The *harmonicmeanp* package

Daniel J. Wilson

17 December 2018

1 Overview

The harmonic mean p-value (HMP) is a method for performing a combined test of the null hypothesis that no p-value is significant (Wilson, 2019). Unlike Fisher's (1934) method, it is robust to dependence between the p-values, making it much more broadly applicable. Like Bonferroni correction, the HMP controls the *strong-sense family-wise error rate* (ssFWER), but it is potentially much more powerful. It is also more powerful than the BH procedure (Benjamini and Hochberg, 1995) which controls both the *weak-sense family-wise error rate* (wsFWER) and the *false discovery rate* (FDR), in the sense that whenever the BH procedure finds one or more p-values significant, the HMP will find one or more p-values or *groups of* p-values significant.

Method	**Robust to dependence**	**Indicative power**[1]			**Controls**		
		Significance very rare	**Significance uncommon**	**Significance common**	**FDR**	**wsFWER**	**ssFWER**
Fisher	×	•	••••	•••••	✓	✓	✓
HMP	✓	••○	•••	••••	✓	✓	✓
BH	✓	••○	••○	••○	✓	✓	×
Bonferroni	✓	••○	••	••	✓	✓	✓

There are two components to the HMP method:

- The harmonic mean p-value itself, which acts as both a test statistic and, when small (e.g. below 0.05), an approximate combined p-value. The harmonic mean of the p-values indexed by subset $\mathcal{R}$ is denoted $\overset{\circ}{p}_{\mathcal{R}}$. The subscript is dropped when $\mathcal{R}$ includes all the p-values.
- A distribution for the HMP which produces a combined p-value that is exact asymptotically (i.e. as more and more p-values are combined). This asymptotically exact p-value is denoted $p_{\overset{\circ}{p}_{\mathcal{R}}}$ where the subset $\mathcal{R}$ defines the p-values it combines. The subscript $\mathcal{R}$ is dropped when $\mathcal{R}$ includes all the p-values.

The HMP equals $\overset{\circ}{p}_{\mathcal{R}} = \left(\sum_{i\in\mathcal{R}} w_i\right) / \left(\sum_{i\in\mathcal{R}} w_i/p_i\right)$, where $p_i, i = 1 \ldots L$ are the individual p-values and $w_i, i = 1 \ldots L$ are weights, which must sum to one, i.e. $\sum_{i=1}^{L} w_i = 1$. The HMP is robust to the choice of weights, so it is reasonable to start with equal weights ($w_i = 1/L$). Optimal weights are considered in more detail later.

The method is used as follows:

- The "headline" HMP is deemed significant when $p_{\overset{\circ}{p}} \leq \alpha$, or (approximately equivalent when $\overset{\circ}{p}$ is small), when $\overset{\circ}{p} \leq \alpha$, where α is the pre-specified ssFWER. Here significant means that we reject the null hypothesis that none of the p-values are significant.
- If the headline HMP is not significant, neither is the HMP for any subset. If the headline HMP is significant, subsets may also be significant. The significance thresholds are all pre-determined so the number of subsets that are tested does not affect them.
- The HMP for a subset of p-values is deemed significant when $p_{\overset{\circ}{p}_{\mathcal{R}}} \leq \alpha\, w_{\mathcal{R}}$, or (approximately equivalent when $\overset{\circ}{p}_{\mathcal{R}}$ is small), when $\overset{\circ}{p}_{\mathcal{R}} \leq \alpha\, w_{\mathcal{R}}$, where $w_{\mathcal{R}} = \sum_{i\in\mathcal{R}} w_i$ is the sum of the weights for subset $\mathcal{R}$. Here significant means that we reject the null hypothesis that none of the p-values in subset $\mathcal{R}$ are significant.

[1] Wilson (2019) SI Appendix, Fig. S8. BH power has been equated with Simes' (1986), on which BH is based.

2 Quick-start guide

Example 1. Sliding Window Analysis

Once you have installed the package, load it in the usual way:

```
library(harmonicmeanp)

## Loading required package:  FMStable
```

Download the 312457 p-values from chromosome 12 of the genome-wide association study (GWAS) for neuroticism (Okbay *et al.*, 2016). This file is an excerpt of http://ssgac.org/documents/Neuroticism_Full.txt.gz. For usage conditions see http://ssgac.org/documents/ReadMe_genetic_variants_associated_with_swb.txt. It took me a few seconds to download the data excerpt. The 8 megabyte file contains rs identifiers and SNP positions as per human genome build GRCh37/hg19 as well as the p-values.

```
system.time((gwas = read.delim("http://www.danielwilson.me.uk/files/Neuroticism_ch12.txt",
  header=TRUE,as.is=TRUE)))

##    user  system elapsed
##   1.635   0.105   3.816

head(gwas)

##            rs    pos      p
## 1   rs7959779 149478 0.3034
## 2   rs4980821 149884 0.5905
## 3 rs192950336 150256 0.1125
## 4  rs61907205 151213 0.4896
## 5   rs2368809 151236 0.7066
## 6   rs4018398 151469 0.9420
```

The harmonic mean p-value (HMP) is a statistic with which one can perform a combined test of the null hypothesis that *none* of the p-values is significant even when the p-values are dependent. In GWAS, p-values will often be dependent because of genetic linkage. The HMP can be used to test the null hypothesis that no SNPs on chromosome 12 are significant. Let's do it manually by first calculating the HMP, assuming equal weights. Note that a total of $L = 6524432$ tests were performed genome-wide, so this number must be used to determine the weights if we are to control the genome-wide ssFWER, even though we are only analysing the 312457 SNPs on chromosome 12 in this example.

```
gwas$w = 1/6524432
R = 1:nrow(gwas)
(HMP.R = sum(gwas$w[R])/sum(gwas$w[R]/gwas$p[R]))

## [1] 0.0008734522
```

One of the remarkable properties of the HMP is that for small values (e.g. below 0.05), the HMP can be directly interpreted as a p-value (Wilson, 2019). Since the HMP equals $\overset{\circ}{p}_{\mathcal{R}} = 0.0008734522$ it can be directly interpreted, suggesting it is strongly significant (before multiple testing correction). However, the recommended approach is to calculate an asymptotically exact p-value based on the HMP statistic.

```
# Use p.hmp instead to compute the HMP test statistic and
# calculate its asymptotically exact p-value in one step
pharmonicmeanp(HMP.R, L=length(R), lower.tail=TRUE)

## [1] 0.0008887255
```

As you can see, the asymptotically exact p-value of $p_{\overset{\circ}{p}_{\mathcal{R}}} = 0.0008887255$ is very close to the HMP of $\overset{\circ}{p}_{\mathcal{R}} = 0.0008734522$ because the HMP is much smaller than one. Note however that direct interpretation of the HMP

is anti-conservative compared to the asymptotically exact test, and this may be important when the HMP is not small (e.g. when it is only marginally below 0.05). The asymptotically exact p-value can be computed in one step, and this is the recommended usage:

```
R = 1:nrow(gwas)
p.hmp(gwas$p[R],gwas$w[R])

##        p.hmp
## 0.0008887255
```

The threshold against which to evaluate the significance of the combined test is determined by the sum of the weights for the p-values being combined. Suppose for example $\alpha = 0.05$, then the Bonferroni-adjusted threshold against which to compare the asymptotically exact HMP is

```
alpha = 0.05
w.R = sum(gwas$w[R])
alpha*w.R

## [1] 0.002394515
```

Therefore we can reject the null hypothesis of no association on chromosome 12 at level $\alpha = 0.05$ because $p^{\circ}_{p_{\mathcal{R}}} < \alpha\, w_{\mathcal{R}}$.

