Reproducing the paper and its online supplement

Marc Jekel

2026-11-25

This vignette walks through the examples of Jekel, Regenwetter, Chen, and Line (in preparation), *“Wait, what are you saying, exactly?” A theoretical framework for codifying and evaluating verbal hypotheses about proportions*, and of its online supplement, one step at a time, with the code that produces each table and number. Every number quoted from the paper is also checked by the package’s tests (`tests/testthat/test-paper-*.R`).

The package does what “the mathematics fairy” does in the paper (it does not store the paper’s models; you enter them as constraint strings, as shown below) (https://modeling-for-everyone.eu/posts/fairy/):

In the paper	In the package
minimal description via order constraints (H-representation)	`fairy_h()`, `fairy_constraints()`
minimal description via extreme cases (V-representation)	`fairy_vertices()`, `fairy_words()`
parsimony (volume) and the largest possible Bayes factor	`fairy_volume()`, `fairy_max_bf()`, `fairy_info()`
overlap, nesting, mixture models	`fairy_intersect()`, `fairy_nested()`, `fairy_mixture()`, `fairy_compare()`
replication models (identical, joint, substitutable)	`fairy_replicate()` (with differing labs or items: `fairy_replicate_labs()`, `fairy_replicate_items()`)
input files for QTEST and multinomineq	`fairy_write_qtest()`, `fairy_write_multinomineq()`

This vignette shows the numbers from which the paper’s figures and tables are made, not the figures and tables themselves: the layout code (colours, symbols, PDF tables) is not needed for the results. To see how few commands reproduce every figure and table, including their PDFs, use the scripts on https://modeling-for-everyone.eu/posts/fairy/#reproduce (`Rscript run_all.R` runs all of them; some very large models are “not computed” by default and can be switched on at the top of the script: `list_all_vertices_model_10` in `Tbl_2_Replication.R`, `compute_direct` in `Tbl_E6_Replication.R`).

Part 1: The cringyness study

Verbal hypotheses become order constraints

Six probabilities describe the study. The probability that a randomly drawn child, teen, or adult cringes at an illogical joke or at a pun gives the order `p1, ..., p6`:

```r
lab <- c("illogical: children", "illogical: teens", "illogical: adults",
         "pun: children", "pun: teens", "pun: adults")
data.frame(parameter = paste0("p", 1:6), meaning = lab)
#>   parameter             meaning
#> 1        p1 illogical: children
#> 2        p2    illogical: teens
#> 3        p3   illogical: adults
#> 4        p4       pun: children
#> 5        p5          pun: teens
#> 6        p6         pun: adults
```

A model is a character string. Constraints are separated by `;`, chains such as `p1 > p2 > p3` are allowed, and `<` and `<=` mean the same thing (as in the paper, models are closed sets, so no software can tell `<` from `<=`). The package does not come with the paper's models: you enter them yourself. Here are Models 1 to 9 as constraint strings:

```r
strings <- c(
  M1 = "(p1+p4)/2 < (p2+p5)/2 < (p3+p6)/2",
  M2 = "p1 < p2 < p3; p4 < p5 < p6",
  M3 = "p1 > p2 > p3; p4 < p5 < p6; p1+p2+p3 < p4+p5+p6",
  M4 = "p1 > p2 > p3; p4 < p5 < p6; p1 < p4; p2 < p5; p3 < p6",
  M5 = "p1 = p2 = p3 = p4 = p5 = p6",
  M6 = "p1 > p2 > p3; p4 < p5 < p6; p1+p2+p3 = p4+p5+p6",
  M7 = "p1 > p2 > p3; p4 < p5 < p6; (p1+p2+p3)/3 = (p4+p5+p6)/3",
  M8 = "{p1,p2,p3,p4,p5,p6} = {p1,p2,p3,p4,p5,p6}",
  M9 = "p1 < p2 < p3; p4 > p5 > p6; p1 > p4; p2 > p5; p3 > p6"
)
```

