**TreeBH in Matlab**

Package created by Shaked Palgi, to implement the TreeBH method suggested in:*Bogomolov, M., Peterson, C. B., Benjamini, Y., & Sabatti, C. (2021). Hypotheses on a tree: new error rates and testing strategies. Biometrika, 108(3), 575-590.*

**Introduction**

Typically, a manuscript contains several experiments with various hypotheses. Every experiment can have multiple results (e.g. for a 2X2 ANOVA: 1-2 main effects of interest, 1 interaction, post-hoc analyses etc.). A possible way of describing this, is by using a tree. A tree is a mathematical graph, where every node has only one parent. In our case, every statistical test will be a node, which has a p-value. We can group a family of nodes (e.g. all outputs of interest from an ANOVA) to a higher level node. This parent node would not have its own p-value (what is the p-value of an experiment? A manuscript? A theory? Life?), but as you will see below, it is still important for the process.



Example tree. Each node is numbered and labeled. A family of hypotheses is a group of hypotheses which share a direct parent. In this example, the families are: Nodes 2-3, Nodes 4-5, Nodes 6-9 and Nodes 10-11.

Ideally, we would like to control the false discovery rate (FDR) at every level of our hypothesis tree. Luckily, there is a way to do that, using an algorithm called Tree-BH (Tree Benjamini-Hochberg), described in Bogomolov et al., 2021.

The algorithm works in two steps:

1. Bottom-up: calculate any missing p-values (experiment level, etc.) based on the children nodes and Simes’ formula. The formula is very simple:

Suppose we have a node with 3 children; we will accordingly have *3* p-values. According to the formula, these should be ordered from the smallest to the largest, and then multiplied based on the order: the 1st one (smallest) by 3 (*3*/1), the 2nd by 1.5 (*3*/2) and the 3rd (largest) by 1 (*3*/3). The Simes’ p-value of the node is the smallest of these values. In our example, the p-values of Exp. 1 and Exp. 2 were calculated by this method.

1. Top-down: correct for multiple comparisons **at the family level** using the Benjamini-Hochberg procedure, with a modified alpha level. The modified alpha depends on **the percentage higher level hypotheses for which the null was rejected**. If a node p-value is non-significant, we do not continue to test its children. The idea behind this is as following: on the one hand, it makes sense to compare families of similar hypotheses. On the other hand, our hypothesis selection process creates a selection bias. The solution is to keep the family-wise BH process, but modify the alpha based on the proportion of rejected hypotheses in higher levels. Intuitively, if many higher-level hypotheses were rejected, we have less of a selection problem, so we don’t need to correct by a lot. And vice versa, if the proportion of rejected higher-level hypotheses is lower, this means there was a stronger selection involved. In our example, for family F3 (post-hoc 1 & 2), the adjusted alpha is based on all the 3 higher-up levels: 0.5 (=>1 out of 2 from F2) × 0.5 (=>1 out of 2 from F1) × 1 (=>1 out of 1 from F0, which is rejected by definition (because if F0 would not be significant, then none of the children is significant, in which case there is no need to apply the correction to begin with)) = ¼×0.05 = 0.0125.



Demonstration of the tree-BH process on an example tree. Each node has a corrected p-value. The top 3 nodes (manuscript, Exp. 1 & Exp. 2) were calculated using Simes’ method. Nodes in red correspond to rejected hypotheses, and nodes in black to non-rejected ones. The dashed lines envelope the tested families: the purple family was not tested, as its parent was not rejected.

A few important points to notice:

* The hypothesis tree should contain **only effects of interest**. E.g. for a typical EEG 2X3X3 ANOVA (Factor of Interest X Region X Laterality), we are only interested in the effects which include our factor (Factor main effect, Factor X Region, Factor X Laterality, Factor X Region X Laterality). The question you should ask yourself is: does this result make the study publishable? All results for which the answer is yes should be included in this correction procedure.
* P-values of post-hoc analyses should be uncorrected at this stage: we will correct them during the FDR process.

**Step 1: Design your tree**

First, draw your tree on paper. Remember: TreeBH controls for FDR for each level of the tree, and for each family of hypotheses. Group your hypotheses in a meaningful way. Consult Liad about this stage, as it is crucial for conducting this procedure correctly.

