P-value Weighting package for MATLAB

Edgar Dobriban*

March 16, 2016

Contents

1 Introduction

This MATLAB package contains open source implementations of p-value weighting methods for multiple testing, including Spjøtvoll, exponential and Bayes weights (proposed in Dobriban et al., 2015).

- Version: 0.0.1
- Requirements: Tested on MATLAB R2014a and R2014b.
- Author: Edgar Dobriban
- License: `GPL-3`

In addition, this package contains the code to reproduce all simulation results from the paper Dobriban et al. (2015). These are contained in the `\Examples` folder. The data analyses from that paper can be reproduced with the code from `\Data Analysis`, see Section 6.

*Department of Statistics, Stanford University, `dobriban@stanford.edu`

2 Installation

Extract the archive in any folder, say to `<path>` . The main functions are in the `Code` directory, which needs to be on the Matlab path. For instance add the following line to your Matlab startup:

```
addpath('<path>/pvalue_weighting_matlab/Code')
```

Alternatively, add that line to scripts that call functions in this package. An example data analysis is in the `\Examples \Example00 - First Example\ example.m` file. This is described in Section 5.

This file is the main documentation for the package. To start, look at the example (Section 5) or at the methods implemented (Sections 3, 4).

3 P-value weighting methods

For each p-value weighting method, we assume we observe data $T_i \sim \mathcal{N}(\mu_i, 1)$ and test each null hypothesis $H_i : \mu_i \geq 0$ against $\mu_i < 0$. The p-value for testing H_i is $P_i = \Phi(T_i)$, where Φ is the normal cumulative distribution function. For a weight vector $w \in [0, \infty)^J$ and significance level $q \in [0, 1]$, the weighted Bonferroni procedure rejects H_i if $P_i \leq qw_i$. Usual Bonferroni corresponds to $w_i = 1$.

Each p-value weighting method assumes some additional independent information about μ_i, and returns a weight vector w. These can then be used for weighted Bonferroni or other multiple testing procedures.

3.1 Bayes

Bayes p-value weights can be computed using: `[w, q_star, q_thresh, c] = bayes_weights(eta, sigma, q)`. The inputs specify the prior distribution of the means μ_i of the test statistics as:

$$\mu_i \sim \mathcal{N}(\eta_i, \sigma_i^2), \ i \in \{1, \ldots, J\}$$

where:

- `eta`: a vector of length J, the estimated means of test statistics, derived from the prior data
- `sigma`: a strictly positive vector of length J, the estimated standard errors of test statistics, derived from the prior data
- `q`: The weights are optimal if each hypothesis is tested at level q. For instance, if we want to control the FWER globally at 0.05, then we should use $q = 0.05/J$.

The outputs are:

- `w`: the optimal weights. A non-negative vector of length J.
- `q_star`: the value q^* for which the weights are optimal. This may differ slightly from the original q if q is large.
- `q_thresh`: the largest value of q for which the weights can be computed exactly
- `c`: the normalizing constant produced by solving the optimization problem.

This method was proposed in Dobriban et al. (2015).

3.2 Spjøtvoll

Spjøtvoll p-value weights can be computed using: `[w, c] = spjotvoll_weights(mu, q)`. The inputs our best guess at μ_i from the prior data:

- `mu`: a vector of length J, the estimated means of test statistics, derived from the prior data
- `q`: The weights are optimal if each hypothesis is tested at level q. For instance, if we want to control the FWER globally at 0.05, then we should use $q = 0.05/J$.

The outputs are:

- `w`: the optimal weights. A non-negative vector of length J.
- `c`: the normalizing constant produced by solving the optimization problem.

This method was proposed by Wasserman and Roeder (2006); Roeder and Wasserman (2009) and independently by Rubin et al. (2006). It was called Spjøtvoll weights in Dobriban et al. (2015), in honor of (Spjøtvoll, 1972).