The combined p-value for chromosome 12 is useful because **if the combined p-value is not significant, neither is any constituent p-value**, after multiple testing correction, as always. Conversely, if the combined p-value is significant, there may be one or more subsets of constituent p-values that are also significant. These subsets can be hunted down because another useful property of the HMP is that the significance thresholds of these further tests are the same no matter how many combinations of subsets of the constituent p-values are tested. Specifically, for any subset $\mathcal{R}$ of the L p-values, the HMP (or, as recommended, the asymptotically exact HMP) is compared against a Bonferroni threshold $\alpha\, w_{\mathcal{R}}$ where $w_{\mathcal{R}} = \sum_{i \in \mathcal{R}} w_i$ and the w_is are the weights of the individual p-values, constrained to sum to one. Assuming equal weights, $w_i = 1/L$, meaning that $w_{\mathcal{R}} = |\mathcal{R}|\,/L$ equals the fraction of all tests being combined.

For example, separately test the p-values occurring at even and odd positions on chromosome 12:

```
R = which(gwas$pos%%2==0)
p.hmp(gwas$p[R],gwas$w[R])

##        p.hmp
## 0.0009159103

w.R = sum(gwas$w[R])
alpha*w.R

## [1] 0.001200587

R = which(gwas$pos%%2==1)
p.hmp(gwas$p[R],gwas$w[R])

##        p.hmp
## 0.0008619354

w.R = sum(gwas$w[R])
alpha*w.R

## [1] 0.001193928
```

Unsurprisingly, in view of genetic linkage, the two tests are both significant: for even positions, the combined p-value was $p^{\circ}_{p_{\mathcal{R}}} = 0.0009159103$ which was less than the significance threshold of $\alpha\, w_{\mathcal{R}} = 0.001200587$ and for odd positions, the combined p-value was $p^{\circ}_{p_{\mathcal{R}}} = 0.0008619354$ which was less than the significance threshold of $\alpha\, w_{\mathcal{R}} = 0.001193928$.

Comparing p-values with different significance thresholds can be confusing. Instead, it is useful to calculate **adjusted *p*-values**, which are compared directly to the nominal significance threshold α. An adjusted p-value is simply divided by its weight w. For example:

```
R = which(gwas$pos%%2==0)
p.R = p.hmp(gwas$p[R],gwas$w[R])
w.R = sum(gwas$w[R])
(p.R.adjust = p.R/w.R)

##      p.hmp
## 0.03814426

R = which(gwas$pos%%2==1)
p.R = p.hmp(gwas$p[R],gwas$w[R])
w.R = sum(gwas$w[R])
(p.R.adjust = p.R/w.R)

##      p.hmp
## 0.03609663
```

Now it is easy to see that we can rule out the null hypotheses of no significant p-values for both even-numbered and odd-numbered positions, assuming $\alpha = 0.05$.

Of course it makes little sense to combine p-values according to whether their position is an even or odd number. Instead we might wish to test the first 156229 SNPs on chromosome 12 separately from the second 156228 SNPs to begin to narrow down regions of significance.

```
R = 1:156229
p.R = p.hmp(gwas$p[R],gwas$w[R])
w.R = sum(gwas$w[R])
(p.R.adjust = p.R/w.R)

##    p.hmp
## 6.582738

R = 156230:312457
p.R = p.hmp(gwas$p[R],gwas$w[R])
w.R = sum(gwas$w[R])
(p.R.adjust = p.R/w.R)

##      p.hmp
## 0.01855863
```

This is much clearer: only in the second half of the chromosome can we reject the null hypothesis of no significant p-values at the $\alpha = 0.05$ level. For the first half of the chromosome, the adjusted p-value was $p_{\circ_{p_{\mathcal{R}}}}/w_{\mathcal{R}} = 6.582738$. While p-values must be 1 or below, adjusted p-values need not be. For the second half of the chromosome, the adjusted p-value was $p_{\circ_{p_{\mathcal{R}}}}/w_{\mathcal{R}} = 0.01855863$ which is below the standard significance threshold of $\alpha = 0.05$.

