The CAIC package: methods benchmarks.

David Orme

October 4, 2008

This file contains details on the benchmark testing of the implementation of various methods in the package against other existing implementations. The main benchmark dataset is a 200 tip tree grown under a pure-birth, constant-rates model using the `growTree()` function. Three co-varying continuous variables and a two and a three level categorical variable were evolved on the tree. The details of the code are presented in the file 'BenchmarkDataCreator.R', which details the covariation between the continuous traits and the transition rate matrix for the categorical variables. An extra variable also tests the special case of having no variation in the reference variable at a polytomy. The simulated dichotomous tree has also been degraded to contain a few polytomies. The creator file also handles exporting the data and tree into a form suitable for use in CAIC.

1 Benchmarking `crunch()` and `brunch()`.

These benchmarks test the implementation of independent contrast calculations (Felsenstein, 1985) using the `crunch()` and `brunch()` algorithms in the package 'CAIC' against the implementations in the Mac Classic program CAIC v2.6.9 (Purvis and Rambaut, 1995), running in Mac OS 9.2.2 emulation under Mac OS 10.4.10.

The benchmark dataset, created using the the file 'BenchmarkDataCreator.R' and saved as 'Benchmark.Rda' contains the following objects:

`BENCH` A 200 tip tree grown under a pure-birth, constant-rates model using the `growTree()` function.

`BENCHPCrDi213oly` A version of the tree in which six polytomies have been created by collapsing the shortest internal branches.

`BenchData` A data frame of tip data for the trees containing the following variables, either evolved using `growTree()` or modified from such evolved variables:

- `node` Identifies which tip on the tree each row of data relates to.
- `contResp, contExp1, contExp2` Three co-varying continuous variables evolved under Brownian motion along the tree.
- `contExp1NoVar` A version of `contExp1` which has identical values for five more distal polytomies. This is present to test the behaviour of the implementations at polytomies which have no variation in the reference variable.
- `contRespNA, contExp1NA, contExp2NA` As above but with a small proportion (~5%) of missing data.
- `biFact, triFact` A binary and ternary categorical variable evolving under a rate matrix across the tree.
- `biFactNA, triFactNA` As above, but with a small proportion (~5%) of missing data.
- `SppRich` Integer species richness values, with clade sizes taken from a broken stick distribution of 5000 species among the 200 extant tips, but with richness values distributed arbitrarily across tips.

1.1 The `crunch` algorithm

Five analyses were performed in CAIC to benchmark the following tests:

CrDi213 A dichotomous tree with complete data in three continuous variables.

CrDi657 A dichotomous tree with incomplete data in three continuous variables.

CrPl213 A polytomous tree with complete data in three continuous variables.

CrPl413 A polytomous tree with complete data in three continuous variables, but with no variation in the reference variable at some polytomies.

CrPl657 A polytomous tree with incomplete data in three continuous variables.

The log files 'CAIC_BenchTreeDi.log' and 'CAIC_BenchTreePoly.log' describe the input used to run these analyses in CAIC, with the column numbers used to identify variables using the same order as the variables in the data frame `BenchData`. The following code loads the output of the CAIC analysis into R. Each of the data frames contains the standard CAIC contrast table consisting of: the CAIC code for the node, the contrast in each variable, the standard deviation of the contrast, the height of the node, the number of subtaxa descending from the node; and the nodal values of the variables.

```
> CAIC.CrDi213 <- read.delim("../../benchmarks/CAIC_outputs/BenchCAIC.dat_CrDi213")
> CAIC.CrDi657 <- read.delim("../../benchmarks/CAIC_outputs/BenchCAIC.dat_CrDi657")
> CAIC.CrPl213 <- read.delim("../../benchmarks/CAIC_outputs/BenchCAIC.dat_CrPl213")
> CAIC.CrPl413 <- read.delim("../../benchmarks/CAIC_outputs/BenchCAIC.dat_CrPl413")
> CAIC.CrPl657 <- read.delim("../../benchmarks/CAIC_outputs/BenchCAIC.dat_CrPl657")
```

The next section of code duplicates these CAIC analyses using the `crunch()` function. Note that the default internal branch length used in calculations at a polytomy (`polytomy.brlen`) needs to be changed from the 0, which is the default in `crunch()`, to 1, which was the default in CAIC. The `caic.table()` function is used to extract a contrast table from the `crunch()` output, including CAIC style node labels.

```
> load("../../benchmarks/Benchmark.Rda")
> library(CAIC)
> crunch.CrDi213 <- crunch(contResp ~ contExp1 + contExp2, data = BenchData,
+     phy = BENCH, names.col = node, polytomy.brlen = 1)
> crunch.CrDi213.tab <- caic.table(crunch.CrDi213, CAIC.codes = TRUE)
> crunch.CrDi657 <- crunch(contRespNA ~ contExp1NA + contExp2NA, data = BenchData,
+     phy = BENCH, names.col = node, polytomy.brlen = 1)
> crunch.CrDi657.tab <- caic.table(crunch.CrDi657, CAIC.codes = TRUE)
> crunch.CrPl213 <- crunch(contResp ~ contExp1 + contExp2, data = BenchData,
+     phy = BENCHPoly, names.col = node, polytomy.brlen = 1)
> crunch.CrPl213.tab <- caic.table(crunch.CrPl213, CAIC.codes = TRUE)
> crunch.CrPl413 <- crunch(contResp ~ contExp1NoVar + contExp2, data = BenchData,
+     phy = BENCHPoly, names.col = node, polytomy.brlen = 1)
> crunch.CrPl413.tab <- caic.table(crunch.CrPl413, CAIC.codes = TRUE)
> crunch.CrPl657 <- crunch(contRespNA ~ contExp1NA + contExp2NA, data = BenchData,
+     phy = BENCHPoly, names.col = node, polytomy.brlen = 1)
> crunch.CrPl657.tab <- caic.table(crunch.CrPl657, CAIC.codes = TRUE)
```

The CAIC codes can now be used to order the datasets from the two implementations in order to compare the calculated values. The contrasts in the response variable are plotted in Fig. 1, with data from CAIC shown in black and overplotting of data from `crunch` in red. The range in the differences between these values is shown in Table 1, where the maximum differences of around

$\pm 1e-5$ arise from the limited precision of the values saved in the CAIC output. The exceptions occur in the case of variables with no variance at a polytomy — these values are identical but currently have opposite signs between the two implementations.

```
> crunch.CrDi213.tab <- crunch.CrDi213.tab[order(crunch.CrDi213.tab$CAIC.code),
+     ]
> crunch.CrDi657.tab <- crunch.CrDi657.tab[order(crunch.CrDi657.tab$CAIC.code),
+     ]
> crunch.CrPl213.tab <- crunch.CrPl213.tab[order(crunch.CrPl213.tab$CAIC.code),
+     ]
> crunch.CrPl413.tab <- crunch.CrPl413.tab[order(crunch.CrPl413.tab$CAIC.code),
+     ]
> crunch.CrPl657.tab <- crunch.CrPl657.tab[order(crunch.CrPl657.tab$CAIC.code),
+     ]
> CAIC.CrDi213 <- CAIC.CrDi213[order(CAIC.CrDi213$Code), ]
> CAIC.CrDi657 <- CAIC.CrDi657[order(CAIC.CrDi657$Code), ]
> CAIC.CrPl213 <- CAIC.CrPl213[order(CAIC.CrPl213$Code), ]
> CAIC.CrPl413 <- CAIC.CrPl413[order(CAIC.CrPl413$Code), ]
> CAIC.CrPl657 <- CAIC.CrPl657[order(CAIC.CrPl657$Code), ]
```

analysis	V1Min	V1Max	V2Min	V2Max	V3Min	V3Max
CrDi213	-0.000004922	0.000004951	-0.000005003	0.000004869	-0.000004984	0.000004919
CrDi657	-0.000004865	0.000005001	-0.000005003	0.000004869	-0.000004986	0.000004941
CrPl213	-0.000004865	0.000004951	-0.000005003	0.000005116	-0.000004983	0.000004919
CrPl413	-0.914135601	0.800559857	-0.000005003	0.000004869	-0.702904714	0.537203571
CrPl657	-0.000004865	0.000005001	-0.000005003	0.000004869	-0.000004986	0.000004941

Table 1: Range in the differences between crunch and CAIC contrasts.