- Model 1 is the average of the two joke types per age group, increasing with age.
- Models 7 and 8 use *approximate* equality: `approx_tol = 0.05` turns every equality into "within five percentage points"; it is passed to `fairy_h()` for these two models below.
- Model 10, the mixture of Models 4 and 9, is not a string: it is built with `fairy_mixture()` below.

```r
m <- lapply(strings, fairy_h, n_p = 6)                          # H-representations of Models 1 to 9
m$M7 <- fairy_h(strings[["M7"]], n_p = 6, approx_tol = 0.05)     # approximate equalities
m$M8 <- fairy_h(strings[["M8"]], n_p = 6, approx_tol = 0.05)
m$M10 <- fairy_mixture(m$M4, m$M9)                               # the mixture of Models 4 and 9
m <- m[paste0("M", 1:10)]
names(m)
#>  [1] "M1"  "M2"  "M3"  "M4"  "M5"  "M6"  "M7"  "M8"  "M9"  "M10"
```

Minimal descriptions via order constraints (Figure 2)

`fairy_h()` returns the non-redundant H-representation. For Model 4 the paper reports five inequalities among pairs of parameters and two range constraints, `0 <= p3 <= p2 <= p1 <= p4 <= p5 <= p6 <= 1`:

```
fairy_constraints(m$M4)
#> [1] "- p1 + p2 <= 0" "- p2 + p3 <= 0" "p4 - p5 <= 0"   "p5 - p6 <= 0"   "p1 - p4 <= 0"
#> [6] "p6 <= 1"        "- p3 <= 0"
```

The first line is the paper's `-1 x p1 + 1 x p2 <= 0`. The same constraints in the machine-readable format of QTEST are written by `fairy_write_qtest()` (and for the R package multinomineq by `fairy_write_multinomineq()`):

```
fairy_write_qtest(strings["M4"], n_p = 6, file = "model4.tar")
```

The no-effect model (Model 5) is two inequalities and five equalities:

```
fairy_info(m$M5)[, c("dim", "inequalities", "equalities", "vertices")]
#>   dim inequalities equalities vertices
#> 1   1           2          5        2
```

The mixture model (Model 10) and its 64 constraints

The mixture of Models 4 and 9 says: each person follows either Model 4 or Model 9, with an unknown weight. Geometrically, that is the convex hull of the two models, and `fairy_mixture()` computes its H-representation:

```
mix <- fairy_mixture(m$M4, m$M9)
fairy_info(mix, volume = FALSE)
#>   dim vertices inequalities equalities volume max_bf
#> 1   6       12           64          0     NA     NA
```

There are 12 vertices and 64 non-redundant constraints. Twelve of them are the range constraints `0 <= p_i <= 1`, which leaves the paper's 52 order constraints:

```
hn <- rcdd::q2d(mix)
single <- rowSums(hn[, -(1:2)] != 0) == 1
c(range = sum(single), other = sum(!single))
#> range other
#>    12    52
```

The first row of the paper's list of 52 constraints reads `p2 <= p1 + p3 <= 1 + p2`. The package lists them too:

```
head(fairy_constraints(mix)[!single], 6)
#> [1] "p1 - p3 - p6 <= 0"            "p2 - p3 - p6 <= 0"
#> [3] "p1 - p2 - p6 <= 0"            "- p3 + p4 - p6 <= 0"
#> [5] "- p1 + p2 - p3 + p4 - p6 <= 0" "- p2 + p4 - p6 <= 0"
```

The paper counts the "edge cases" of the vertex V1 of Figure 3 (`p1 = p4 = p5 = p6 = 1`, `p2 = p3 = 0`): it satisfies 26 of the 52 order constraints with equality, and 6 range constraints, 32 in all:

```r
v1 <- c(1, 0, 0, 1, 1, 1)
slack <- hn[, 2] + as.vector(hn[, -(1:2)] %*% v1)
c(order_constraints = sum(abs(slack[!single]) < 1e-9),
  range_constraints = sum(abs(slack[single]) < 1e-9),
  total = sum(abs(slack) < 1e-9))
#> order_constraints range_constraints             total
#>                26                 6                32
```

Minimal descriptions via extreme cases (Figures 3 and 4)