After we have a tree, we can import it into Matlab. The way we build the tree in Matlab is by defining node parents. The nodes are numbered Top🡪Bottom, Left🡪Right. We skip the root of the tree, because it doesn’t have any parent node. For the example above, we have a vector of nodes (1-11) and a vector of the **parents of these nodes**:

nodes\_vec = [2 3 4 5 6 7 8 9 10 11];

parents\_vec = [1 1 2 2 3 3 3 3 5 5];

This is already enough to define our tree, but we want **node names** and **node p-values**. The node names variable should contain all the names of the nodes in our tree, ordered according to their number. In the example above, this is how the variable was defined:

node\_names = [{'Manuscript'},...

compose('Exp. %s',["1","2"]),...

{'Main effect','Interaction',...

'Main effect','Interaction 1','Interaction 2','Interaction 3'},...

compose('post-hoc %s',["1","2"])];

The node p-values is a vector of the p-values for all the nodes in our tree, ordered according to the node number (same order as for the node names). In the example here, we have two experiments and a post-hoc analysis, each with its own statistical results (defined here as: exp1, exp2 and exp1\_post\_hoc). We combine these numbers into one vector – the first 3 p-values are defined as NaN, because they correspond to the higher-level nodes (manuscript level, whole experiment level), which start without p-values of their own (these p-values will be calculated during the process).

exp1 = [0.1, 0.01];

exp2 = [0.04, 0.4, 0.4, 0.7];

exp1\_post\_hoc = [0.001, 0.1];

node\_p\_values = [nan(1,3), exp1, exp2, exp1\_post\_hoc];

Now, we can build our tree using the function ***createtree***:

G = createtree(parents\_vec,node\_names,node\_p\_values);

Sometimes, it is easier to build a few smaller trees and then combine them into a bigger tree. For that, we use the function ***mergetrees***. Mergetrees accepts two trees as inputs, and combines them as sisters. We can optionally include the name for the root.

G\_big\_tree = mergetrees(G\_small\_left,G\_small\_right,'New Root Node');

For example, in the above example, you could have created the tree for Exp.1 and Exp. 2 separately, and then combine them under the parent “manuscript”.

**Step 2: run TreeBH**

We run the tree BH process by using the function ***treeBH***:

[G\_output,G\_plot\_handle] = treeBH(G,plot\_tree,interactive\_plot,recalculate\_p,alpha\_level);

|  |  |
| --- | --- |
| **Inputs:** |  |
| G | Tree object |
| plot\_tree | Plot the tree – yes/no (1/0) |
| interactive\_plot | Plot an interactive plot of the treeBH process – yes/no (1/0).  Useful for debugging, as it shows the iterative process along the levels, with the adjusted p-values and significance thresholds. The code advances with each mouse click, showing the tree until the currentstep. Both this and plot\_tree can be set together to 1, and the result would be two plots: one during the run of the algorithm and one at the end. |
| recalculate\_p | Override existing p-values when calculating missing p-values bottom-up – yes/no (1/0). Default is 0.  Consider the case of a significant interaction, with post-hoc tests. Because we calculate the p-values of the tree bottom-up, we have a choice regarding which value to use for the interaction:   * The default (0) uses the original p-value of the interaction, which is the recommendation. That’s way we can interpret the p-value of the interaction like we are used to (A × B: a different effect of A at different levels of B) * If set to 1, we use the Simes’ formula to calculate the p-value of the interaction, based on the p-values of the post-hoc tests. The advantage here is increased power for the test of interaction, but it is not recommended since the new p-value no longer means exactly what we are used to. |
| alpha\_level | Unadjusted alpha level. Default is 0.05. |
| **Outputs:** |  |
| G\_output | G\_output.Nodes has 2 new properties:   * Corr\_p: corrected p-values * Alpha: corrected alpha level * Reject: reject H0 (yes=1) |
| G\_plot\_handle | If we plot the tree, this is the handle for the graph plot object. Useful to make our plots prettier. |

**Appendix 1: list of functions in the package**

* createtree
* mergetrees
* treeBH
* ucp\_treeBH (demo script)

**Appendix 2: a detailed example – Unconscious Pairs, by Tamara Bester-Arest & Shaked Palgi**

Data collected by Liad, Tzahi Kravitz, Shaked and Tamara, 2013-2021.

In this study, we showed our participants pairs of objects, which could be associatively related to each other (e.g. a camera and a tripod) or unrelated (e.g. chewing gum and fireworks). This task requires semantic integration of the two objects, and we tested it under conscious and unconscious conditions. Importantly, our study included 6 different experiments – 3 behavioral studies (using priming) and 3 EEG studies. 2 of them belonged to the conscious condition, and 4 belonged to the unconscious condition. Due to the different design of the priming and EEG studies, we used different statistical testing schemes – mixed linear models and ANOVAs.