1.2 The `brunch` algorithm

Eight analyses were performed in CAIC to benchmark the following tests:

BrDi813 and BrPl813 A binary factor as the primary variable with two continuous variables on both the dichotomous and polytomous tree.

BrDi913 and BrPl913 An ordered ternary factor as the primary variable with two continuous variables on both the dichotomous and polytomous tree.

BrDi1057 and BrPl1057 A binary factor and two continuous variables, all with missing data, on both the dichotomous and polytomous tree.

BrDi1157 and BrPl1157 A ordered ternary factor and two continuous variables, all with missing data, on both the dichotomous and polytomous tree.

Again, the log files 'CAIC_BenchTreeDi.log' and 'CAIC_BenchTreePoly.log' describe the input used to run these analyses in CAIC. The following code loads the output of the CAIC analysis into R. Each of the data frames contains the standard CAIC contrast table consisting of: the CAIC code for the node, the contrast in each variable, the standard deviation of the contrast, the height of the node, the number of subtaxa descending from the node; and the nodal values of the variables.

```
> CAIC.BrDi813 <- read.delim("../../benchmarks/CAIC_outputs/BenchCAIC.dat_BrDi813")
> CAIC.BrDi913 <- read.delim("../../benchmarks/CAIC_outputs/BenchCAIC.dat_BrDi913")
> CAIC.BrDi1057 <- read.delim("../../benchmarks/CAIC_outputs/BenchCAIC.dat_BrDi1057")
> CAIC.BrDi1157 <- read.delim("../../benchmarks/CAIC_outputs/BenchCAIC.dat_BrDi1157")
```

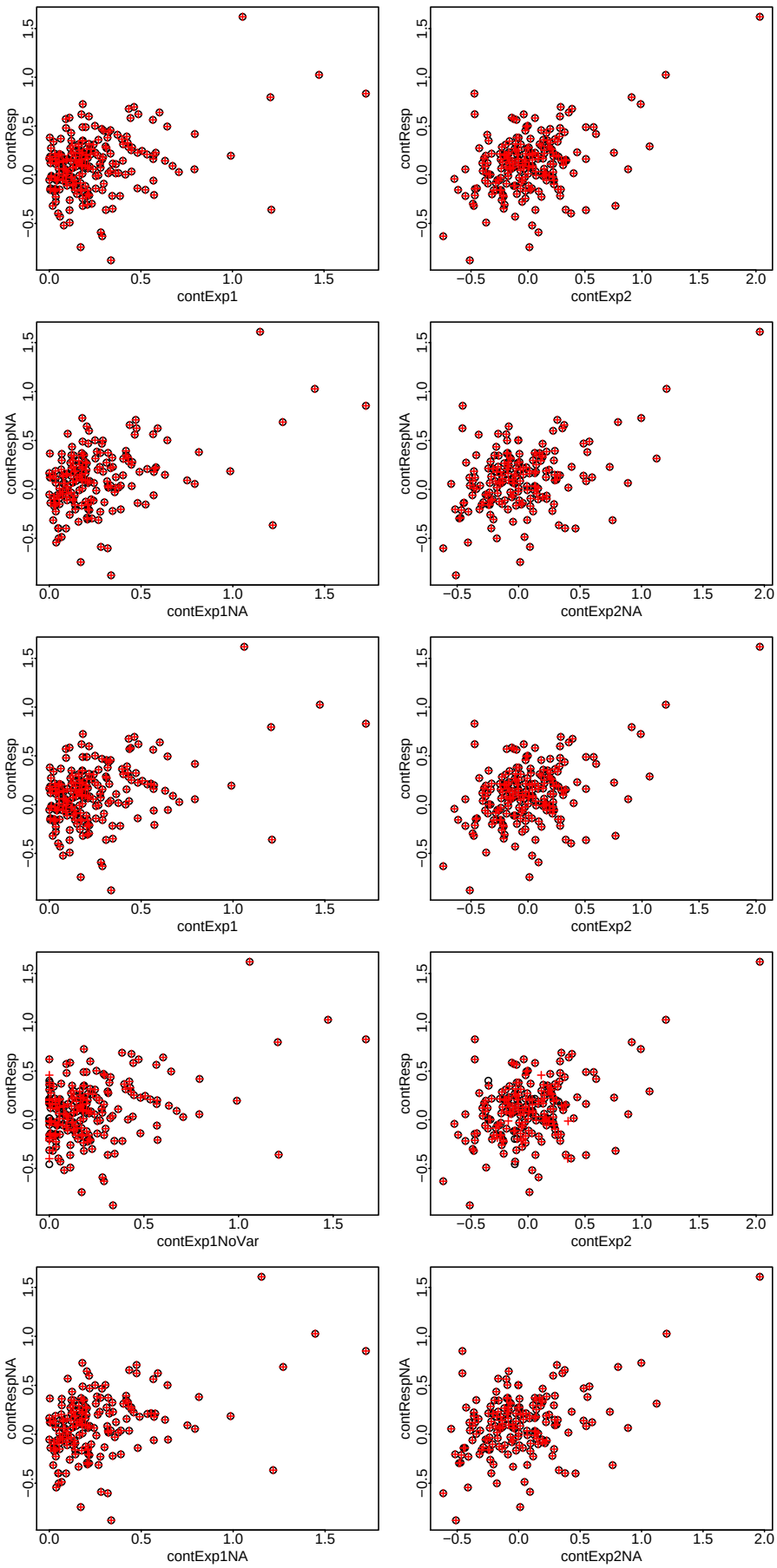

Figure 1: Overplotting of results from CAIC (black) and `crunch()` (red) analyses.

```
> CAIC.BrPl813 <- read.delim("../../benchmarks/CAIC_outputs/BenchCAIC.dat_BrPl813")
> CAIC.BrPl913 <- read.delim("../../benchmarks/CAIC_outputs/BenchCAIC.dat_BrPl913")
> CAIC.BrPl1057 <- read.delim("../../benchmarks/CAIC_outputs/BenchCAIC.dat_BrPl1057")
> CAIC.BrPl1157 <- read.delim("../../benchmarks/CAIC_outputs/BenchCAIC.dat_BrPl1157")
```