`fairy_words()` lists the vertices, writing a probability of 0 as “none” and of 1 as “all” (valid for a finite population). The row order of the underlying algorithm is arbitrary, so the rows are sorted. The labels are the six columns of the paper’s tables:

```r
fairy_words(m$M4, labels = lab)       # Figure 3, Model 4
#>    illogical: children illogical: teens illogical: adults pun: children pun: teens
#> V1                none             none              none          none       none
#> V2                none             none              none          none       none
#> V3                none             none              none          none        all
#> V4                none             none              none           all        all
#> V5                 all             none              none           all        all
#> V6                 all              all              none           all        all
#> V7                 all              all               all           all        all
#>    pun: adults
#> V1        none
#> V2         all
#> V3         all
#> V4         all
#> V5         all
#> V6         all
#> V7         all
```

Model 9 is the mirror image, `all` and `none` swapped:

```r
fairy_words(m$M9, labels = lab)
#>    illogical: children illogical: teens illogical: adults pun: children pun: teens
#> V1                none             none              none          none       none
#> V2                none             none               all          none       none
#> V3                none              all               all          none       none
#> V4                 all              all               all          none       none
#> V5                 all              all               all           all       none
#> V6                 all              all               all           all        all
#> V7                 all              all               all           all        all
#>    pun: adults
#> V1        none
#> V2        none
#> V3        none
#> V4        none
#> V5        none
```

```
#> V6        none
#> V7         all
```

Figure 4, the 12 vertices of Model 6 (equal marginal means), contains fractions:

```
fairy_words(m$M6, labels = lab)
#>     illogical: children illogical: teens illogical: adults pun: children pun: teens
#> V1                 none             none              none          none       none
#> V2                  1/3              1/3               1/3          none       none
#> V3                  1/2              1/2              none          none       none
#> V4                  2/3              2/3               2/3          none        all
#> V5                  all             none              none          none       none
#> V6                  all             none              none          none        1/2
#> V7                  all             none              none           1/3        1/3
#> V8                  all              1/2               1/2          none        all
#> V9                  all              all              none          none        all
#> V10                 all              all              none           1/2        1/2
#> V11                 all              all              none           2/3        2/3
#> V12                 all              all               all           all        all
#>     pun: adults
#> V1         none
#> V2          all
#> V3          all
#> V4          all
#> V5          all
#> V6          1/2
#> V7          1/3
#> V8          all
#> V9          all
#> V10         all
#> V11         2/3
#> V12         all
```

The mixture model has the 12 vertices that are the union of those of Models 4 and 9 (the two share the vertices “all, all, all, all, all, all” and “none, …, none”):

```
nrow(fairy_vertices(m$M10))
#> [1] 12
```

Out of the 2^6 = 64 corners of the unit cube, only 12 are allowed, so the mixture model is falsifiable.

Parsimony (Figure 5)

The parsimony of a model is its volume inside the unit cube of all conceivable families of Binomial parameters. The inverse of the volume is the largest Bayes factor the model can reach against the unconstrained model (uniform prior). Figure 5 of the paper has two rectangles, their overlap and a line segment:

```
I   <- fairy_h("p1 < .5; p2 > .25; p2 < .75", n_p = 2)
II  <- fairy_h("p1 > .3; p1 < .7; p2 > .5", n_p = 2)
III <- fairy_intersect(I, II)
IV  <- fairy_h("p1 = .5; p2 > .25; p2 < .75", n_p = 2)

rbind(I = fairy_info(I), II = fairy_info(II), III = fairy_info(III), IV = fairy_info(IV))
#>     dim vertices inequalities equalities volume max_bf
#> I     2        4            4          0   0.25      4
#> II    2        4            4          0   0.20      5
#> III   2        4            4          0   0.05     20
#> IV    1        2            2          1   0.00    Inf
```

Model I has area 0.25 (maximal Bayes factor 4), Model II 0.2 (5), their overlap Model III 0.05 (20). Model IV contains an equality, so it is a line segment with volume zero and **no** upper bound on the Bayes factor. The vertices of the overlap include two new ones that are not vertices of either rectangle:

```
fairy_vertices(III)
#>      p_{1} p_{2}
#> [1,]   0.3  0.50
#> [2,]   0.5  0.50
#> [3,]   0.5  0.75
#> [4,]   0.3  0.75
```