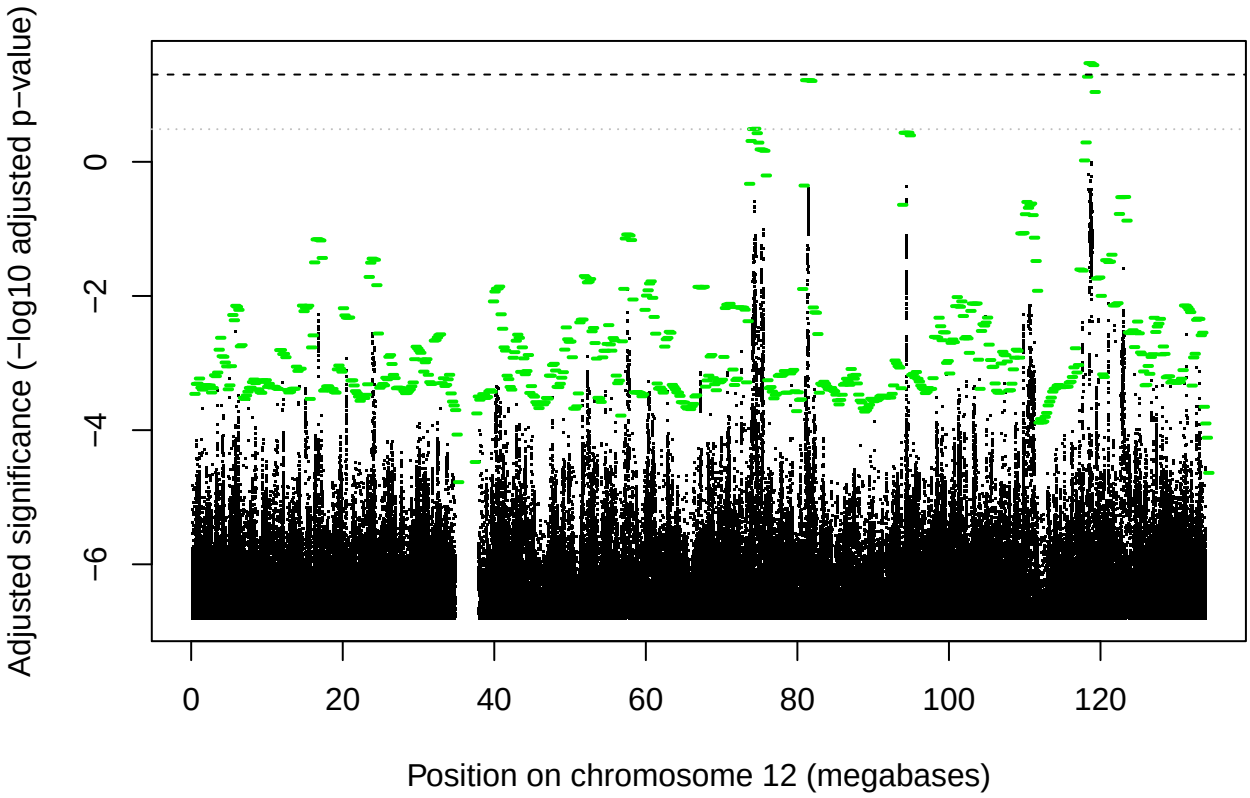
Note that it was completely irrelevant that we had already performed tests of even- and odd-positioned SNPs: as mentioned above, the significance thresholds are pre-determined by the $w_{\mathcal{R}}$'s no matter how many subsets of p-values are tested and no matter in what combinations. We can test any subset of the p-values without incurring further multiple testing penalties. For example, let's test 1 megabase windows overlaping at 0.2 megabase intervals. Testing overlapping versus non-overlapping windows has no effect on the significance thresholds, but of course it has an effect on the resolution of our conclusions and on the computational time.

```
# Define overlapping sliding windows of 1 megabase at 0.2 megabase intervals
win.1M.beg = outer(0:floor(max(gwas$pos/1e6)),(0:4)/5,"+")*1e6
# Calculate the combined p-values for each window
system.time({
  p.1M = apply(win.1M.beg,c(1,2),function(beg) {
```

```
    R = which(gwas$pos>=beg & gwas$pos<(beg+1e6))
    p.hmp(gwas$p[R],gwas$w[R])
  })
})

##    user  system elapsed
##  15.887   0.784  16.679

# Calculate sums of weights for each combined test
system.time({
  w.1M = apply(win.1M.beg,c(1,2),function(beg) {
    R = which(gwas$pos>=beg & gwas$pos<(beg+1e6))
    sum(gwas$w[R])
  })
})

##    user  system elapsed
##  12.145   0.801  12.952

# Calculate adjusted p-value for each window
p.1M.adj = p.1M/w.1M
```

Now plot them

```
# Took a few seconds, plotting over 312k points
gwas$p.adj = gwas$p/gwas$w
plot(gwas$pos/1e6,-log10(gwas$p.adj),pch=".",xlab="Position on chromosome 12 (megabases)",
  ylab="Adjusted significance (-log10 adjusted p-value)",
  ylim=sort(-log10(range(gwas$p.adj,p.1M.adj,na.rm=TRUE))))
arrows(win.1M.beg/1e6,-log10(p.1M.adj),(win.1M.beg+1e6)/1e6,len=0,col="green2",lwd=2)
# Superimpose the significance threshold, alpha, e.g. alpha=0.05
abline(h=-log10(0.05),col="black",lty=2)
# For comparison, plot the conventional GWAS threshold of 5e-8. Need to convert
# this into the adjusted p-value scale. Instead of comparing each raw p-value
# against a Bonferonni threshold of alpha/L=0.05/6524432, we would be comparing each
# against 5e-8. So the adjusted p-values p/w=p*L would be compared against
# 5e-8*L = 5e-8 * 6524432 = 0.3262216
abline(h=-log10(0.3262216),col="grey",lty=3)
```



The black dashed line shows the $\alpha = 0.05$ significance threshold and the grey dotted line shows $\alpha = 0.326$. Evaluating the adjusted p-values against the latter threshold produces a procedure equivalent to applying a threshold of 5×10^{-8} to the raw p-values, which has been adopted as a convention in human GWAS.

No SNPs are individually significant by either threshold. However, the HMP detects several 1 megabase regions of significance. List their positions

```
win.1M.beg[which(p.1M.adj<=0.05)]

## [1] 118000000 118200000 118400000 118600000

# Also list the position of the most significant individual (adjusted) p-value
(peakpos = gwas$pos[gwas$p.adj==min(gwas$p.adj)])

## [1] 118876918
```

As you can see, the genome-wide significant regions are consecutive windows spanning from megabases 118.0-118.6 to 119.0-119.6 on chromosome 12. Not too surprisingly, the region encompasses the most significant individual SNP at position 118876918.

A natural question is **can we identify the smallest groups of significant p-values?** As explained above, there is no penalty for conducting the extra *combined* tests required to answer this question because the significance thresholds are all predetermined. This is because conceptually the HMP is a *multilevel test*, meaning that when a combined test is conducted, all subsets of combined tests are implicitly performed at the same time - although to perform them explicitly requires additional computation.

Let's test windows of varying lengths centred on the most significant individual SNP (which recall is not, by itself, genome-wide significant).

```
# Window of 100 base pairs
wlen = 100
R = which(abs(gwas$pos-peakpos)<wlen)
(p.R.adjust = p.hmp(gwas$p[R])/sum(gwas$w[R]))

##     p.hmp
## 0.9396381

# Window of 1 kilobase
wlen = 1e3
R = which(abs(gwas$pos-peakpos)<wlen)
(p.R.adjust = p.hmp(gwas$p[R])/sum(gwas$w[R]))

##    p.hmp
## 0.646783

# Window of 10 kilobases
wlen = 1e4
R = which(abs(gwas$pos-peakpos)<wlen)
(p.R.adjust = p.hmp(gwas$p[R])/sum(gwas$w[R]))

##     p.hmp
## 0.2329343

# Window of 100 kilobases
wlen = 1e5
R = which(abs(gwas$pos-peakpos)<wlen)
(p.R.adjust = p.hmp(gwas$p[R])/sum(gwas$w[R]))

##      p.hmp
## 0.08232894

# Window of 1 megabase
wlen = 1e6
R = which(abs(gwas$pos-peakpos)<wlen)
(p.R.adjust = p.hmp(gwas$p[R])/sum(gwas$w[R]))

##      p.hmp
## 0.03382166
```

The 1 megabase window centred on position 118876918 is genome-wide significant at $\alpha = 0.05$, but the 100 kilobase window is not. So this manual approach did not get us much closer. With the HMP, it is valid to use optimization to find the smallest significant group of SNPs centred on position 118876918.

```
# Find the smallest window centred on position 118876918 significant at alpha=0.05
f = function(wlen) {
  R = which(abs(gwas$pos-peakpos)<wlen)
  p.R.adjust = p.hmp(gwas$p[R])/sum(gwas$w[R])
  return(p.R.adjust - 0.05)
}
(wlen.opt = uniroot(f,c(1e5,1e6))$root)

## [1] 166197

# Show that the group of SNPs in this window is indeed significant
wlen = wlen.opt
R = which(abs(gwas$pos-peakpos)<wlen)
(p.R.adjust = p.hmp(gwas$p[R])/sum(gwas$w[R]))
```

```
##      p.hmp
## 0.04953369

# The number of individual SNPs included in this group
length(R)

## [1] 820
```

This shows that the smallest significant window occurs with window length 166197 base pairs and encompasses 820 SNPs. In this example, we have optimized in a highly constrained way, only testing combinations of p-values that correspond to consecutive SNPs centred on position 118876918. However, it would be conceptually valid, if computationally challenging, to test all combinations of p-values and report the smallest groups that are significant, after correcting for multiple testing, as always, by using the threshold $p^{\circ}_{p_{\mathcal{R}}} \leq \alpha\, w_{\mathcal{R}}$.