The next section of code duplicates these CAIC analyses using the `brunch()` function. Note that `brunch()` does not calculate contrasts at polytomies and the tests using the polytomous tree are to check that the algorithms are drawing the same contrasts from the data. The `caic.table()` function is again used to extract a contrast table from the `brunch()` output.

```
> brunch.BrDi813 <- brunch(contResp ~ binFact + contExp2, data = BenchData,
+     phy = BENCH, names.col = node)
> brunch.BrDi813.tab <- caic.table(brunch.BrDi813, CAIC.codes = TRUE)
> brunch.BrDi913 <- brunch(contResp ~ triFact + contExp2, data = BenchData,
+     phy = BENCH, names.col = node)
> brunch.BrDi913.tab <- caic.table(brunch.BrDi913, CAIC.codes = TRUE)
> brunch.BrDi1057 <- brunch(contRespNA ~ binFactNA + contExp2NA, data = BenchData,
+     phy = BENCH, names.col = node)
> brunch.BrDi1057.tab <- caic.table(brunch.BrDi1057, CAIC.codes = TRUE)
> brunch.BrDi1157 <- brunch(contRespNA ~ triFactNA + contExp2NA, data = BenchData,
+     phy = BENCH, names.col = node)
> brunch.BrDi1157.tab <- caic.table(brunch.BrDi1157, CAIC.codes = TRUE)
> brunch.BrPl813 <- brunch(contResp ~ binFact + contExp2, data = BenchData,
+     phy = BENCHPoly, names.col = node)
> brunch.BrPl813.tab <- caic.table(brunch.BrPl813, CAIC.codes = TRUE)
> brunch.BrPl913 <- brunch(contResp ~ triFact + contExp2, data = BenchData,
+     phy = BENCHPoly, names.col = node)
> brunch.BrPl913.tab <- caic.table(brunch.BrPl913, CAIC.codes = TRUE)
> brunch.BrPl1057 <- brunch(contRespNA ~ binFactNA + contExp2NA, data = BenchData,
+     phy = BENCHPoly, names.col = node)
> brunch.BrPl1057.tab <- caic.table(brunch.BrPl1057, CAIC.codes = TRUE)
> brunch.BrPl1157 <- brunch(contRespNA ~ triFactNA + contExp2NA, data = BenchData,
+     phy = BENCHPoly, names.col = node)
> brunch.BrPl1157.tab <- caic.table(brunch.BrPl1157, CAIC.codes = TRUE)
```

The CAIC codes can again now be used to order the datasets from the two implementations in order to compare the calculated values. The contrasts from each test are overplotted in Fig. 2 and Fig. 3 and the range in the differences between these values is shown in Table 2.

```
> brunch.BrDi813.tab <- brunch.BrDi813.tab[order(brunch.BrDi813.tab$CAIC.code),
+     ]
> brunch.BrDi913.tab <- brunch.BrDi913.tab[order(brunch.BrDi913.tab$CAIC.code),
+     ]
> brunch.BrDi1057.tab <- brunch.BrDi1057.tab[order(brunch.BrDi1057.tab$CAIC.code),
+     ]
> brunch.BrDi1157.tab <- brunch.BrDi1157.tab[order(brunch.BrDi1157.tab$CAIC.code),
+     ]
> brunch.BrPl813.tab <- brunch.BrPl813.tab[order(brunch.BrPl813.tab$CAIC.code),
+     ]
> brunch.BrPl913.tab <- brunch.BrPl913.tab[order(brunch.BrPl913.tab$CAIC.code),
+     ]
> brunch.BrPl1057.tab <- brunch.BrPl1057.tab[order(brunch.BrPl1057.tab$CAIC.code),
+     ]
> brunch.BrPl1157.tab <- brunch.BrPl1157.tab[order(brunch.BrPl1157.tab$CAIC.code),
+     ]
> CAIC.BrDi813 <- CAIC.BrDi813[order(CAIC.BrDi813$Code), ]
> CAIC.BrDi913 <- CAIC.BrDi913[order(CAIC.BrDi913$Code), ]
```

```
> CAIC.BrDi1057 <- CAIC.BrDi1057[order(CAIC.BrDi1057$Code), ]
> CAIC.BrDi1157 <- CAIC.BrDi1157[order(CAIC.BrDi1157$Code), ]
> CAIC.BrPl813 <- CAIC.BrPl813[order(CAIC.BrPl813$Code), ]
> CAIC.BrPl913 <- CAIC.BrPl913[order(CAIC.BrPl913$Code), ]
> CAIC.BrPl1057 <- CAIC.BrPl1057[order(CAIC.BrPl1057$Code), ]
> CAIC.BrPl1157 <- CAIC.BrPl1157[order(CAIC.BrPl1157$Code), ]
```

analysis	V1Min	V1Max	V2Min	V2Max	V3Min	V3Max
BrDi813	-0.000004737	0.000004933	0.000000000	0.000000000	-0.000004932	0.000004812
BrPl813	-0.010980512	0.000004933	0.000000000	0.000000000	-0.000004932	0.000523125
BrDi1057	-0.000004863	0.000004865	0.000000000	0.000000000	-0.000004932	0.000004722
BrPl1057	-0.000004737	0.000004865	0.000000000	0.000000000	-0.000004932	0.000004722
BrDi913	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
BrPl913	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
BrDi1157	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
BrPl1157	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000

Table 2: Range in the differences between brunch and CAIC contrasts.

2 Benchmarking `macrocaic()`.

The function `macrocaic()` is a re-implementation of the program 'MacroCAIC' (Agapow and Isaac, 2002), which calculates standard contrasts (Felsenstein, 1985) for the explanatory variables but alternative contrasts in clade species richness for the explanatory variable. The program calculates two species richness contrast types which are included in the output file. Thes are 'PDI', the proportion dominance index, and 'RRD', the relative rate difference (Agapow and Isaac, 2002). The benchmark tests included 8 sets of analyses using all combinations of the following:

Di/Poly The benchmark tree used: either dichotomous or polytomous.

Spp/Tax Whether the tips are treated as single species or as taxa containing a known number of species.

23/67 The completeness of the explanatory data used, where 2 and 3 are two complete continuous variables and the variables in 6 and 7 have missing data.

The results from the original program are read in by the following code, sorted by the CAIC codes for the internal nodes and assigned to objects of the form 'MacroCAIC.DiSpp23'. The original 'MacroCAIC' log files from these analyses are concatenated together in the file 'MacroCAIC_bench.log'.

```
> combos <- expand.grid(c("Di", "Poly"), c("Spp", "Tax"), c("23",
+     "67"))
> comboNames <- apply(combos, 1, paste, collapse = "")
> for (cm in comboNames) {
+     objName <- paste("MacroCAIC", cm, sep = ".")
+     fName <- paste("../../benchmarks/MacroCAIC_outputs/", cm, ".txt",
+         sep = "")
+     MacroCAICTab <- read.delim(fName)
+     MacroCAICTab <- MacroCAICTab[order(MacroCAICTab$Code), ]
+     assign(objName, MacroCAICTab)
+ }
```

The next section of code repeats each of these analyses using the function `macrocaic()` and then sorts the contrasts from the two implementations into the same order. The resulting contrasts