Model relationships (Table 1)

`fairy_compare()` returns the layout of Table 1: the diagonal gives each model's volume, entries above the diagonal the volume of the overlap of two models, entries below the diagonal tell whether the row model is nested in (subset sign) or contains (superset sign) the column model, and otherwise the dimension of their overlap. The last row is the largest possible Bayes factor against the unconstrained model.

```
cmp <- fairy_compare(m)
cmp
#>       M1     M2     M3     M4       M5       M6       M7       M8       M9       M10
#> M1    0.17   0.028  0.0028 3.47e-04 0        0        7.96e-04 2.99e-07 3.47e-04 0.011
#> M2    subset 0.028  0      0        0        0        0        4.99e-08 0        0.0028
#> M3    6D     4D     0.014  0.0014   0        0        0.0023   2.50e-08 0        0.0025
#> M4    6D     4D     subset 0.0014   0        0        4.83e-08 2.50e-09 0        0.0014
#> M5    subset subset subset subset   0        0        0        0        0        0
#> M6    5D     3D     subset 1D       superset 0        0        0        0        0
#> M7    6D     4D     6D     6D       superset superset 0.0046   4.99e-08 0        1.70e-04
#> M8    6D     6D     6D     6D       superset 5D       6D       1.80e-06 2.50e-09 1.69e-06
#> M9    6D     4D     1D     1D       superset 1D       2D       6D       0.0014   0.0014
#> M10   6D     6D     6D     superset superset 5D       6D       6D       superset 0.044
#> maxBF    6     36     72     720      Inf      Inf       220     556522     720     22.5
```

Individual entries are available as numbers, for example the largest Bayes factor of each model and the volume of the mixture model:

```
round(cmp$max_bf, 1)
#>       M1       M2       M3       M4       M5       M6       M7       M8       M9      M10
#>      6.0     36.0     72.0    720.0      Inf      Inf    219.7 556521.7    720.0     22.5
cmp$volume["M10", "M10"]
#> [1] 0.04444444
```

The paper reports 6, 36, 72, 720, infinity, infinity, about 200, about 10^6, 720, and about 23. The values for Models 7 and 8 depend on the approximate-equality tolerance; with a tolerance of 0.05 the exact volumes give about 220 and 560,000, which the paper rounds to 200 and 10^6.

Checks of single entries

Three relations that are easy to get wrong: Model 6 is not a subset of Model 1 (their overlap is 5-dimensional), Model 7 contains Model 6 (relaxing an equality can only add points), and the overlap of Models 3 and 10 is 6-dimensional.

```
c(M6_in_M1 = fairy_nested(m$M6, m$M1), M6_in_M7 = fairy_nested(m$M6, m$M7),
  M3_M10_dim = fairy_dim(fairy_intersect(m$M3, m$M10)))
#>   M6_in_M1   M6_in_M7 M3_M10_dim
#>          0          1          6
```

Replication models (Table 2)

With r = 3 labs, each with six probabilities, replication models live in 18 dimensions. `fairy_replicate()` builds the three versions of the paper:

- **identical**: every probability is the same in all labs, 12 equalities, dimension 6, volume zero.
- **joint**: each lab satisfies the model, labs are unrelated; the volume is the cube of the single-lab volume.
- **substitutable**: each constraint holds whichever lab's parameter is plugged in, for every term.