3.3 Exponential

Exponential weights with can be computed as `w = exp_weights(mu, beta, q)`. Here the inputs are

- `mu`: a vector of length J, the estimated means of test statistics, derived from the prior data
- `beta`: the tilt β, which defines the exponent of the weight. The weights are defined as: $w_i = \exp(\beta|\eta_i|)/c$, where $c = \sum_{i=1}^{J} \exp(\beta|\eta_i|)$.
- `q`: The weights are optimal if each hypothesis is tested at level q. For instance, if we want to control the FWER globally at 0.05, then we should use $q = 0.05/J$.

The outputs are:

- `w`: the exponential weights. A non-negative vector of length J.

Exponential weights are sensitive to large means. To guard against this sensitivity, we truncate the weights larger than $1/q$ and re-distribute their excess weight among the next largest weights.

This weighting scheme was proposed in (Roeder et al., 2006), who recommend $\beta = 2$ as a default.

4 Other Utilities

P-value weighting methods consist of two components: (1) computing the weights and (2) running a multiple testing procedure. The focus of this package is in (1), but for the user's convenience, we provide some implementations of (2) as well.

4.1 Bonferroni Multiple Testing

Bonferroni multiple testing is performed with `[h]=bonferroni(pvals,[fwer],[report])`. Here the inputs are

- `pvals`: the p-values; a vector of length J with values between 0 and 1,
- `fwer`: (optional) The family-wise error rate (FWER) that needs to be controlled. Default is `fwer = 0.05`. This is the probability of making at least one error.

- `report`: (optional) `'yes'` or `'no'`. Print to screen a report of the form `'Out of 100 tests, 10 are significant using a family-wise error rate of 0.05.` Default is no.

The outputs are:

- `h`: a binary vector of length J. The indicators of the significant tests: 1 for significant, 0 for non-significant.

For a vector of J p-values P_i, and a FWER of α, Bonferroni rejects the hypotheses $P_i \leq \alpha/J$. For a weight vector $w \in [0,\infty)^J$ and significance level $q \in [0,1]$, the weighted Bonferroni procedure rejects H_i if $P_i \leq qw_i$. Usual Bonferroni corresponds to $w_i = 1$. Therefore, to run weighted Bonferroni, one must call the function `bonferroni` with the weighted p-values $Q_i = P_i/w_i$. An example:

```
%assume we have a vector of p-values P, and weights w
fwer = 0.05;
P_w = P/w;
h=bonferroni(P_w,alpha);
```

5 An Example with Synthetic Data

We perform an experiment with synthetic data, showing how using prior data can improve power in the current study. The code is in the `\Examples \Example00 - First Example\ example.m` file. To run it, set your working directory to the folder where the script resides. We will walk through `example.m` step by step.

The first step is to get the data. You have two options: load the data, or go though the generating steps.

5.1 Option 1: Load the readily generated data

The first option is to load the readily generated data. Run the second cell in the matlab script `example.m`

```
load('./Data/example_data.mat', 'J','P_current','t1','t2');
```

5.2 Option 2: Generate data

The second option is to walk through the data generating process. We generate two sets of test statistics, the prior and the current data. A small fraction of the prior data holds some information about the current data. However, most prior data is noise. In our experience working with GWAS data, this is a reasonable model for association studies done on two independent samples and two distinct traits (such as cardiovascular disease and aging).

We do this by drawing from a mixture distribution. We generate a large number J of tests. For each test i we flip a coin X_i: If $X_i = 1$, then the prior is meaningful, else it is noise. If the prior is meaningful, we draw a random negative μ_i and both test statistics (T_i^1, T_i^2) are Gaussians centered at μ_i. Else we draw two independent normal test statistics (T_i^1, T_i^2). This ensures that this small fraction of the data is correlated. The code is:

```
rng(0); %set seed
J = 1e3;
mu = - 2*abs(randn(J,1));
frac_sig = 0.1;
X = binornd(1,frac_sig,J,1);
t1 = normrnd(X.*mu,1);
t2 = normrnd(X.*mu,1);
```

The data that this generates shows the desired patter, as can be seen on a scatterplot. Most pairs have no correlation, but there is a small fraction that does.