It turns out that the SNPs in this window, spanning chromosome 12 positions 118710721-119042500, overlap the genes *TAOK3* and *SUDS3*. Searching the GWAS database reveals that SNPs in *TAOK3* are associated with the following traits (https://www.ebi.ac.uk/gwas/genes/TAOK3):

- Blood protein levels
- Glucose homeostasis traits
- Morphine dose requirement in tonsillectomy and adenoidectomy surgery
- Neuroticism
- Post bronchodilator FEV1 in COPD
- Red cell distribution width
- Systemic lupus erythematosus

Likewise, SNPs in *SUDS3* are associated with the following traits (https://www.ebi.ac.uk/gwas/genes/SUDS3):

- Depression (broad)
- Hippocampal sclerosis

The association between *TAOK3* and neuroticism was reported in a meta-analysis involving 449,484 individuals (Nagel *et al.*, 2018), 2.6 times as many as Okbay *et al.* (2016), on whose data alone the HMP reveals a genome-wide significant association overlapping this gene, albeit not at the individual SNP level.

2.1 Example 2. Model-averaging in Regression

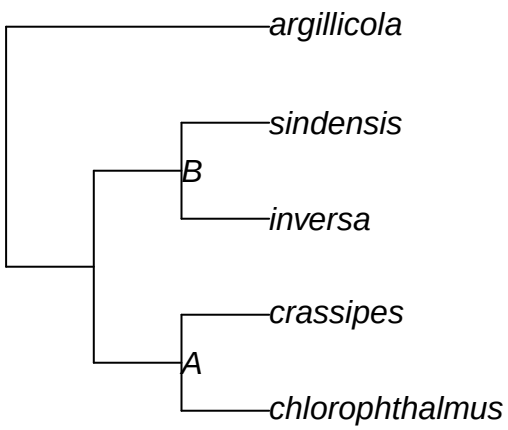
The aim of this example is to use linear regression to test whether a response variable is associated with a regressor of interest, controlling for potential confounding regressors. The problems are that one does not know which of the potential confounders to include (none, some, or all), the confounders are correlated, and power is limited. These complexities are common in the application of linear models, they exemplify a wider class of model selection problems, and they can be addressed by model averaging using the HMP to combine tests.

The specific example comes from the field of comparative phylogenetics, in which a common question is whether two traits are non-randomly associated across different species (Symonds and Blomberg, 2014). Fiddler crabs (genus *Uca*) have enlarged claws which are used in male-to-male competition. Species differ in claw size and the question is whether bigger claws are associated with bigger crabs. However, phylogeny can confound the analysis because species that share common ancestors are likely to have more similar traits including body size and claw size, regardless of other forces driving any association such as developmental constraints or natural selection. (The use of linear regression in this example is for demonstrating the use of the HMP, and is not meant to imply this is the recommended method. Phylogenetic regression (Grafen, 1989) or linear mixed models would normally be preferred).

The phylogeny below shows the relationship between the five species of Fiddler crabs under consideration. The tips of the tree are labelled by the species names and the internal nodes of the tree (the common ancestors) of

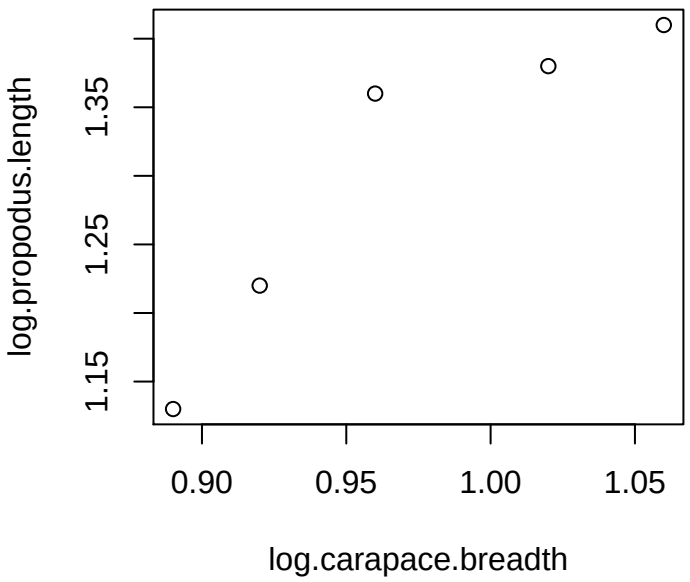
interest[2] are labelled A and B. A scatterplot of claw size (quantified as log propodus length) versus body size (log carapace breadth) is shown. The example data come from Symonds and Blomberg (2014):

```
# Load the ape package for reading and plotting the tree
library(ape)

## Warning:  package 'ape' was built under R version 3.2.3

tree = read.tree((PIPE=pipe(
'echo "(((chlorophthalmus:1,crassipes:1)A:1,(inversa:1,sindensis:1)B:1):1,argillicola:3);"'
))); close(PIPE)
plot(tree, show.node.label=TRUE)
```




```
log.carapace.breadth = c("chlorophthalmus"=1.02,"crassipes"=1.06,"inversa"=0.96,
"sindensis"=0.92,"argillicola"=0.89)
log.propodus.length = c("chlorophthalmus"=1.38,"crassipes"=1.41,"inversa"=1.36,
"sindensis"=1.22,"argillicola"=1.13)

plot(log.propodus.length ~ log.carapace.breadth)
```

[2]Here, ancestral changes in a trait are estimable only for common ancestors of between 2 and $n-2$ species, which is why they are deemed of interest.



The following code defines the data frame for further analysis

```
# Convert branches in the tree into informative 'partitions'
informative.partitions = function(tree) {
  n = length(tree$tip.label)
  m = sapply(n+1:tree$Nnode,function(node) {
    1*is.na(match(tree$tip.label,extract.clade(tree,node)$tip.label))
  })
  rownames(m) = tree$tip.label
  colnames(m) = paste0("node.",tree$node.label)
  cs = colSums(m)
  is.informative = pmin(cs,n-cs)>1
  m[,is.informative]
}
# Extract phylogenetically informative partitions from the tree
partition = informative.partitions(tree)

# Create a data frame combining all the information
Uca = data.frame(log.propodus.length,log.carapace.breadth,partition)
```

The essence of the model selection problem is that we wish to test for a direct association between claw size and body size but we do not know which confounders to include in the analysis. I.e. there is uncertainty in the model. The principal concerns with the model uncertainty are:

- Which confounders do I need to include?
- How do I avoid unnecessary loss of power?
- How do I control the false positive rate for multiple tests?