The columns of Table 2 are dimension, number of vertices, non-redundant inequalities, equalities, and the upper bound on the Bayes factor:

```
rep_table <- function(h, r = 3, bf_substitutable = NULL) {
  ident <- fairy_info(fairy_replicate(h, r, "identical"))
  joint <- fairy_info(fairy_replicate(h, r, "joint"))
  subst <- fairy_info(fairy_replicate(h, r, "substitutable"), volume = FALSE)
  subst$max_bf <- if (is.null(bf_substitutable)) NA_real_ else bf_substitutable
  out <- rbind(identical = ident, joint = joint, substitutable = subst)
  out[, c("dim", "vertices", "inequalities", "equalities", "max_bf")]
}
```

For Model 4 the substitutable bound has a closed form: the 18 values fall into six ordered groups of three, which happens with probability 3!^6 / 18!, so the bound is `18! / 3!^6`. (For high-dimensional polytopes of this kind `fairy_volume(method = "volesti")` gives a stochastic estimate if the volesti package is installed.)

```
exact_sub <- factorial(18) / factorial(3)^6
rep_table(m$M4, bf_substitutable = exact_sub)
```

```
#>              dim vertices inequalities equalities       max_bf
#> identical      6        7            7         12          Inf
#> joint         18      343           21          0    373248000
#> substitutable 18       43           51          0 137225088000
format(exact_sub, scientific = TRUE, digits = 3)      # 1.37e11 in the paper
#> [1] "1.37e+11"
format(720^3, scientific = TRUE, digits = 3)          # "720^3 = 4e8" in the paper
#> [1] "3.73e+08"
```

Model 9 gives the same numbers:

```
rep_table(m$M9, bf_substitutable = exact_sub)
#>              dim vertices inequalities equalities       max_bf
#> identical      6        7            7         12          Inf
#> joint         18      343           21          0    373248000
#> substitutable 18       43           51          0 137225088000
```

For the mixture model (Model 10) the joint model has 1,728 = 12^3 vertices and 3 x 64 = 192 inequalities, and a bound of about 23^3 = 10^4:

```
joint10 <- fairy_info(fairy_replicate(m$M10, 3, "joint"))
joint10[, c("dim", "vertices", "inequalities", "equalities", "max_bf")]
#>   dim vertices inequalities equalities   max_bf
#> 1  18     1728          192          0 11390.62
fairy_info(fairy_replicate(m$M10, 3, "identical"), volume = FALSE)[, 1:4]
#>   dim vertices inequalities equalities
#> 1   6       12           64         12
```

The substitutable version of Model 10 has 4,032 constraints. Removing redundant constraints in 18 dimensions takes several minutes, so the vignette shows the constraint count before that step (`reduce = FALSE`), which already equals the paper's number of non-redundant inequalities:

```
nrow(fairy_replicate(m$M10, 3, "substitutable", reduce = FALSE))
#> [1] 4032
```

The remaining cells of this row, the 46,739 vertices and the upper bound on the Bayes factor, take very long (many hours) in 18 dimensions: `fairy_info(fairy_replicate(m$M10, 3, "substitutable"), volume = FALSE)`. The script `Tbl_2_Replication.R` reads "not computed" for them unless its switch `list_all_vertices_model_10` is set to `TRUE`.

Replication analysis (Table E6, Online Supplement E)

Table E6 fits the replication models to the three hypothetical data sets (three labs, 18 probabilities). The fairy hands the model to multinomineq (`fairy_analyze_multinomineq()`), which returns the Bayesian p-value (`ppp`) and the Bayes factor against the unconstrained model (`bf_0u`). Here for Model 10 (the mixture of Models 4 and 9); the other models work the same way.

```
labs <- list(c(60, 64, 68, 60, 50, 38), c(150, 160, 170, 150, 125, 95), c(44, 40, 36, 50, 60,
        70))
n    <- c(100, 250, 100)                        # observations per cell in the three labs
dat  <- data.frame(probability = paste0("p", 1:18), n = rep(n, each = 6), frequency =
        unlist(labs))
```

```
settings <- list(bf_samples = 1e6, seed = 2026)
# identical replication model: six probabilities shared by all labs
id <- fairy_analyze_multinomineq(list(model = fairy_replicate(m$M10, 3, "identical")), data =
        dat, n_p = 18, settings = settings)$results
c(Bp = id$ppp, BF = id$bf_0u)
#>           Bp           BF
#> 0.000000e+00 2.478926e-10

# joint replication model: the labs are independent, so its Bayes factor is the product of
        the three single-lab Bayes factors
lab_bf <- sapply(1:3, function(k) {
  d <- dat[(6 * k - 5):(6 * k), ]; d$probability <- paste0("p", 1:6)
  fairy_analyze_multinomineq(list(model = m$M10), data = d, n_p = 6, settings =
        settings)$results$bf_0u
})
prod(lab_bf)
#> [1] 11381.45
```