The p-values for the one-sided tests $\mu_i = 0$ vs $\mu_i < 0$ utilizing only the current data are $P_i = \Phi(T_i^2)$.

5.3 Visualize Data

We should now have the variables `t1,t2,P_current,J` in memory. Next we plot a scatter of the prior and current test statistics:

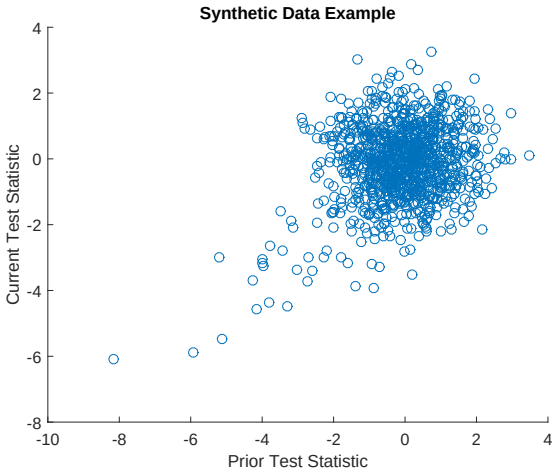

Figure 1: Scatter Plot of the Prior and Current Effects

There is only a weak global correlation between the two effects. However, there is a significant correlation in the tails.

5.4 Data Analysis

We want to test the hypotheses $\mu_i = 0$ against $\mu_i < 0$ for each i utilizing the current data T_i^2. The simplest way is by Bonferroni-corrected multiple testing. We choose an uncorrected significance level $\alpha = 0.05$ and call the `bonferroni(...)` function on the p-values.

```
alpha = 0.05; report = 'yes';
h_u=bonferroni(P_current,alpha,report);
```

The output should be:

```
Out of 1000 tests, 7 are significant using a  family-wise error rate  of 0.050000.
```

Bonferroni leads to 7 significant test statistics. Alternatively, one can do p-value weighting. For this, we use T^1 as prior information. We set the prior standard errors $\sigma = 1$ in this example. More detailed discussion on the choice of σ can be found in (Dobriban et al., 2015). As explained earlier, we set $q = 0.05/J$. Then we compute the weights. Finally, we run weighted Bonferroni on the weighted p-values $P_i' = P_i/w_i$. This is the code that accomplishes it:

```
q =alpha/J; %expected fraction of false rejections under 'null'
sigma = ones(J,1);
w = informed_weights(t1,sigma,q);
P_wr = P_current./w;
h_r=bonferroni(P_wr,alpha,report);
```

We should get the following output:

```
Out of 1000 tests, 12 are significant using a  family-wise error rate  of 0.050000.
```

Hence, in this example weighting increases the number of significant hits from 7 to 12.

5.5 Post-Analysis

One can get some insight into the procedure by examining which hypotheses were declared significant by the two methods. Typing

```
find(h_u==1)
find(h_r==1)
```

reveals that the significant hypotheses were:

```
ans =

    35
    84
   121
   563
   596
   734
   740


ans =

    35
    84
   121
   188
   221
   429
   563
   596
   645
   656
   734
   740
```

In this particular case weighting leads to a strict increase in power, selecting an addional 5 hypotheses. Taking 429 as an example we see that its P-value in the current data equals `P_current(429)=  1.0778e-04`, corresponding to a z-score `t2(429)=  -3.7001`, and this is not enough for it to be significant since the threshold is $0.05/1000 = 5e-5$. However, it gets assigned a large weight `w(429)=  8.9348`, because its prior effect is large, `t1(429)= -4.2573`; so it's selected after weighting.

Another insight can be gained from plotting the weights as a function of the prior mean. We see that the weights are non-monotonic as a function of the prior mean. Indeed they place a large mass on the middle means between (-6 and -2).