Power is lost in a number of ways:

- Fitting too many regressors, which uses up degrees of freedom.
- Fitting mutually correlated regressors, whose effects are difficult to disentangle.

- Applying punitive multiple testing correction.

The counterpoint to the last difficulty is that failing to apply proper multiple testing correction risks inflating the false positive rate. The HMP helps address these problems.

The models can be fully enumerated (in this simple example) as follows:

```
# Claw size does not vary by species
m0 = formula(log.propodus.length ~ 1) # grand null
# Claw size is associated with body size and there is no phylogenetic correlation
m1 = formula(log.propodus.length ~ log.carapace.breadth)
# Claw size isn't associated with body size but it is different in the descendents of ancestor A
m2 = formula(log.propodus.length ~ node.A)
# Claw size isn't associated with body size but it is different in the descendents of ancestor B
m3 = formula(log.propodus.length ~ node.B)
# Claw size is associated with body size and it is different in the descendants of ancestor A
m4 = formula(log.propodus.length ~ log.carapace.breadth + node.A)
# Claw size is associated with body size and it is different in the descendants of ancestor B
m5 = formula(log.propodus.length ~ log.carapace.breadth + node.B)
# Claw size isn't associated with body size but is different in descendents of ancestors A & B
m6 = formula(log.propodus.length ~ node.A + node.B)
# Claw size is associated with body size and is different in descendants of ancestors A & B
m7 = formula(log.propodus.length ~ log.carapace.breadth + node.A + node.B) # grand alternative
# List the alternatives together
mA = list(m1,m2,m3,m4,m5,m6,m7)
```

(More generally, the models can be exhaustively enumerated with the function in Appendix 3.1.) The key point is that for each model that includes body size as a regressor, there is a complementary model that excludes it. Each of these pairs of nested models constitutes a candidate test for the association between claw size and body size:

- Model $\mathcal{M}_1$ versus $\mathcal{M}_0$.
- Model $\mathcal{M}_4$ versus $\mathcal{M}_2$.
- Model $\mathcal{M}_5$ versus $\mathcal{M}_3$.
- Model $\mathcal{M}_7$ versus $\mathcal{M}_6$.

The problems are that the p-values from these four tests may produce different answers, the tests may differ in power, the scenarios may differ in plausibility, and the p-values will not be independent. So how will the results be interpreted? The HMP helps because it allows non-independent tests to be combined. The HMP is not overly sensitive to the weights, so initially I will assume equal weights, but later I will optimize the power of the HMP by accounting for the power of the constituent tests, and their prior plausibility, in the weights.

Before proceeding, it is worth taking a step back, because after using the HMP to perform a model-averaged test for a significant effect of body size on claw size, one might wish to perform analogous tests for the other regressors. I.e. one might wish to test whether the descendants of ancestor A (or B, or both) differ systematically in claw size from the other species. In total we have $s = 3$ regressors for selection in the model (log.carapace.breadth, node.A, node.B), excluding regressors that will be included in every model (in this case, an intercept term). Therefore there are 2^s possible models to fit. For each of the s regressors, there are 2^{s-1} possible tests of significance, because there are 2^{s-1} combinations of the other s regressors. In total, that makes $2^{s-1}s$ possible tests. Clarifying the total number of possible tests is important for setting the weights and, through the weights, the significance thresholds.

- **It is the responsibility of the user to correctly specify the total number of tests if the false positive rate is to be controlled using the HMP.**
- *Do not count combined tests performed using the HMP in this total, only the number of tests that produced the input p-values. Count tests you have not even done yet if you intend to do them.*

Assuming equal weights,

$$w_i = \frac{1}{2^{s-1}s} = \frac{1}{12}$$

for every test. Each individual test has significance threshold αw_i, as usual - this is equivalent to the Bonferroni threshold. The power of the HMP stems from the ability to (i) combine test outcomes and (ii) evaluate them against less stringent thresholds.

To test for a direct association between claw size and body size, we are going to calculate a HMP test statistic

$$\overset{\circ}{p}_{\text{body size}} = \frac{w_{1:0} + w_{4:2} + w_{5:3} + w_{7:6}}{w_{1:0}/p_{1:0} + w_{4:2}/p_{4:2} + w_{5:3}/p_{5:3} + w_{7:6}/p_{7:6}}$$

where the notation $p_{A:0}$ and $w_{A:0}$ indicates the p-value and weight from the test of model $\mathcal{M}_A$ versus $\mathcal{M}_0$. From the HMP test statistic we calculate an asymptotically exact combined p-value $p_{\overset{\circ}{p}_{\text{body size}}}$ which we then evaluate against significance threshold $\alpha\, w_{\text{body size}}$ where

$$w_{\text{body size}} = w_{1:0} + w_{4:2} + w_{5:3} + w_{7:6}.$$

[Recall that $p_{\overset{\circ}{p}_{\text{body size}}} \approx \overset{\circ}{p}_{\text{body size}}$ when $\overset{\circ}{p}_{\text{body size}} \ll 1$. Note also that the calculation of $p_{\overset{\circ}{p}_{\text{body size}}}$ via $\overset{\circ}{p}_{\text{body size}}$ will be performed in a single R command (p.hmp).]

The first step is to calculate the p-values that are to be combined. Since the total number of tests is large, we will calculate them in batches immediately before combining them. There is one p-value for each possible pair of models that include and do not include body size (log carapace breadth).

```
# Output p-values from all tests for the inclusion of the primary regressor
pairwise.p = function(response,primary,data) {
  # Define a model space including the grand null
  rid = which(colnames(data)==response)
  if(length(rid)!=1) stop("Could not find response variable")
  # Define the 'primary' regressor
  pid = which(colnames(data)==primary)
  if(length(pid)!=1) stop("Could not find primary regressor")
  # Define the 'secondary' regressors, excluding the response and 'primary' regressor
  xid = (1:ncol(data))[-c(rid,pid)]
  if(length(xid)<1) stop("Could find only the primary regressor")
  # Create a table of every unique combination of models involving the secondary regressors
  delta = expand.grid(lapply(xid,function(j) 0:1))
  colnames(delta) = colnames(data)[xid]
  # Sort them by the number of regressors included, from fewest to most
  delta = delta[order(rowSums(delta)),]
  # Enumerate the models, adding the primary regressor to every one
  mpairs = apply(delta,1,function(x) {
    if(all(x==0)) {
      formula(paste0(colnames(data)[rid],"~",colnames(data)[pid]))
    } else {
      formula(paste0(colnames(data)[rid],"~",colnames(data)[pid],"+",
        paste(colnames(data)[xid][x==1],collapse="+")))
    }
  })
  names(mpairs) = gsub(colnames(data)[pid],paste0("[",colnames(data)[pid],"]"),
    as.character(mpairs),perl=TRUE)
  # Calculate a p-value for the inclusion of the primary regressor in each model
  lapply(mpairs,function(m) {
    fit = lm(m, data=data)
    drop1(fit,colnames(data)[pid],test="Chisq")[colnames(data)[pid],"Pr(>Chi)"]
  })
}
# Calculate the p-values from all tests for the inclusion of log.carapace.breadth
(p = pairwise.p(response="log.propodus.length",primary="log.carapace.breadth",data=Uca))

## $`log.propodus.length ~ [log.carapace.breadth]`
```

```
## [1] 0.002120586
##
## $`log.propodus.length ~ [log.carapace.breadth] + node.A`
## [1] 0.001897942
##
## $`log.propodus.length ~ [log.carapace.breadth] + node.B`
## [1] 0.0001486883
##
## $`log.propodus.length ~ [log.carapace.breadth] + node.A + node.B`
## [1] 0.01350417
```