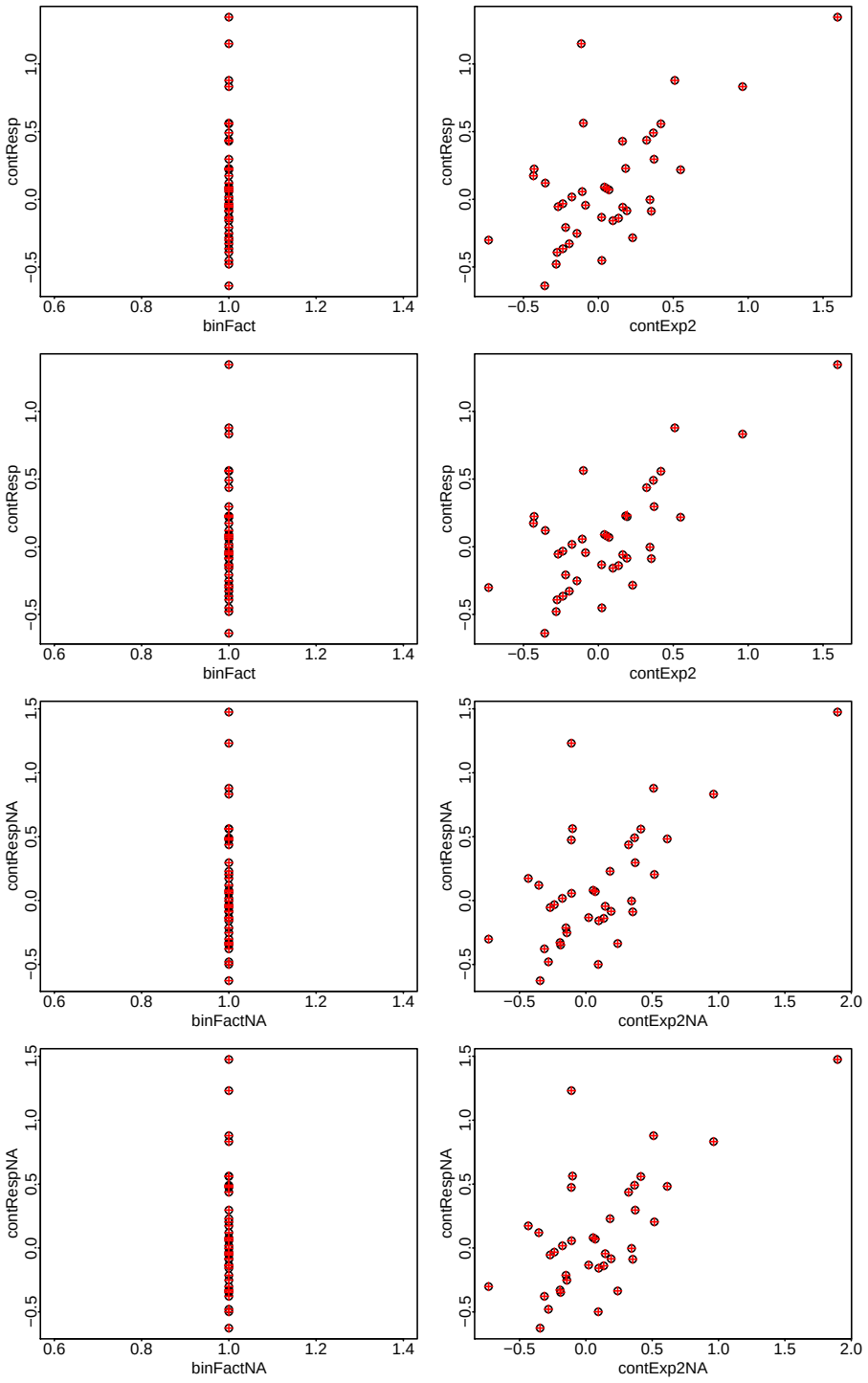

Figure 2: Overplotting of results from CAIC (black) and `brunch()` (red) analyses on the dichotomous tree.

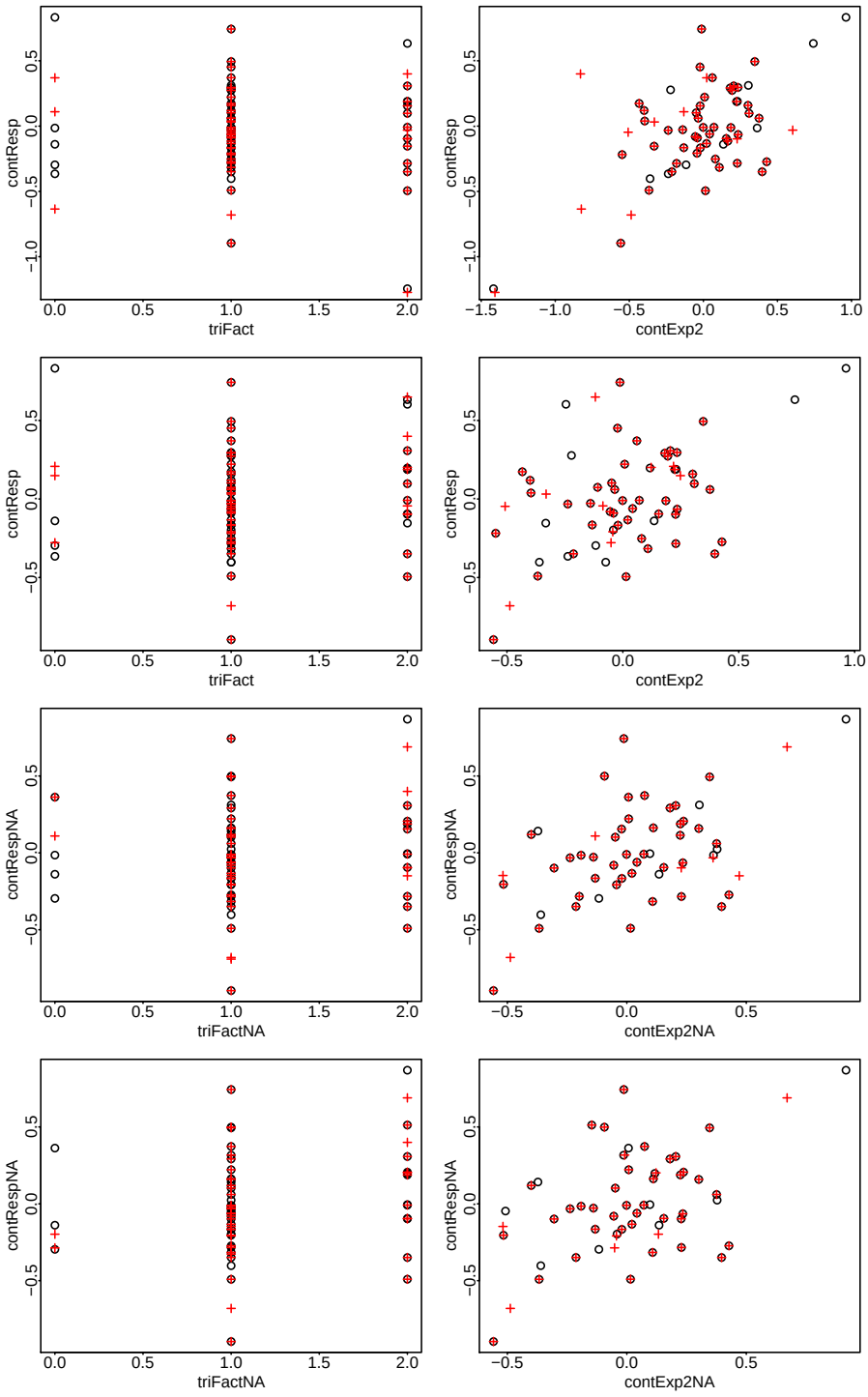

Figure 3: Overplotting of results from CAIC (black) and `brunch()` (red) analyses using a three level factor.