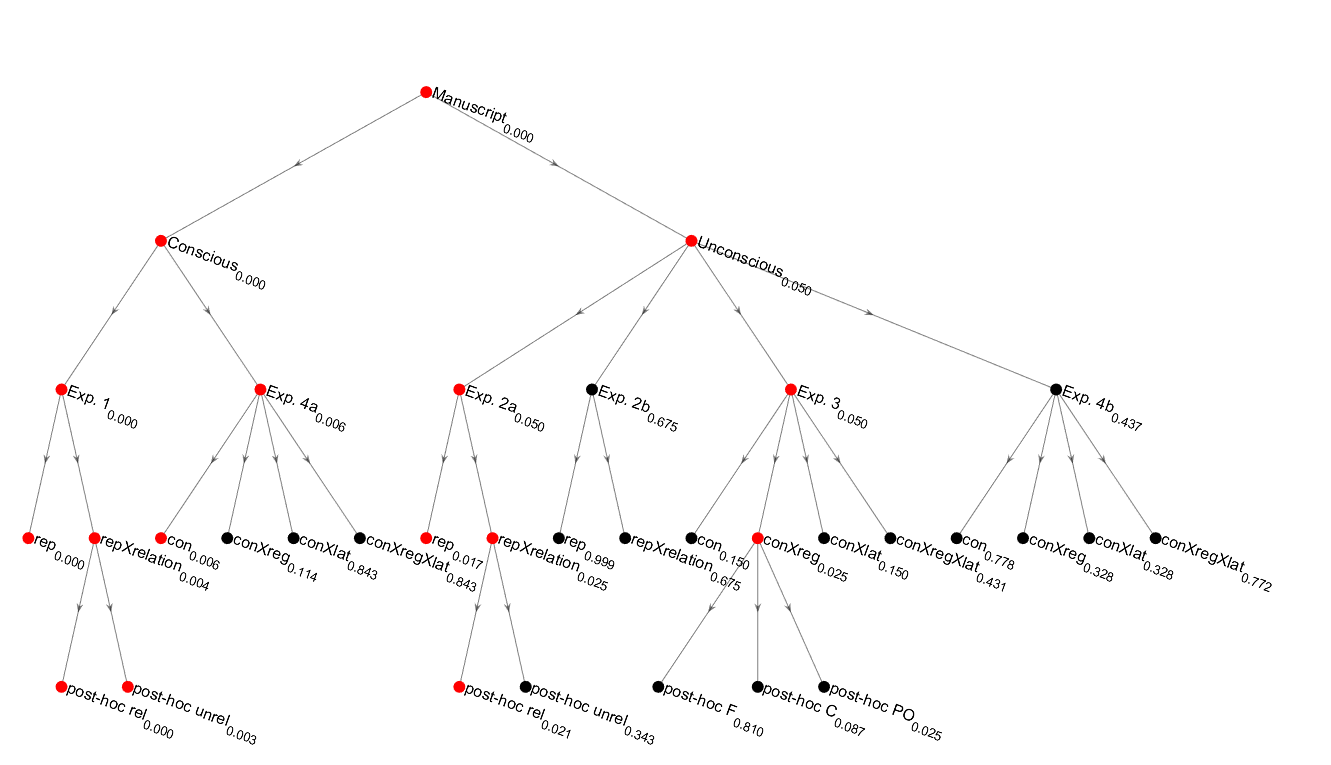


We realized that due to the very different nature of conscious and unconscious studies, and our different hypotheses for them, it makes more sense to first group the experiments by conscious/unconscious.

Each priming experiment had 1 main effect of interest + 1 interaction, with additional post-hoc tests which we did if the interaction was significant. Each EEG experiment had 1 main effect of interest + 3 interactions, with additional post-hoc tests if needed.

Overall, we had 18 p-values (level 4) and 7 post-hoc p-values (level 5). The p-values of the rest of the nodes were undefined.

After using the tree-BH algorithm, we rejected the null-hypothesis in 4 different experiments (1, 4a, 2a and 3).



UCP hypothesis tree. Red nodes are hypothesis where we rejected H0, and black nodes are hypotheses where we did not reject H0. Numbers correspond to corrected p-values. However, p-values whose parent node was not rejected, were not corrected by the tree-BH process, as they automatically do not pass significance. Code to replicate this plot is in **ucp\_treeBH.m**.