Next one performs the combined test for an association between claw size (the response) and body size (the regressor of interest). It is essential to get the weights right. Although we are only combining four p-values right now, there are $2^{s-1}s = 12$ tests in total:

```
# Specify the weight of each test, assuming equal weights
ntotaltests = 12
(w = rep(1/ntotaltests,length(p)))

## [1] 0.08333333 0.08333333 0.08333333 0.08333333

# Calculate the model-averaged (asymptotically exact) HMP
(p.comb = p.hmp(p,w))

##        p.hmp
## 0.0005153205

# Sum the weights of the constituent tests
(w.comb = sum(w))

## [1] 0.3333333

# Calculate an adjusted model-averaged p-value for comparison to the ssFWER alpha
(p.comb.adj = p.comb/w.comb)

##       p.hmp
## 0.001545961
```

So we find that, after multiple testing correction, there is a significant association between claw size and body size ($p = 0.0015$) at the $\alpha = 0.05$ level. How does this compare to the individual tests (after multiple testing correction) and Bonferroni correction? [Note that Bonferroni correction is frequently used in both senses - to adjust individual p-values for multiple testing and to perform a combined test by taking the minimum of the adjusted p-values. Recall that HMP and Bonferroni produce the same adjusted p-values for *individual* tests.]

```
(p.adj = unlist(p)/w)

##                   log.propodus.length ~ [log.carapace.breadth]
##                                                     0.02544703
##          log.propodus.length ~ [log.carapace.breadth] + node.A
##                                                     0.02277530
##          log.propodus.length ~ [log.carapace.breadth] + node.B
##                                                     0.00178426
## log.propodus.length ~ [log.carapace.breadth] + node.A + node.B
##                                                     0.16205008

(p.Bonf = min(p.adj))

## [1] 0.00178426
```

We find that the Bonferroni method produces a combined p-value that is also significant ($p = 0.0017$), but less significant than the HMP. The difference is small in this example because one adjusted p-value dominated, in the sense of being much smaller than the others.

Repeat the above for the other two regressors:

```
# Is there a significant difference in claw size between the descendants of ancestor A
# and other species?
p = pairwise.p(response="log.propodus.length",primary="node.A",data=Uca)
w = rep(1/ntotaltests,length(p))
p.hmp(p,w)/sum(w)

##      p.hmp
## 0.03830541

# Individual tests and Bonferroni
(p.adj = unlist(p)/w)

##                                  log.propodus.length ~ [node.A]
##                                                      0.64798554
##          log.propodus.length ~ [node.A] + log.carapace.breadth
##                                                      0.57382974
##                         log.propodus.length ~ [node.A] + node.B
##                                                      0.04067963
## log.propodus.length ~ [node.A] + log.carapace.breadth + node.B
##                                                      7.02317893

(p.Bonf = min(p.adj))

## [1] 0.04067963

# Is there a significant difference in claw size between the descendants of ancestor B
# and other species?
p = pairwise.p(response="log.propodus.length",primary="node.B",data=Uca)
w = rep(1/ntotaltests,length(p))
p.hmp(p,w)/sum(w)

##    p.hmp
## 0.183788

# Individual tests and Bonferroni
(p.adj = unlist(p)/w)

##                                  log.propodus.length ~ [node.B]
##                                                     10.3740094
##          log.propodus.length ~ [node.B] + log.carapace.breadth
##                                                      0.3084609
##                         log.propodus.length ~ [node.B] + node.A
##                                                      0.3220758
## log.propodus.length ~ [node.B] + log.carapace.breadth + node.A
##                                                      2.9282418

(p.Bonf = min(p.adj))

## [1] 0.3084609
```

We find that the descendants of ancestor A do have significantly different claw sizes from other species, taking into account uncertainty in the model, but the descendants of ancestor B do not.

2.2 Optimizing the Weights in the Comparative Phylogenetics Example

So far equal weights have been used in the calculation of the HMP. Equal weights are defensible because theory and simulations show that the HMP is robust to the choice of weights (e.g. Wilson (2019) SI Figure S3). Nevertheless, the power of the HMP can be optimized by specifying weights that are informative about (i) the prior probability that the alternative hypothesis associated with each p-value is true and (ii) the power of the test associated with each p-value. The optimal significance threshold for p-value i is approximately

$$\alpha\, w_i = \left(\frac{\mu_i \xi_i}{\lambda}\right)^{\frac{1}{1-\xi_i}} \tag{1}$$

(Wilson (2019) SI Equation 62, after rescaling λ by $1/\alpha$). In this equation:

- μ_i is the prior probability that the alternative hypothesis associated with p_i is true, normalized so that $\sum_{i=1}^{L} \mu_i = 1$.
- ξ_i is a parameter describing the distribution of p_i under the alternative hypothesis, assuming it can be approximated by a Beta $(\xi_i, 1)$ distribution. This is an L-shaped distribution representing the enrichment of p-values near zero under the alternative hypothesis. The parameter ranges from $\xi_i = 0$ (optimally informative test) to $\xi_i = 1$ (completely uninformative test).
- λ is a normalizing constant that is chosen to impose the constraint that $\sum_{i=1}^{L} w_i = 1$.

The optimal weights (or, equivalently, the optimal thresholds) are therefore a non-linear function of the μ_is and ξ_is except when all the ξ_is are much less than one, in which case $1/(1-\xi_i) \approx 1$. To illustrate how the weights are optimized by specifying these variables, I will use an the Fiddler crabs example.

2.2.1 Prior specification

Specifying the relative probability that alternative hypothesis $\mathcal{M}_i$ is true requires some thought. In some cases it might be reasonable to assume all alternatives are equally likely, for example if every model contains the same number of regressors. However, in model selection problems, the number of possible models increases as the number of regressors included in the model increases, so a uniform prior on the alternative hypotheses has a built-in preference for more complex models, which may not be desired.

A simple prior for model selection might be based around a single probability, m, that any regressor is included in the model, which is the same for every regressor. Then the prior probability of alternative hypothesis $\mathcal{M}_i$ is

$$\mu_i = \Pr(\mathcal{M}_i) = m^{\tau_i} (1-m)^{s-\tau_i}$$

where τ_i is the number of regressors, out of s, included in alternative hypothesis $\mathcal{M}_i$, excluding terms included in every model (such as an intercept) and $L = 2^s - 1$ is the total number of alternative hypotheses. Smaller values of m will favour less complex models.