are overplotted on the outputs from 'MacroCAIC' for both RRD and PDI using complete and incomplete data (Figs. 4,5, 6,7). Again, the differences between the contrasts calculated at each node by each implementation are shown in Table 3.

```
> BenchData$TipsAsSpecies <- rep(1, 200)
> for (cm in seq(along = combos[, 1])) {
+     if (combos[cm, 1] == "Di")
+         currTree <- BENCH
+     else currTree <- BENCHPoly
+     if (combos[cm, 2] == "Spp")
+         currRich <- "TipsAsSpecies"
+     else currRich <- "SppRich"
+     if (combos[cm, 3] == "23")
+         currExpl <- "contExp1  + contExp2"
+     else currExpl <- "contExp1NA + contExp2NA"
+     fm <- as.formula(paste(currRich, "~", currExpl))
+     RRDMod <- macrocaic(fm, data = BenchData, phy = currTree, names.col = node,
+         macroMethod = "RRD")
+     PDIMod <- macrocaic(fm, data = BenchData, phy = currTree, names.col = node,
+         macroMethod = "PDI")
+     RRDName <- paste("mcRRD", comboNames[cm], sep = ".")
+     PDIName <- paste("mcPDI", comboNames[cm], sep = ".")
+     RRDTab <- caic.table(RRDMod, CAIC.codes = TRUE, nodalValues = TRUE)
+     RRDTab <- RRDTab[order(RRDTab$CAIC.code), ]
+     PDITab <- caic.table(PDIMod, CAIC.codes = TRUE, nodalValues = TRUE)
+     PDITab <- PDITab[order(PDITab$CAIC.code), ]
+     assign(RRDName, RRDTab)
+     assign(PDIName, PDITab)
+ }
```

analysis	RRDMin	RRDMax	PDIMin	PDIMax	Exp1Min	Exp1Max	Exp2Min	Exp2Max
DiSpp23	-4.92e-08	1.01e-07	-1.38e-08	1.19e-08	-2.76e-07	6.23e-07	-3.36e-07	2.19e-07
PolySpp23	-4.92e-08	1.01e-07	-1.38e-08	1.19e-08	-2.76e-07	6.23e-07	-2.19e-07	2.19e-07
DiTax23	-1.32e-07	1.09e-07	-1.49e-08	1.48e-08	-8.99e-07	6.23e-07	-5.15e-07	2.19e-07
PolyTax23	-1.32e-07	1.09e-07	-1.49e-08	1.48e-08	-8.99e-07	6.23e-07	-5.15e-07	2.19e-07
DiSpp67	-4.92e-08	1.01e-07	-1.38e-08	1.19e-08	-2.28e-07	2.34e-07	-1.99e-07	2.16e-07
PolySpp67	-4.92e-08	1.01e-07	-1.38e-08	1.19e-08	-2.25e-07	2.06e-07	-2.19e-07	2.16e-07
DiTax67	-1.32e-07	1.09e-07	-1.49e-08	1.48e-08	-8.99e-07	3.61e-07	-5.15e-07	2.16e-07
PolyTax67	-1.32e-07	1.09e-07	-1.49e-08	1.48e-08	-8.99e-07	3.61e-07	-5.15e-07	2.16e-07

Table 3: Range of differences between contrasts calcualted using MacroCAIC and the function `macrocaic()`.

3 Benchmarking `fusco.test()`.

3.1 Testing against the original FUSCO implementation.

This runs tests against Giuseppe Fusco's implementation of the phylogenetic imbalance statistic I (Fusco and Cronk, 1995). The original package consists of two DOS programs 'IMB_CALC' and 'NULL_MDL' which calculate the imbalance of phylogeny and then the expected distribution of imbalance values on a equal rates Markov tree, given the size of the tree. The program contains the option to consider the imbalance of the topology, treating each tip as a species, or of the distribution of species across tips, treating each tip as a taxon of one or more species. Each output file in the 'benchmarks' folder contains the binned distribution data across nodes, some summary data for the tree and then the individual node calculations. These benchmark tests use both options on

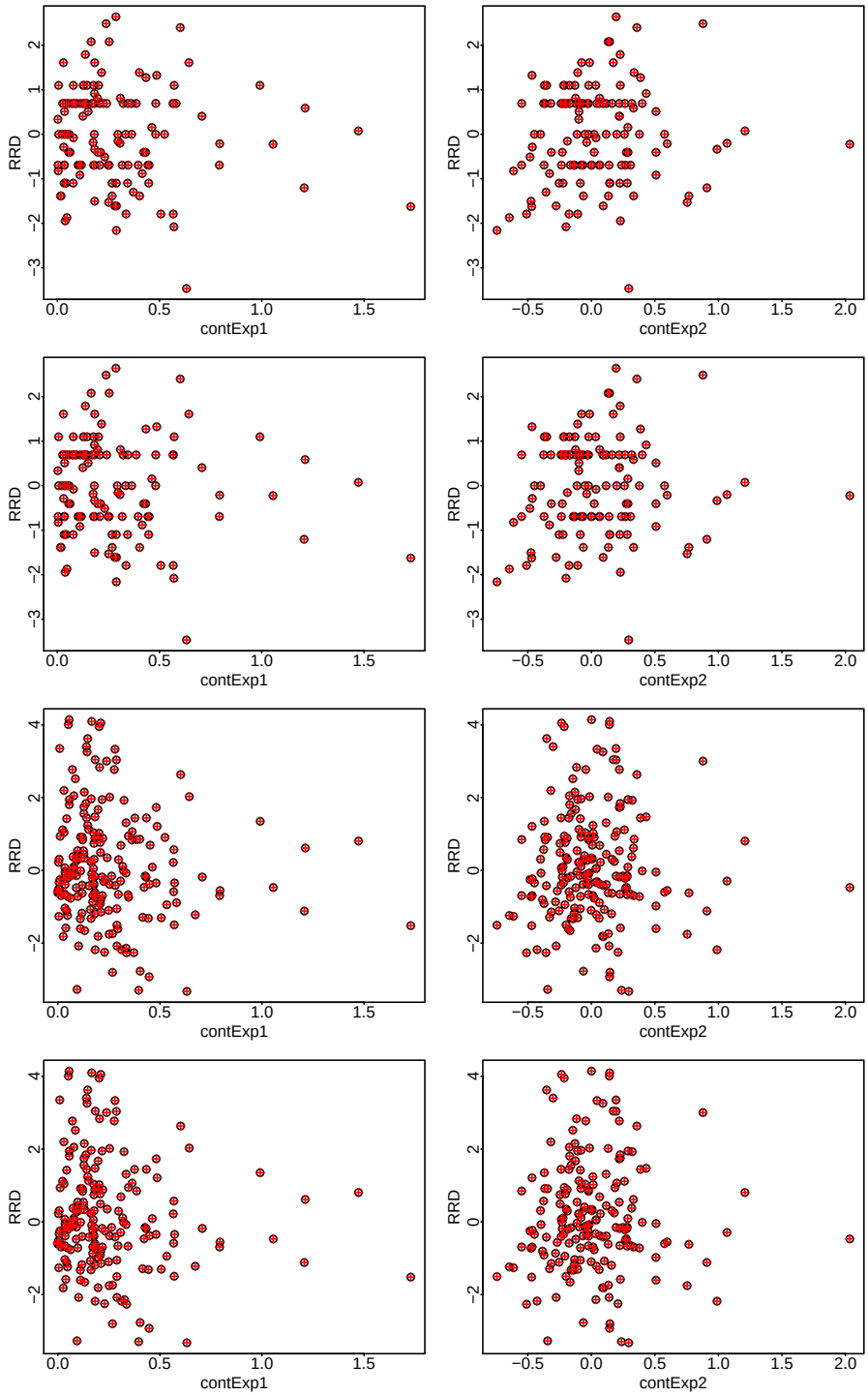

Figure 4: Overplotting of results from MacroCAIC (black) and `macrocaic()` (red) analyses using RRD and complete data.