The Bayesian p-value of the joint model and both values of the substitutable model need the 18-dimensional posterior. This is fast for Model 4 but takes many hours for the largest models, so the script `Tbl_E6_Replication.R` reads “not computed” for these cells unless `compute_direct` is set to `TRUE`. (Model 6 of Tables E3 and E5 has an exact equality, which multinomineq approximates by a thin slab around it, so its values are estimates.)

Part 2: The Heuristic-Systematic Model (online supplement)

Here are the theoretical predictions of the supplement, again entered as constraint strings. In the 2 x 2 design the four probabilities are `PbW, PbS, PsW, PsS` (`p1` to `p4`): big or small gap, weak or strong argument. For the cue-congruence models the six probabilities `PiW, PnW, PcW, PiS, PnS, PcS` are `p1` to `p6`.

```
hsm <- lapply(c(
  pred1    = "p1 < p2; p3 < p4",                                  # stronger arguments never
        hurt
  pred2    = "p1 < p3; p2 < p4",                                  # a smaller gap never hurts
  pred3    = "p1 < p2 < p4; p1 < p3 < p4",                        # both
  pred6    = "p1 < p2 < p3; p4 < p5 < p6; p7 < p8 < p9; p1 < p4 < p7; p2 < p5 < p8; p3 < p6 <
        p9",
  override = "p6 > p3 > p2 = p1; p6 > p4 = p5 > p2",
  tradeoff = "p6 > p3 > p2 > p1; p6 > p5 > p4 > p1; p5 > p2",
  ttb      = "p1 = 0; p4 = 0; p2 = 1; p5 = 1; p3 = 1; p6 = 1"    # take the best
), fairy_h)
hsm$override_ttb_mixture <- fairy_mixture(hsm$override, hsm$ttb)
lab4 <- c("big, weak", "big, strong", "small, weak", "small, strong")
```

Prediction 1 says stronger arguments never reduce the share who act; prediction 2 says a smaller gap never does. Each has nine extreme cases (Tables F1 and F2) and takes up a quarter of the cube:

```
fairy_words(hsm$pred1, labels = lab4)    # Table F1
#>    big, weak big, strong small, weak small, strong
#> V1      none        none        none          none
#> V2      none        none        none           all
#> V3      none        none         all           all
#> V4      none         all        none          none
#> V5      none         all        none           all
#> V6      none         all         all           all
#> V7       all         all        none          none
#> V8       all         all        none           all
#> V9       all         all         all           all
fairy_words(hsm$pred2, labels = lab4)    # Table F2
#>    big, weak big, strong small, weak small, strong
#> V1      none        none        none          none
#> V2      none        none        none           all
#> V3      none        none         all          none
#> V4      none        none         all           all
#> V5      none         all        none           all
#> V6      none         all         all           all
#> V7       all        none         all          none
#> V8       all        none         all           all
#> V9       all         all         all           all
sapply(hsm[c("pred1", "pred2")], fairy_max_bf)
#> pred1 pred2
#>     4     4
```

Their combination, Prediction 3, has six extreme cases (Table F3), a volume of 1/12 and a largest Bayes factor of 12:

```
fairy_words(hsm$pred3, labels = lab4)    # Table F3
#>    big, weak big, strong small, weak small, strong
#> V1      none        none        none          none
#> V2      none        none        none           all
#> V3      none        none         all           all
#> V4      none         all        none           all
#> V5      none         all         all           all
#> V6       all         all         all           all
fairy_info(hsm$pred3)
#>   dim vertices inequalities equalities     volume max_bf
#> 1   4        6            6          0 0.08333333     12
```

All six are vertices of both predictions, so intersecting the predictions adds no new extreme case:

```
fairy_nested(hsm$pred3, hsm$pred1) && fairy_nested(hsm$pred3, hsm$pred2)
#> [1] TRUE
```

