Statistics for the Luria-Delbrück distribution: using the R functions

A. Hamon and B. Ycart

March 15, 2012

1 Foreword

1.1 You have never used R

1. Download and install R from `http://www.r-project.org/`;
2. download `LD.R` from `http://ljk.imag.fr/Membres/Bernard.Ycart/LD/` and save it to your disk.
3. launch a R terminal or interface and source the `LD.R` script, either through the interface or directly by a line of command. For Windows users:

   ```
   > source("C:\\path\\directory\\LD.R")
   ```

   For Linux users:

   ```
   > source("~/path/directory/LD.R")
   ```

The functions are ready to use. The prompt is `>`. Type a command after it, then return. Recall previous commands by $\boxed{\uparrow}$ and $\boxed{\downarrow}$. Start with a few simulations to see what random samples look like:

```
> rLD(10,1,1)
 [1] 12 23  1  0  6  0  2  0  5  2
> rLD(10,1,0.5)
 [1]    0    0    4    6 1633    2    2    1    1    2
> rLD(10,10,0.5)
 [1]  16109   577    60    59  1211 113795   423   332   108   123
> rLD(10,10,2)
 [1] 36 13  9 20 12 14 62 11 19 28
```

You can type several commands in a same line, separated by a semi-column.

```
> n<-1000; a<-20; r<-0.8; S<-rLD(n,a,r)
```

This affects the value 1000 to `n` (the sample size), 20 to `a` (the mean number of mutations α), and 0.8 to `r` (the fitness ρ). You now have a sample of size 1000 of the LD(20,0.8). You can display it by simply typing `S` after the prompt. You can get its lower and higher values, its quartiles, etc.

```
> c(min(S),max(S))
[1]     16 584202
> quantile(S,c(0.25,0.5,0.75))
  25%   50%   75%
 79.0 145.5 316.0
```

You can also count how many values are higher than 500 (500 is the default Winsorizing value in the Maximum Likelihood estimation procedure):

```
> length(which(S>500))
[1] 146
```

The Generating Function estimates of α and ρ are given by:

```
> GF.est(S)
[1] 19.9284288  0.8026976
```

You can get confidence intervals for α and ρ:

```
> GF.confint(S)
           2.5 %     97.5 %
alpha 18.7608463 21.0960112
rho    0.7739842  0.8314111
```

You can compute the Maximum Likelihood estimates, but for a large α, the result will be biased:

```
> ML.est(S)
           [,1]
[1,] 21.6293118
[2,]  0.8532976
```

You can change the Winsorization parameter: the higher `win`, the better the estimates, but the longer the execution.

```
> ML.est(S,win=1000)
           [,1]
[1,] 20.5886809
[2,]  0.8229966
```

You can also test hypotheses. The testing functions return p-values for one sided tests. For instance: is ρ significantly larger than 0.75?

```
> GF.test.rho(S,0.75)
[1] 0.0004834452
```

Is α significantly lower than 21?

```
> GF.test.alpha(S,21)
[1] 0.03083403
```

You can also try more challenging values.

```
> n<-10000; a<-200; r<-0.5; S<-rLD(n,a,r)
> c(min(S),max(S))
[1]         2615 587966401965
> quantile(S,c(0.25,0.5,0.75))
      25%       50%       75%
 36402.75 106906.50 472755.75
> GF.est(S)
[1] 200.7923817   0.4998954
```

Other estimators are given for sake of comparison. You can compare different methods on simulated samples by `compare.est`. The function retuns boxplots and mean squared errors: the lower the value, the better the estimator.

```
> compare.est(2,0.8,100,100)
[1] 0.2416981 0.2935468 0.4946447 0.4613086 0.6068226 1.2763606
```

We have included a few sets of published data. Open the `LD.R` script in an editor to see the references and data.

```
> GF.est(LD43a)
[1] 7.056536 1.083392
> GF.est(LD43b)
[1] 0.6932664 0.4944009
> GF.est(B94)
[1] 0.7086430 0.8240959
> GF.confint(B94)
          2.5 %    97.5 %
alpha 0.6501004 0.7671857
rho   0.7382408 0.9099511
> GF.test.rho(B94,1)
[1] 2.963898e-05
```

You can declare your own data using the same commands as in the script (parenthesize if more than one line).

```
> mydata <- c(0,1,0,0,2,34,0,5,0,0,89,3,0,4)
> mydata<-c(rep(0,49),rep(1,34),rep(2,8),rep(8,2),45,76,89)
```

If your data are in an Excel file, save it as a `.csv` file, then import it into R.

```
> MD <- scan("C:\\path\\directory\\mydata.csv")
```

The simplest may be to type your data in a script file like `LD.R`, and source it.