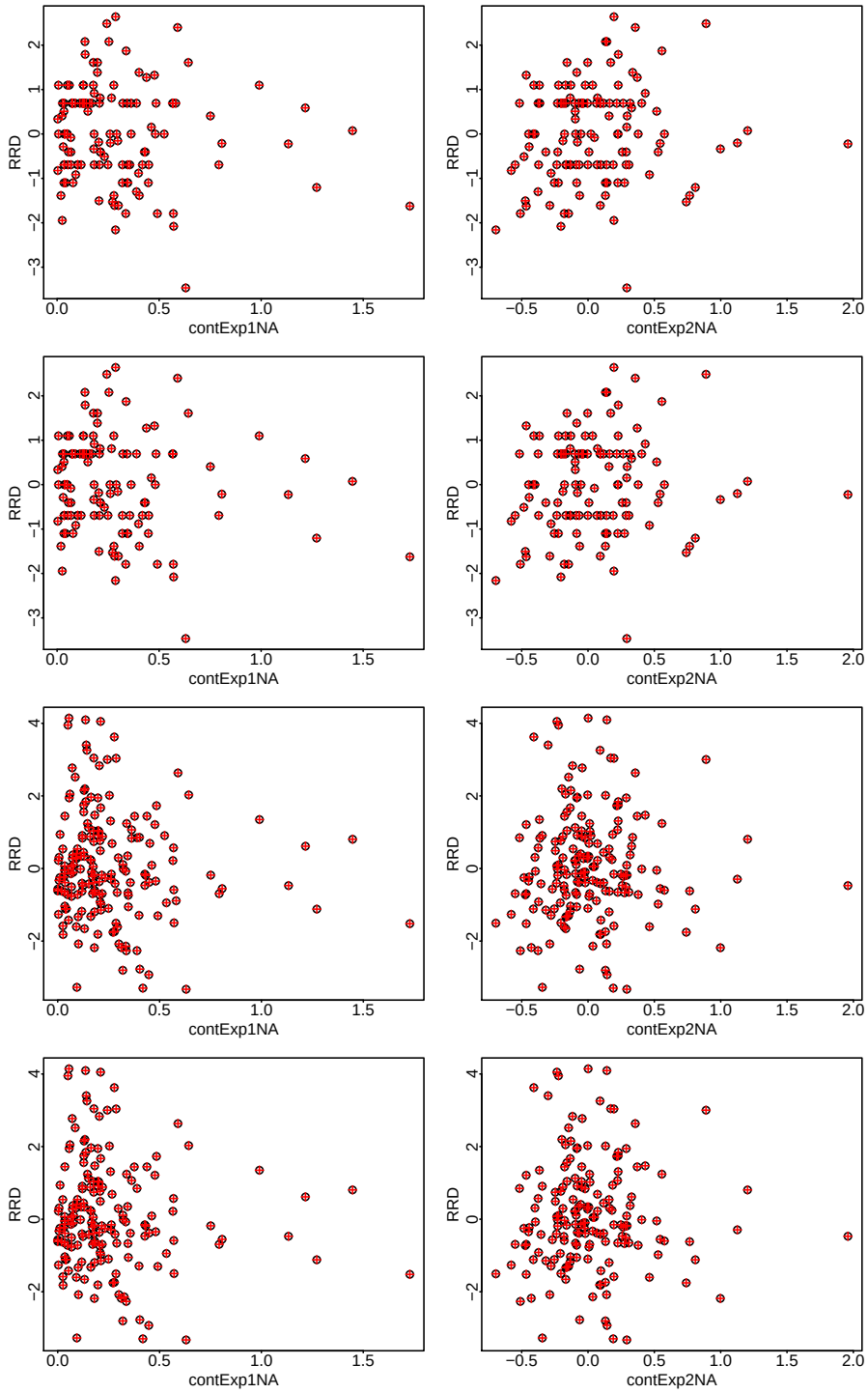

Figure 5: Overplotting of results from MacroCAIC (black) and `macrocaic()` (red) analyses using RRD and incomplete data.