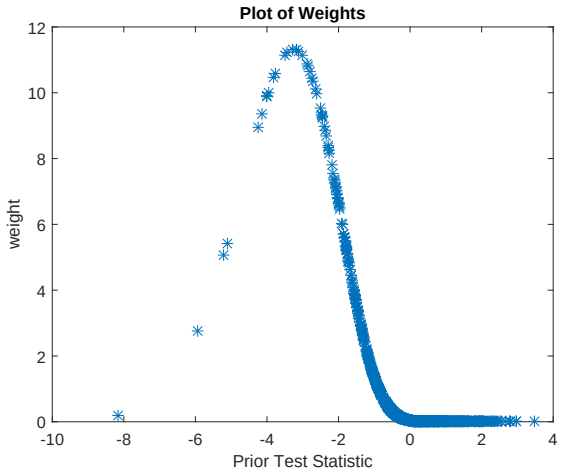

Figure 2: Plot of weights

6 Reproducing Data Analysis Results

6.1 Pipeline

To reproduce the data analysis results from the Dobriban et al. (2015) paper, we provide the pipeline used to generate those results. This pipeline can be used to efficiently and reproducibly perform standard data analyses with p-value weighting of GWAS data. It has the following features:

- High-level functions that perform a standard set of analyses: The user specifies the prior and current data sets, and the methods described in our paper are automatically performed.
- Customizable parameters: The user can can change many aspects of the analysis by specifying appropriate parameters. For instance, it is possible to change the parameters of the weighting schemes.
- Reproducibility: The results of analyses are recorded in a standard format and directory structure, which enable a reproducible analysis.
- Extensibility: It is possible to add new data sets, weighting schemes, and perform custom workflows. Some of this will require writing new code.

6.2 Reproducing results

Please follow these steps:

- You first need to download and process the original data files. Due to data access policies these could not be included in the package.
- To download data files, go to `Data/Raw/[Data Set Name]` and consult the description.txt file for the description web link for the appropriate data sets.
    1. Download and unpack each data set into its own `Data/Raw/[Data Set Name]` folder
    2. Run the MATLAB scripts in the `Data/Raw/[Data Set Name]/Code` folder to process the raw data into MATLAB files. They will be deposited in `Data/Raw/`.
    3. Repeat the steps above for each data set.
- To reproduce or change existing analyses go to the folder `Data Analysis/[AnalysisName]` and run or modify analysis.m

It is also possible to extend the pipeline:

- Add new analyses (for instance on newly specified pairs of GWAS data sets) by creating a new folder `Analysis/[AnalysisName]`
- Add new data sets by creating a new folder `Data/Raw/[DataName]/` where you can copy the data for a first processing. Using the existing data sets as a template, process the new data into a MATLAB format, which will then be saved in the folder `Data/Processed`.

Note: The change directory statement in analysis.m is configured to the author's local machine and should be modified accordingly.

References

Dobriban, E., Fortney, K., Kim, S. K., and Owen, A. B. (2015). Optimal multiple testing under a Gaussian prior on the effect sizes. *Biometrika*, 102(4):753–766.

Roeder, K., Bacanu, S.-A., Wasserman, L., and Devlin, B. (2006). Using linkage genome scans to improve power of association in genome scans. *The American Journal of Human Genetics*, 78(2):243–252.

Roeder, K. and Wasserman, L. (2009). Genome-wide significance levels and weighted hypothesis testing. *Statistical Science*, 24(4):398–413.

Rubin, D., Dudoit, S., and Van der Laan, M. (2006). A method to increase the power of multiple testing procedures through sample splitting. *Statistical Applications in Genetics and Molecular Biology*, 5(1):1–19.

Spjøtvoll, E. (1972). On the optimality of some multiple comparison procedures. *The Annals of Mathematical Statistics*, 43(2):398–411.

Wasserman, L. and Roeder, K. (2006). Weighted hypothesis testing. *arXiv preprint math/0604172*.