1.2 You are an experienced R user

This is only a script, and not a R package. The entries have not been protected and some functions might fail on extreme entries. We have tried to respect the spirit and scope of R but we also have focused on clarity and readability of the codes. There is certainly room for improvement in precision and computing time. You are welcome to read and modify the code for your own usage, and get back to us for possible improvement. The main parameters of the Generating Function estimators are entered in the `tuning()` function. They have been adjusted in order to minimize the variance and optimize the range of applicability. With these values, Generating Function estimates can be computed for samples of size n of the LD(α, ρ), with:

- $n \leqslant 10^6$,
- $\alpha \leqslant 200$,
- $\rho \geqslant 0.5$,

We believe these values cover any potential application.

2 Functions

The functions in the `LD.R` script match the theoretical results exposed in the paper. Each code comes with a short description of the function, and comments explaining the meaning of each command. Some functions were needed for programming structure, but they are not called independently; they are not documented here. We only describe functions of current usage.

2.1 Distributions

The functions treating the Yule and Luria-Delbrück distributions have a similar syntax as for other distributions in R.

- `rY(n,rho)` returns a sample of size `n` of the Y(ρ);
- `rLD(n,alpha,rho)` returns a sample of size `n` of the LD(α, ρ);
- `dY(m,rho)` returns in a vector the probabilities p_k of the Y(ρ) for k from 0 to `m`;
- `dLD(m,alpha,rho)` returns in a vector the probabilities q_k of the LD(α, ρ) for k from 0 to `m`;
- `pY(m,rho)` returns the probability to be lesser or equal to `m` for the Y(ρ);
- `pLD(m,alpha,rho)` returns the probability to be lesser or equal to `m` for the LD(α, ρ);
- `qY(p,rho)` returns the quantile of order p, $0 < p < 1$ for the Y(ρ);
- `qLD(p,alpha,rho)` returns the quantile of order p, $0 < p < 1$ for the LD(α, ρ).

2.2 Generating Function estimates

The following functions can be used.

- `GF.est(S)` returns the GF point estimates of α and ρ from a sample `S` of the LD(α, ρ);
- `GF.confint(S,level)` returns the confidence intervals for α and ρ from a sample `S` of the LD(α, ρ) (default level: 0.95);
- `GF.test.alpha(S,a0)` returns the p-value for the one-sided test of $\alpha =$ `a0` from a sample `S` of the LD(α, ρ); The side tested is that of the estimate, so the returned p-value is lesser than 0.5.
- `GF.test.rho(S,r0)` returns the p-value for the one-sided test of $\rho =$ `r0` from a sample `S` of the LD(α, ρ); The side tested is that of the estimate, so the returned p-value is lesser than 0.5.

2.3 Maximum Likelihood estimates

The following functions can be used.

- `ML.est(S,win)` returns the ML point estimates of α and ρ from a sample `S` of the LD(α, ρ); The data are winsorized at `win` (default=500).
- `Fisher.info(alpha,rho,max.sum)` returns the Fisher information matrix for the LD(α, ρ). Infinite sums are truncated at `max.sum`.

2.4 Other estimates

For sake of comparison, we have included other estimators of α, though we do not recommend their use. They are documented in Foster (2006).

- `P0.est(S)` returns an estimates of α for a sample `S` of the LD(α, ρ), using the p_0-method
- `LC.est(S)` returns an estimates of α for a sample `S` of the LD(α, ρ), using the Lea-Coulson median method (assuming $\rho = 1$).
- `JM.est(S)` returns an estimates of α for a sample `S` of the LD(α, ρ), using Jones' median method (assuming $\rho = 1$).
- `KQ.est(S)` returns an estimates of α for a sample `S` of the LD(α, ρ), using Koch's quartiles method.
- `AC.est(S)` returns an estimates of α for a sample `S` of the LD(α, ρ), using the accumulation of clones method.

We have included a function `compare.est(alpha,rho,E,n)` that compares the GF estimator to the 5 estimation methods above. The function simulates `E` samples of size `n` (default value 100) of the LD(α, ρ). On each of the `E` samples, it computes estimates of α by the 6 methods. The 6 mean quadratic errors are returned. Boxplots of the estimates are represented, together with the target value in red.