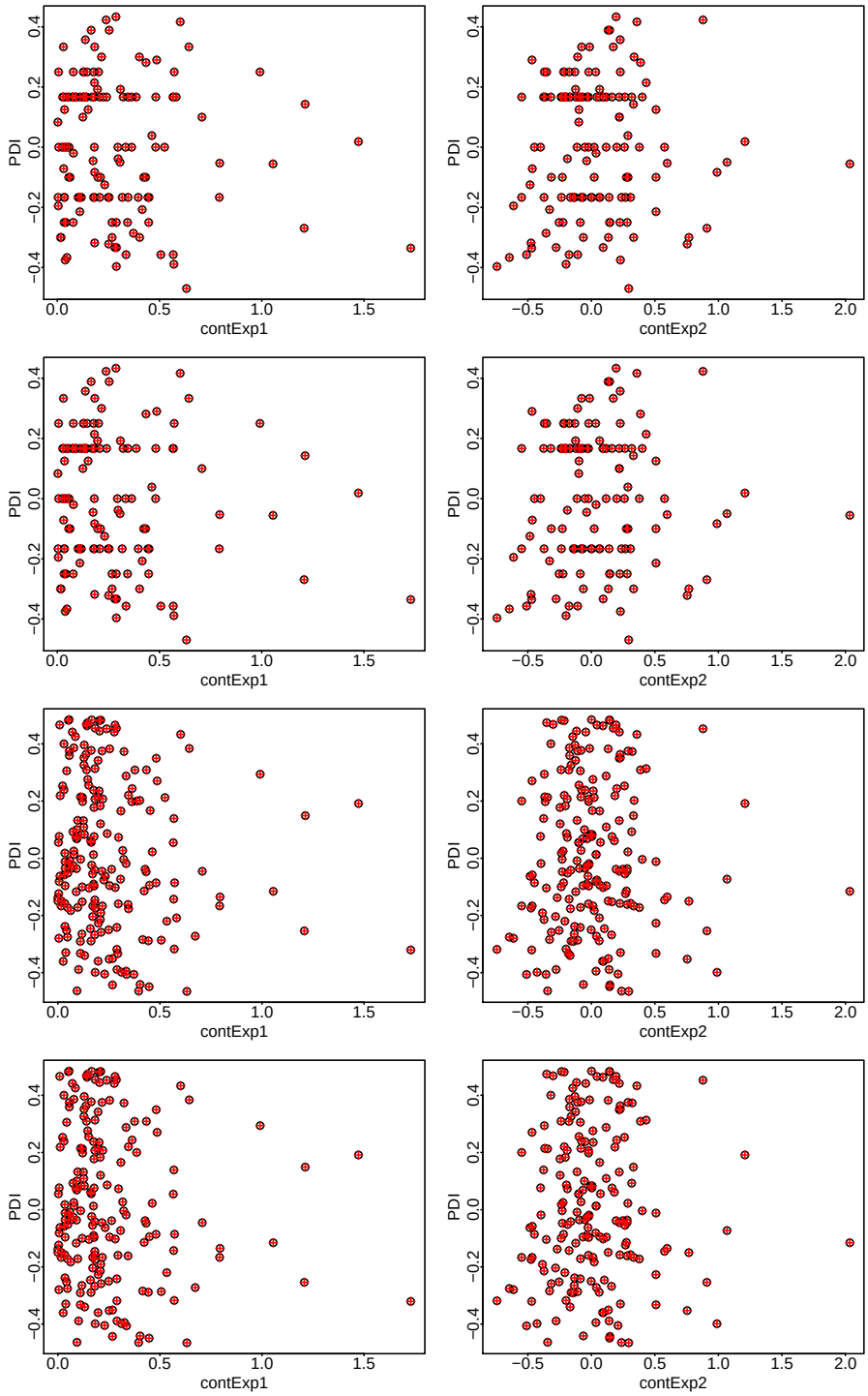

Figure 6: Overplotting of results from MacroCAIC (black) and `macrocaic()` (red) analyses using PDI and complete data.

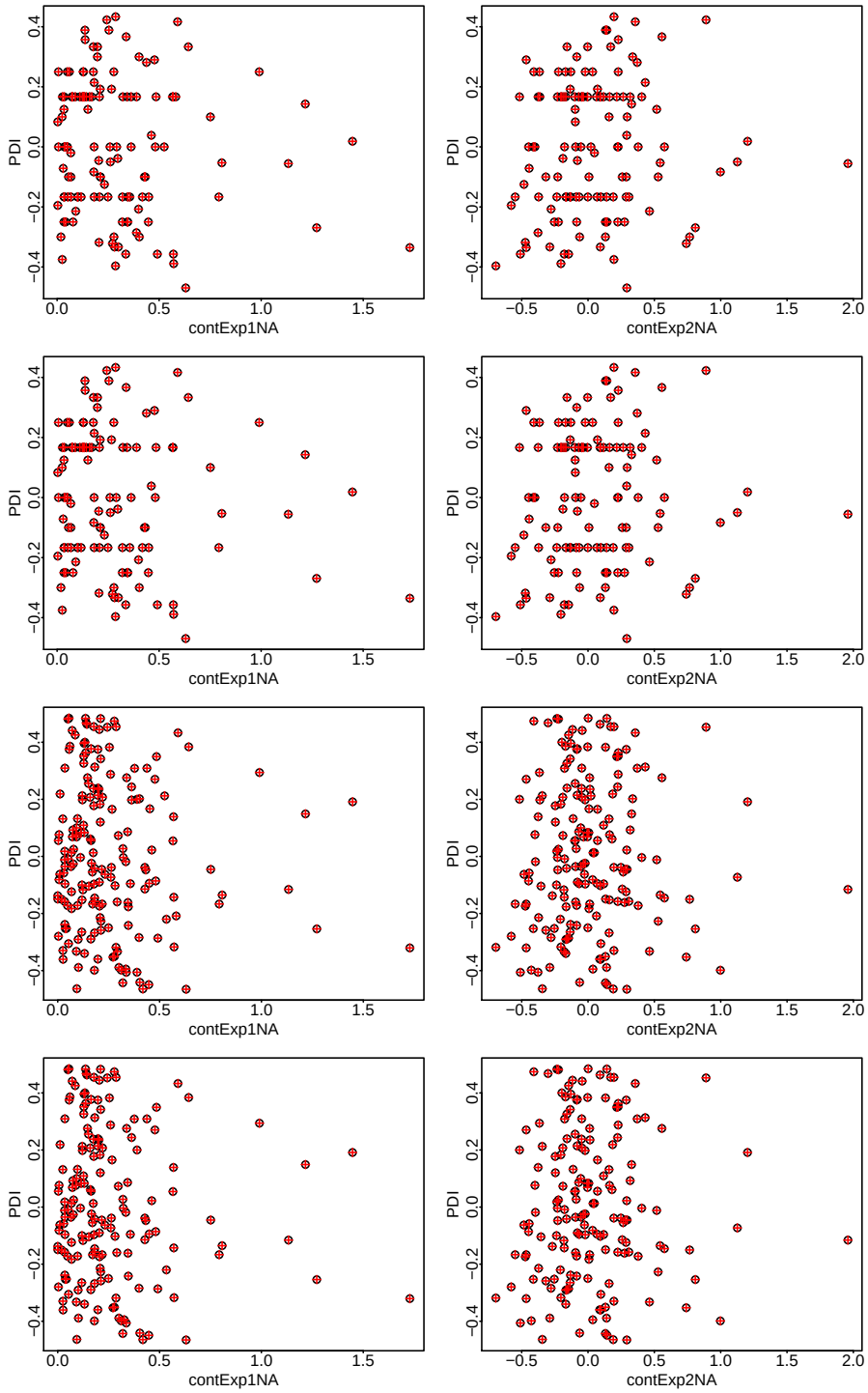

Figure 7: Overplotting of results from MacroCAIC (black) and `macrocaic()` (red) analyses using PDI and incomplete data.

both the dichotomous and polytomous benchmark trees. The following code loads the benchmark outputs:

```
> readFuscoOut <- function(fname) {
+     distTab <- read.table(fname, skip = 3, nrows = 10, sep = "|")
+     distTab <- distTab[, -c(1, 6)]
+     colnames(distTab) <- c("imbalanceBin", "obsFreq", "corr", "corrFreq")
+     distSum <- as.list(scan(fname, skip = 15, n = 10, what = list(NULL,
+         1), sep = "=")[[2]])
+     names(distSum) <- c("nSpp", "nTips", "nInfNodes", "medianI",
+         "qdI")
+     nodes <- read.table(fname, skip = 24, sep = "|")
+     nodes <- nodes[, -c(1, 7)]
+     colnames(nodes) <- c("node", "nodeT", "nodeB", "nodeS", "nodeI")
+     return(list(distTab = distTab, distSum = distSum, nodes = nodes))
+ }
> FuscoDiSpp <- readFuscoOut("../../benchmarks/FUSCO_outputs/FuscoDiSpp.txt")
> FuscoDiTax <- readFuscoOut("../../benchmarks/FUSCO_outputs/FuscoDiTax.txt")
> FuscoPolySpp <- readFuscoOut("../../benchmarks/FUSCO_outputs/FuscoPolySpp.txt")
> FuscoPolyTax <- readFuscoOut("../../benchmarks/FUSCO_outputs/FuscoPolyTax.txt")
```

