hypergeo`, which you can install by the R command

```
install.packages('hypergeo')
```

- rSalvador also requires the R package `gdata`, which in turn requires the `perl` software. However, most Windows computers do not have the software `perl` pre-installed. We recommend `Strawberry Perl`, which is available on the Internet for free download. Choose the stable version to download and accept all defaults during installation. After installing `perl`, install the R package `gdata` with the R command:

```
install.packages('gdata')
```

- Users not interested in the Excel file import capability can skip the installation of both `gdata` and `perl`. In this case, you can ignore the harmless warning messages about the unavailability of `perl` and `gdata` that you may see each time you load rSalvador.
- Use the R command `setwd('c:/rsalvador')` to set the new folder as your working directory. If your working folder has a different name, replace `'c:/rsalvador'` with the actual name. You can verify your working directory with the R command `getwd()`. You can check whether the rSalvador distribution file `rsalvador_1.1.zip` is present in your working folder by executing `list.files()`. Finally, install rSalvador as follows:

```
install.packages('rsalvador_1.1.zip')
```

2 Installation on the Linux Platform

First, install R and the two R packages (`hypergeo`, `gdata`) as described in the above section. Second, copy the file `rsalvador_1.1.tar.gz` to a directory of your choice. Third, from the same directory, execute `R CMD INSTALL rsalvador_1.1.tar.gz`.

3 Staring rSalvador

Each time you invoke an R session, you need to load rSalvador with the R command `library(rsalvador)`.

4 Your First rSalvador Calculations

Here we use the Demerec data to demonstrate the basic capabilities of rSalvador. To view this data set, type `demerec.data`.

- to find the maximum likelihood estimate of m:

  ```
  newton.LD(demerec.data)
  ```

- to find the 95% confidence interval for m:

  ```
  confint.LD(demerec.data)
  ```

- To view the iteration details:

  ```
  newton.LD(demerec.data,show.iter=TRUE)
  ```

- to display the log-likelihood function:

  ```
  plot.likelihood.LD(demerec.data)
  ```

5 Importing and Exporting Data

There are three ways to transfer data into rSalvador.

- creating a sequence of numbers within R:

  ```
  y=c(0, 16, 20, 2, 2, 56, 3, 361, 9)
  ```

- importing data from a text file. This data file should have just one column of numbers. It can have one or more memo lines. (Use built-in help for details.)

  ```
  y=import.text.data('example1.txt')
  ```

- Typically, you may have saved your data in an Excel spreadsheet file, like the accompanying example file `example2.xlsx`. (See built-in help for details.) To import data from that file, type

  ```
  y=import.excel.data('example2.xlsx')
  ```

- Now you can repeat the calculations in the above section by replacing `demerec.data` with the new data variable `y`.

- Occasionally it may be desirable to save your data (say in `y`) into a plain text file for future use. This can be done by

  ```
  export.text.data('mytest.txt',y)
  ```

  To read this data file back into rSalvador for analysis:

  ```
  import.text.data('mytest.txt')
  ```

6 Adjusting for Plating efficiency

We use data from experiment #16 of Luria and Delbruck as an example. This experiment has a plating efficiency of 0.4. You can view the data by typing `luria.16.data` and learn more by typing `?luria.16.data`.

- to find the maximum likelihood estimate of m:

  ```
  newton.LD.plating(luria.16.data,e=0.4)
  ```

- to find the 95% confidence interval for m:

  ```
  confint.LD.plating(luria.16.data,e=0.4)
  ```

- To view the iteration details:

  ```
  confint.LD.plating(luria.16.data,e=0.4,show.iter=TRUE)
  ```

- to view the log-likelihood function graphically:

  ```
  plot.likelihood.LD.plating(luria.16.data,e=0.4)
  ```

7 Comparison of mutation rates

Two mutation rates can be compared by checking whether the two 84% confidence intervals overlap.

```
confint.LD.plating(unlist(crane.data[1]),0.1)/3.6e9
confint.LD.plating(unlist(crane.data[2]),0.1)/3.9e9
```

The two confidence intervals for the two mutation rates overlap, and hence the difference is not significant. If terminal cell population sizes are the same in the two experiments, `compare.LD` performs a likelihood ratio test. In the Newcombe experiments, the difference in N_t between Experiments F and H are small, so a likelihood ratio test is possible.

```
compare.LD(unlist(newcombe.data[6]),unlist(newcombe.data[8]))
```

The p-value is 0.9130.