In the Fiddler crab example there are $2^{s-1}s$ tests, around a factor $s/2$ more than the number of models, because some models, such as the grand alternative, are compared to several different nested null hypotheses. Each test is assigned the prior probability of its alternative hypothesis; in general these probabilities may not sum to one, but this is unimportant for Equation 1 because the normalizing constant will be absorbed by λ.

In what follows, I will assume *a priori* that $m = 1/s$ so the expected number of regressors averaged over all models (the alternatives and the grand null) is one. I will begin by performing all tests so I have them in one place:

```
p = c(
  unlist(pairwise.p(response="log.propodus.length",primary="log.carapace.breadth",data=Uca)),
  unlist(pairwise.p(response="log.propodus.length",primary="node.A",data=Uca)),
  unlist(pairwise.p(response="log.propodus.length",primary="node.B",data=Uca)))
```

I will count the number of terms in each alternative hypothesis and calculate an (unnormalized) prior probability

```
terms = lapply(names(p),function(s) labels(terms(as.formula(gsub("\\[|\\]","",s,perl=TRUE)))))
(nterms = unlist(lapply(terms,length)))

##  [1] 1 2 2 3 1 2 2 3 1 2 2 3
```

```
(s = ncol(Uca)-1)

## [1] 3

(m = 1/s)

## [1] 0.3333333

(mu = m^nterms * (1-m)^(s-nterms))

##  [1] 0.14814815 0.07407407 0.07407407 0.03703704 0.14814815 0.07407407
##  [7] 0.07407407 0.03703704 0.14814815 0.07407407 0.07407407 0.03703704
```

2.2.2 Power specification

The power of the test associated with an individual p-value, defined as the probability of significance given the alternative hypothesis is true, might appear more objective than specifying a prior probability, and it would be but for the problem that power can only be calculated with respect to specific values (or distributions of values) of the parameters under the alternative hypothesis. Wilson (2019) Equation 58 shows that the power of the Wald test for the inclusion of a single regressor in a linear model is

$$\Pr\left(p_i < \alpha\right) \approx \Pr\left(\chi_1^2 > \frac{Q_{\chi_1^2}\left(1-\alpha\right)}{1 + \frac{\sigma_\beta^2}{\sigma_\epsilon^2}\left\{\left(X'X\right)^{-1}\right\}_{cc}^{-1}}\right)$$

where χ_1^2 is a chi-squared random variable with 1 degree of freedom, $Q_{\chi_1^2}\left(1-\alpha\right)$ is the critical value of χ_1^2 for significance at level α, X is the matrix of regressors, including intercepts, of which column c corresponds to the regressor whose inclusion is being tested, and $\sigma_\beta^2/\sigma_\epsilon^2$ is the relative magnitude of the variance of a Normal prior on the effect size of the regressor-of-interest, compared to the error variance. This last term must be chosen carefully for every regressor.

In what follows, I will assume $\sigma_\beta^2/\sigma_\epsilon^2 = 2/V\left(X_c\right)$, i.e. the expected magnitude of the effect size of regressor c is twice the expected magnitude of the error term, after standardizing the variances of all the regressors to equal one.

```
test.term = sapply(names(p),function(s)
  gsub("\\[|\\]","",regmatches(s,regexpr("\\[.*?\\]",s,perl=TRUE)),perl=TRUE)
)
Uca.var = apply(Uca,2,var)
ssqb.over.ssqe = 2/Uca.var[test.term]
names(ssqb.over.ssqe) = names(p)
Var.beta.over.ssqe = sapply(names(p),function(s) {
  test.term = gsub("\\[|\\]","",
    regmatches(s,regexpr("\\[.*?\\]",s,perl=TRUE)),perl=TRUE)
  X = model.matrix(as.formula(gsub("\\[|\\]","",s,perl=TRUE)), data=Uca)
  solve(crossprod(X))[test.term,test.term]
})

# When the beta approximation performs poorly, best to evaluate
# near the likely value of the final threshold
smallp = 0.05/length(p)
# These are the test powers assuming threshold smallp
(smallp.pow = pchisq(qchisq(smallp,1,lower.tail=FALSE)/
  (1+ssqb.over.ssqe/Var.beta.over.ssqe),1,lower.tail=FALSE))

##                   log.propodus.length ~ [log.carapace.breadth]
##                                                      0.33953383
##          log.propodus.length ~ [log.carapace.breadth] + node.A
```

```
##                                                   0.06068919
##          log.propodus.length ~ [log.carapace.breadth] + node.B
##                                                   0.30416527
## log.propodus.length ~ [log.carapace.breadth] + node.A + node.B
##                                                   0.02584481
##                                  log.propodus.length ~ [node.A]
##                                                   0.33953383
##          log.propodus.length ~ [node.A] + log.carapace.breadth
##                                                   0.06068919
##                         log.propodus.length ~ [node.A] + node.B
##                                                   0.21945873
## log.propodus.length ~ [node.A] + log.carapace.breadth + node.B
##                                                   0.01651147
##                                  log.propodus.length ~ [node.B]
##                                                   0.33953383
##          log.propodus.length ~ [node.B] + log.carapace.breadth
##                                                   0.30416527
##                         log.propodus.length ~ [node.B] + node.A
##                                                   0.21945873
## log.propodus.length ~ [node.B] + log.carapace.breadth + node.A
##                                                   0.10793392
```

```
# Sanity checks
if(any(smallp.pow==1))
  warning("Perfect power test detected, check this is plausible")
if(any(smallp.pow<smallp))
  stop("Tests with worse power than smallp violate assumptions")
if(any(!is.finite(smallp.pow)) | any(smallp.pow<0) | any(smallp.pow>1))
  stop("Power cannot be outside range 0-1")
# Convert them into the parameter of the Beta(xi,1) distribution
xi = log(smallp.pow)/log(smallp)
if(any(!is.finite(xi)) | any(xi<0) | any(xi>1))
  stop("Beta(xi,1): xi cannot be outside range 0-1")

# Optimize the weights
wfunc = function(mu,xi,lambda,alpha) (mu*xi/lambda)^(1/(1-xi))/alpha
lambdafunc = function(lambda,alpha) sum(wfunc(mu,xi,lambda,alpha))-1
lambda = uniroot(lambdafunc,c(1e-6,1e6),0.05)$root
lambda = uniroot(lambdafunc,lambda*c(0.1,1.1),0.05)$root
(w = wfunc(mu,xi,lambda,0.05))
```

```
##                   log.propodus.length ~ [log.carapace.breadth]
##                                                   2.074948e-01
##          log.propodus.length ~ [log.carapace.breadth] + node.A
##                                                   1.874035e-02
##          log.propodus.length ~ [log.carapace.breadth] + node.B
##                                                   8.617939e-02
## log.propodus.length ~ [log.carapace.breadth] + node.A + node.B
##                                                   1.991202e-04
##                                  log.propodus.length ~ [node.A]
##                                                   2.074948e-01
##          log.propodus.length ~ [node.A] + log.carapace.breadth
##                                                   1.874035e-02
##                         log.propodus.length ~ [node.A] + node.B
##                                                   7.693724e-02
## log.propodus.length ~ [node.A] + log.carapace.breadth + node.B
##                                                   7.436276e-06
```

```
##                                log.propodus.length ~ [node.B]
##                                                  2.074948e-01
##          log.propodus.length ~ [node.B] + log.carapace.breadth
##                                                  8.617939e-02
##                         log.propodus.length ~ [node.B] + node.A
##                                                  7.693724e-02
## log.propodus.length ~ [node.B] + log.carapace.breadth + node.A
##                                                  1.359365e-02

# Check the weights sum to one
sum(w)

## [1] 0.9999987

if(abs(1-sum(w))>1e-4) stop("weights do not sum to one, check")
```