Two populations

If Prediction 3 must hold jointly in two populations, the bound is `12^2 = 144`. If it holds with substitutable parameters, it is 420:

```
c(joint = fairy_max_bf(fairy_replicate(hsm$pred3, 2, "joint")),
  substitutable = fairy_max_bf(fairy_replicate(hsm$pred3, 2, "substitutable")))
#>         joint substitutable
#>           144           420
```

A 3-by-3 design

Three gap sizes and three argument strengths give nine probabilities. The twelve non-redundant order constraints of equation (6) plus two range constraints give a bound of about 8,600 (8,640), about 60 times the bound 144 of the two-population replication:

```
fairy_info(hsm$pred6)
#>   dim vertices inequalities equalities       volume max_bf
#> 1   9       20           14          0 0.0001157407   8640
fairy_max_bf(hsm$pred6) / 144
#> [1] 60
```

Cue congruence: Override, Trade-off, and take-the-best

The six probabilities are `PiW, PnW, PcW, PiS, PnS, PcS` (incongruent, no, or congruent cues; weak or strong arguments). The Override Model (equation 7) contains two equalities and so has volume zero. Table F4 (top) lists its six extreme cases; take-the-best (bottom) is a single point:

```
lab6 <- c("inc, weak", "no, weak", "con, weak", "inc, strong", "no, strong", "con, strong")
fairy_words(hsm$override, labels = lab6)       # Table F4, top
#>    inc, weak no, weak con, weak inc, strong no, strong con, strong
#> V1       none     none      none        none       none        none
#> V2       none     none      none        none       none         all
#> V3       none     none      none         all        all         all
#> V4       none     none       all        none       none         all
#> V5       none     none       all         all        all         all
#> V6        all      all       all         all        all         all
fairy_words(hsm$ttb, labels = lab6)            # Table F4, bottom
#>    inc, weak no, weak con, weak inc, strong no, strong con, strong
#> V1       none      all       all        none        all         all
fairy_info(hsm$override)[, c("dim", "vertices", "volume", "max_bf")]
#>   dim vertices volume max_bf
#> 1   4        6      0    Inf
```

The Trade-off Model (equation 8) has 10 vertices and a bound of 144. The Override Model is nested in it, but take-the-best violates the Override Model:

```r
fairy_info(hsm$tradeoff)[, c("dim", "vertices", "max_bf")]
#>   dim vertices max_bf
#> 1   6       10    144
c(override_in_tradeoff = fairy_nested(hsm$override, hsm$tradeoff),
  tradeoff_in_override = fairy_nested(hsm$tradeoff, hsm$override),
  ttb_in_override = fairy_nested(hsm$ttb, hsm$override))
#> override_in_tradeoff tradeoff_in_override      ttb_in_override
#>                 TRUE                FALSE                FALSE
```

Finally, the mixture of the Override Model and take-the-best adds the one vertex “none, all, all, none, all, all”. It keeps one equality (`PnW + PiS = PiW + PnS`), so its volume is zero and there is no upper bound on the Bayes factor. Both components are nested in it (equation 10):

```r
mixture <- hsm$override_ttb_mixture
fairy_constraints(mixture, labels = c("PiW", "PnW", "PcW", "PiS", "PnS", "PcS"))
#> [1] "- PiW + PnW + PiS - PnS = 0"  "PiW - PiS <= 0"
#> [3] "PnW - PcW <= 0"               "- PiW + PnW + PiS - PcS <= 0"
#> [5] "PcW - PcS <= 0"               "PcS <= 1"
#> [7] "- PiW <= 0"                   "PiW - PnW <= 0"
c(override_nested = fairy_nested(hsm$override, mixture), ttb_nested = fairy_nested(hsm$ttb,
        mixture))
#> override_nested      ttb_nested
#>            TRUE            TRUE
fairy_info(mixture)[, c("dim", "vertices", "inequalities", "equalities", "max_bf")]
#>   dim vertices inequalities equalities max_bf
#> 1   5        7           7          1    Inf
```