The following code duplicates these analyses using the `fusco.test` function. Summary statistics comparing the two implementations are then show in Table 4. Note that the median and quartile deviation of I in the summary information produced by 'IMB_CALC' and included in the benchmark files are *not directly reproducible* using `fusco.test`. However, the values presented here are medians and quartile deviations calculated in R directly from the node table generated by 'IMB_CALC'. The corrected nodal distributions of the original I statistic (Fusco and Cronk, 1995), calculated using the `plot` method, are shown plotted over the values from 'IMB_CALC' in Fig. 8.

```
> fstest.DiSpp <- fusco.test(BENCH)
> fstest.DiTax <- fusco.test(BENCH, BenchData, rich = SppRich, names.col = node)
> fstest.PolySpp <- fusco.test(BENCHPoly)
> fstest.PolyTax <- fusco.test(BENCHPoly, BenchData, rich = SppRich,
+     names.col = node)
> fstest.DiSpp.Hist <- plot(fstest.DiSpp, I.prime = FALSE, plot = FALSE)
> fstest.DiTax.Hist <- plot(fstest.DiTax, I.prime = FALSE, plot = FALSE)
> fstest.PolySpp.Hist <- plot(fstest.PolySpp, I.prime = FALSE, plot = FALSE)
> fstest.PolyTax.Hist <- plot(fstest.PolyTax, I.prime = FALSE, plot = FALSE)
```

analysis	nNode	nTips	nInfNodes	medianI	qdI
FuscoDiSpp	200	200	96	0.541667	0.450790
fstest.DiSpp	200	200	96	0.541667	0.450790
FuscoDiTax	5000	200	199	0.425000	0.253035
fstest.DiTax	5000	200	199	0.425000	0.253035
FuscoPolySpp	200	200	93	0.500000	0.448276
fstest.PolySpp	200	200	93	0.500000	0.448276
FuscoPolyTax	5000	200	188	0.444444	0.259901
fstest.PolyTax	5000	200	188	0.444444	0.259901

Table 4: Calculation of Fusco and Cronk's (1995) I statistic.

3.2 Testing against the extended implementation (I, I', I_w)in MeSA.

The original I imbalance statistic showed a bias related to node size, demonstrated and corrected by Purvis et al. (2002). This revised calculation using either weights (I_w) or a modification to the

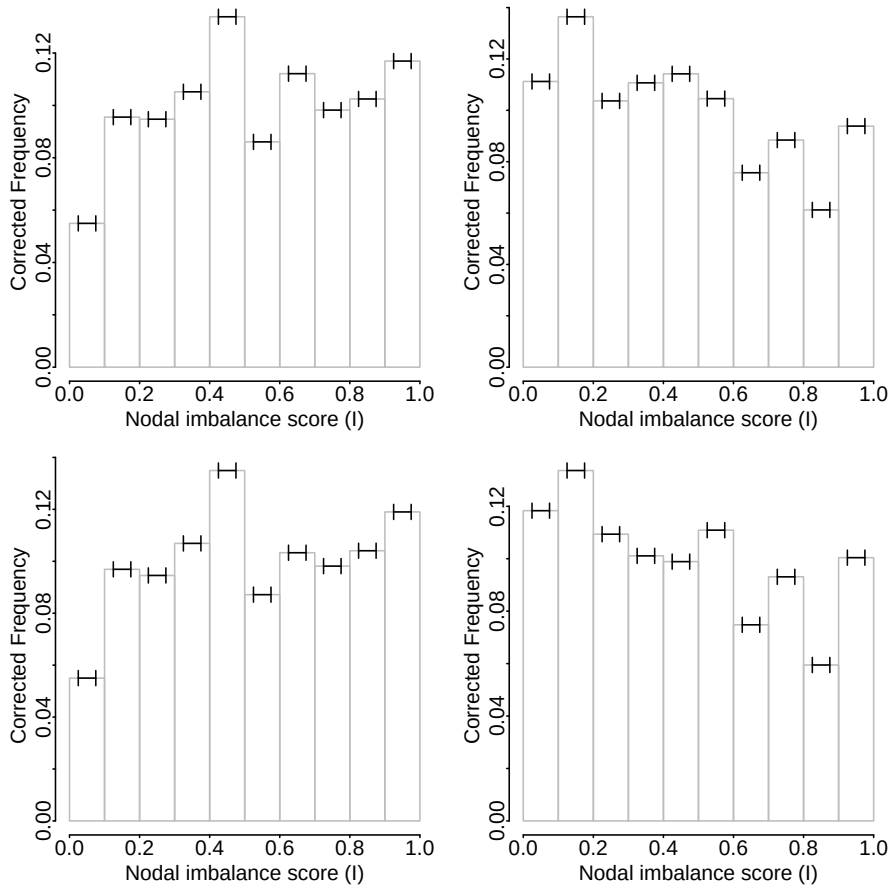

Figure 8: Histograms of nodal imbalance values from the program 'IMB_CALC' (grey bars) showing the corresponding values calculated using `fusco.test` in black.

calculation (I') was implemented in the program MeSA (Agapow, 2006). The following code loads output from MeSA v1.9.23 running under Mac OS 10.5.3, repeating the calculations in Purvis et al. (2002) of weights, I and I' on a genus-level tree of the Syrphidae.

```
> MeSA.I <- scan("../../benchmarks/MeSA_outputs/MeSA_FuscoI.txt",
+     sep = "\n", what = "character")
> MeSA.I <- strsplit(MeSA.I, split = "\t")
> MeSA.I <- lapply(MeSA.I, function(X) X[-1:-2])
> MeSA.I <- as.data.frame(MeSA.I, stringsAsFactors = FALSE)[, c(-5,
+     -7)]
> names(MeSA.I) <- c("clade", "I", "weight", "nodeSize", "Iprime")
> MeSA.I$I <- as.numeric(MeSA.I$I)
> MeSA.I$weight <- as.numeric(MeSA.I$weight)
> MeSA.I$nodeSize <- as.numeric(MeSA.I$nodeSize)
> MeSA.I$Iprime <- as.numeric(MeSA.I$Iprime)
```

The next code block replicates these calculations using `fusco.test` and displays the calculated mean values for both the original I and I' from both implementations.

```
> syrphTree <- read.nexus("../../benchmarks/syrphidae.nexus")
> syrphRich <- read.delim("../../benchmarks/syrphidae_tabbed.txt",
+     header = FALSE, col.names = c("genus", "nSpp"))
> fstest.Syrph <- fusco.test(syrphTree, dat = syrphRich, rich = nSpp,
+     names = genus)
> summary(fstest.Syrph)


Fusco test for phylogenetic imbalance

  Tree with 105 informative nodes and 204 tips.
  Tips are higher taxa containing 5330 species.
  95.0% confidence intervals randomised or simulated using 1000 replicates.

  Mean I prime: 0.629 [0.436,0.564]
  Median I: 0.727
  Quartile deviation in I: 0.291

        Wilcoxon signed rank test with continuity correction

data:  object$observed$I.prime
V = 3986, p-value = 0.0001197
alternative hypothesis: true location is not equal to 0.5


> mean(MeSA.I$Iprime)


[1] 0.6293965


> median(MeSA.I$I)


[1] 0.727405
```

References

Paul-Michael Agapow and Nick J B Isaac. Macrocaic: revealing correlates of species richness by comparative analysis. *Diversity and Distributions*, 8:41–43, 2002.

Joseph Felsenstein. Phylogenies and the comparative method. *American Naturalist*, 125:1–15, 1985.

Guiseppe Fusco and Quentin C B Cronk. A new method for evaluating the shape of large phylogenies. *Journal of Theoretical Biology*, 175:235–243, 1995.

Andy Purvis and Andy Rambaut. Comparative analysis by independent contrasts (caic): an apple macintosh application for analysing comparative data. *Computer Applications In the Biosciences*, 11:247–251, 1995.

Andy Purvis, Aris Katzourakis, and Paul-Michael Agapow. Evaluating phylogenetic tree shape: Two modifications to fusco & cronk's method. *Journal of Theoretical Biology*, 214:99–103, 2002. doi: DOI 10.1006/jtbi.2001.2443.