Now repeat the previous analyses using the new weights

```
# Compare the weighted and unweighted results
wequal = rep(1/length(w),length(w))
# For log.carapace.breadth
incl = which(test.term == "log.carapace.breadth")
c("weighted"=p.hmp(p[incl],w[incl])/sum(w[incl]),
  "unweighted"=p.hmp(p[incl],wequal[incl])/sum(wequal[incl]))

##   weighted.p.hmp unweighted.p.hmp
##      0.001460927      0.001545961

# For node.A
incl = which(test.term == "node.A")
c("weighted"=p.hmp(p[incl],w[incl])/sum(w[incl]),
  "unweighted"=p.hmp(p[incl],wequal[incl])/sum(wequal[incl]))

##   weighted.p.hmp unweighted.p.hmp
##       0.03971510       0.03830541

# For node.B
incl = which(test.term == "node.B")
c("weighted"=p.hmp(p[incl],w[incl])/sum(w[incl]),
  "unweighted"=p.hmp(p[incl],wequal[incl])/sum(wequal[incl]))

##   weighted.p.hmp unweighted.p.hmp
##        0.1975921        0.1837880

# Headline
incl = 1:length(p)
c("weighted"=p.hmp(p[incl],w[incl])/sum(w[incl]),
  "unweighted"=p.hmp(p[incl],wequal[incl])/sum(wequal[incl]))

##   weighted.p.hmp unweighted.p.hmp
##      0.001405027      0.001480099
```

The weighted HMP produces a (slightly) more significant association between claw size and body size, and a (slightly) more significant headline p-value (rejecting the null that none of the alternatives are true). The weighted HMP produces (slightly) less significant associations between claw size and descent from ancestors A and B. The small magnitude of the differences support the claim that the HMP is relatively insensitive to weights. While the power of the weighted HMP is expected to be better on average (if the prior probabilities and powers are calculated correctly), it does not follow that the result of any particular test will necessarily be more significant.

3 Appendices

3.1 Appendix I. Function to enumerate all possible models

```
enumerate.models = function(response,data) {
  # Define the response variable
  rid = which(colnames(data)==response)
  if(length(rid)!=1) stop("Could not find the response variable")
  # Define the regressors
  xid = (1:ncol(data))[-rid]
  # Create a table defining every unique combination of alternative hypotheses
  delta = expand.grid(lapply(xid,function(j) 0:1))
  colnames(delta) = colnames(data)[xid]
  # Sort them from fewest to most terms
  delta = delta[order(rowSums(delta)),]
  # Remove the grand null model
  delta = delta[rowSums(delta)>0,]
  # Define the grand null model separately
  m0 = formula(paste0(colnames(data)[rid],"~1"))
  # Define the alternative models
  mA = apply(delta,1,function(x) formula(paste0(colnames(data)[rid],"~",
    paste(colnames(data)[xid][x==1],collapse="+"))))
  names(mA) = as.character(mA)
  return(list("m0"=m0,"mA"=mA))
}
# E.g. on the Uca data
enumerate.models("log.propodus.length",Uca)

## $m0
## log.propodus.length ~ 1
## <environment: 0x7fd7e62d2a38>
##
## $mA
## $mA$`log.propodus.length ~ log.carapace.breadth`
## log.propodus.length ~ log.carapace.breadth
## <environment: 0x7fd7e63e4aa0>
##
## $mA$`log.propodus.length ~ node.A`
## log.propodus.length ~ node.A
## <environment: 0x7fd7e63e0498>
##
## $mA$`log.propodus.length ~ node.B`
## log.propodus.length ~ node.B
## <environment: 0x7fd7e63dc478>
##
## $mA$`log.propodus.length ~ log.carapace.breadth + node.A`
## log.propodus.length ~ log.carapace.breadth + node.A
## <environment: 0x7fd7e6025b08>
##
## $mA$`log.propodus.length ~ log.carapace.breadth + node.B`
## log.propodus.length ~ log.carapace.breadth + node.B
## <environment: 0x7fd7e631bea8>
##
## $mA$`log.propodus.length ~ node.A + node.B`
## log.propodus.length ~ node.A + node.B
## <environment: 0x7fd7e63a23c0>
```

```
##
## $mA$`log.propodus.length ~ log.carapace.breadth + node.A + node.B`
## log.propodus.length ~ log.carapace.breadth + node.A + node.B
## <environment: 0x7fd7e639cb48>
```

References

[1] Benjamini, Y., Hochberg, Y. (1995) Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B* 57(1):289–300.

[2] Fisher, R. A. (1934) *Statistical Methods for Research Workers*. (Oliver and Boyd, Edinburgh), Fifth edition.

[3] Grafen, A. (1989) The phylogenetic regression. *Philosophical Transactions of the Royal Society Series B, Biological Sciences* 326(1233):119–157.

[4] Imhof, J. P. (1961) Computing the distribution of quadratic forms in normal variables. *Biometrika* 48(3/4):419–426.

[5] Nagel, M., et al (2018) Meta-analysis of genome-wide association studies for neuroticism in 449,484 individuals identifies novel genetic loci and pathways. *Nature Genetics* 50(7):920–927.

[6] Okbay, A., et al. (2016) Genetic variants associated with subjective well-being, depressive symptoms, and neuroticism identified through genome-wide analyses. *Nature Genetics* 740 48(6):624–633.

[7] Scheffé, H. (1959) *The Analysis of Variance*. Wiley, New York.

[8] Sellke, T., Bayarri, M. J., Berger, J. O. (2001) Calibration of p values for testing precise null hypotheses. *The American Statistician* 55(1):62–71.

[9] Simes, R. J. (1986) An improved Bonferroni procedure for multiple tests of significance. *Biometrika* 73(3):751–754.

[10] Symonds, M. R. E., Blomberg, S. P. (2014) A primer on phylogenetic generalised least squares. In Zsolt Garamszegi, L. (ed) *Modern Phylogenetic Comparative Methods and Their Applications in Evolutionary Biology*, pp. 105-130. Springer, Berlin.

[11] Wilson, D. J. (2019) The harmonic mean p-value for combining dependent tests. *Proceedings of the National Academy of Sciences USA*, in press.