A tutorial dbEmpLikeGOF R package

Jeffrey C. Miecznikowski*, Albert Vexler†, and Lori A. Shepherd◇

April 6, 2012

Statistical Genetics and Genomics Research Group
Department of Biostatistics, University at Buffalo
New York State Center of Excellence in Bioinformatics and Life Sciences
Roswell Park Cancer Institute

*`jcm38@buffalo.edu`, †`avexler@buffalo.edu`, ◇`las65@buffalo.edu`

Contents

1 Introduction

The dbEmplikeGOF package provides a function dbEmplikeGOF to be used for density based empirical likelihood (EL) goodness-of-fit tests based on sample entropy, as well as to perform the two sample EL ratio test for distribution equality. The function provides the test statistic and associated p-values. The p-value can be calculated by Monte-Carlo methods or estimated based on pre-calculated tables of selected sample sizes and alpha values. For details and algorithms:

```
Vexler A, Gurevich G, Empirical likelihood ratios applied to
goodness-of-fit tests based on sample entropy.  Computational
Statistics and Data Analysis 54(2010) 531-545.

Gurevich G, Vexler A, A two-sample empirical likelihood ratio
test based on sample entropy.  Statistics and Computing, 2011.
```

2 Examples

The following performs a density-based empirical likelihood based goodness-of-fit tests based on sample entropy and calculates the p-value based on Monte-Carlo methods. The examples examine three null hypothesis, 1) data follows a normal distribution with unknown mean and standard deviation, 2) data follows a uniform distribution on 0 to 1 and 3) data from two samples are from the same distribution. The example below tests the data (`normData`) against the normal distribution.

```
> library(dbEmpLikeGOF)
> normData = rnorm(25)
> dbEmpLikeGOF(x=normData, testcall="normal", pvl.Table=FALSE)

...Working on teststat
...Working on p-value
$teststat
[1] 5.936877

$pvalue
[1] 0.439

>
```

The p-value can be estimated based on precalulated tables rather than preforming Monte-Carlo methods. This is controlled by the argument pvl.Table. To estimate based on tables pvl.Table argument is TRUE, which is the default setting.

```
> dbEmpLikeGOF(x=normData, testcall="normal", pvl.Table=TRUE)

...Working on teststat
estimating pvalue based on table
$teststat
[1] 5.936877

$pvalue
[1] 0.4416973

>
```

Similar calculations can be made to test data against a uniform distribution on zero to one.

```
> unifData = runif(30)
> # calculates pvalue based on Monte-Carlo methods
> dbEmpLikeGOF(x=unifData, testcall="uniform", pvl.Table=FALSE)

...Working on teststat
...Working on p-value
$teststat
[1] 8.390682

$pvalue
[1] 0.072

> # estimates pvalue based on tables
> dbEmpLikeGOF(x=unifData, testcall="uniform", pvl.Table=TRUE)

...Working on teststat
estimating pvalue based on table
$teststat
[1] 8.390682

$pvalue
[1] 0.07787386

>
```

Notice the data in each of the above examples was designed to match the proposed distribution. Below is an example where the data does not follow the proposed distribution

```
> dbEmpLikeGOF(x=unifData, testcall="normal", pvl.Table=TRUE)

...Working on teststat
estimating pvalue based on table
```

```
$teststat
[1] 15.95158

$pvalue
[1] 0.001

>
```

It is also possible to test for distribution equality between two samples. When specifying an `x` and `y` samples, the `dbEmpLikeGOF` function will test for distribution equality between the two samples.

```
> dbEmpLikeGOF(x=unifData, y=normData, pvl.Table=TRUE)

...Working on teststat
estimating pvalue based on table
$teststat
[1] 29.46118

$pvalue
[1] 0.001

> normDataSet2 = rnorm(40)
> dbEmpLikeGOF(x=normDataSet2, y=normData, pvl.Table=TRUE)

...Working on teststat
estimating pvalue based on table
$teststat
[1] 15.05777

$pvalue
[1] 0.1890284

>
```

Notice the sample vectors do not have to be of equal length.