# Package: cash (via r-universe)

June 9, 2026

**Version** 1.0.2

**Title** Discrete Choice and Competitive Reactions: End-to-End Simulation

**Author** Jan H. R. Dressler [aut, cre, cph], Peter Kurz [ths], Winfried J. Steiner [ths]

**Maintainer** Jan H. R. Dressler <jhrd13@tu-clausthal.de>

**License** GPL (>= 3)

**Imports** reshape2, ggplot2, idefix (>= 1.1.0), DoE.base, evd, bayesm, coda, ggridges, foreach, doParallel, parallel, doRNG, Rcpp, arrangements, grDevices, stats, utils

**LinkingTo** Rcpp

**Encoding** UTF-8

**RoxygenNote** 8.0.0

**Description** Although discrete choice (choice-based conjoint) analysis has become a widely used technique for the elicitation of consumer preferences and hence a foundation for product design, to the best of our knowledge, there exists neither free and open-source nor commercial software that covers the game-theoretic simulation of competitive reactions among firms based on discrete choice models to improve decision making beyond traditional product (line) optimization. The package does not only provide functions to fill this gap but comprises an entire simulation pipeline including the upstream processes of discrete choice analysis itself. It ranges from preference generation, choice design, design assessment, error and response simulation, through hierarchical Bayesian estimation of mixed logit models as well as convergence and model assessment, to Nash equilibrium computation. Doing so, it partly draws from established packages concerned with discrete choice analysis. While its structure generally aims towards end-to-end simulation as well as simulation of competitive dynamics based on real data, all its key elements mentioned above may be of use independently of each other. A paper accompanying the package will be available and linked here shortly.

**NeedsCompilation** yes

**Repository** https://cran.r-universe.dev

**Date/Publication** 2026-06-09 08:44:06 UTC

**RemoteUrl** https://github.com/cran/cash

**RemoteRef** HEAD

**RemoteSha** 28c799adaf732f99dc9b0d531521c5a11434f3ba

# Contents



---

`B_prefgen` *Preference generation*

---

## Description

Simulates a multivariate normal preference structure following Wirth (2010a,b).

## Usage

```
B_prefgen(
  resps,
  lvls,
  het = TRUE,
  randomizecols = TRUE,
  sortfirstattdesc = TRUE,
  saveplot = FALSE,
  seed = 1,
  temp = TRUE
)
```

### Arguments

| | |
|---|---|
| resps | An integer defining the number of respondents. |
| lvls | A vector defining the number of features (length) and levels (values). |
| het | If TRUE, the utilities of each feature level are more heterogeneous across respondents. |
| randomizecols | If TRUE, the generated column vectors containing the utilities for all respondents for each feature level are randomized. |
| sortfirstattdesc | If TRUE, the utilities of the first feature are sorted descendingly (taking the reference level (first level) utility of zero into account). |
| saveplot | If TRUE, a very basic ggplot of the final densities is saved. |
| seed | Addresses set.seed(). |
| temp | If FALSE, output files are not saved into the session's temporary but the current working directory. |

### Value

Automatically stores output in .rds files to be re-imported by subsequent functions or via readRDS().

### References


- Dressler JHR, Kurz P, Steiner WJ (2025). "Computing Nash Equilibria for Product Design based on Hierarchical Bayesian Mixed Logit Models." *arXiv preprint (Econometrics)*.
- Wirth R (2010a). *Best-Worst Choice-Based Conjoint-Analyse. Eine Neue Variante der Wahlbasierten Conjoint-Analyse*. Tectum.
- Wirth R (2010b). "HB-CBC, HB-Best-Worst-CBC or No HB At All?" In *Proceedings of the 15th Sawtooth Software Conference*, pp. 321–356.


### Examples

```
B_prefgen(resps=10,lvls=c(5,5))
```

---

## C_choicedes *Choice design*

---

### Description

Creates choice designs using the R package *idefix* or prepares external designs.

## Usage

```
C_choicedes(
  lvls = "./B_levels.rds",
  setstraining,
  altstraining,
  setstest = setstraining,
  altstest = altstraining,
  prior = "./B_simulatedindividualbetas.rds",
  zeroprior = TRUE,
  bayesian = FALSE,
  createdesign = TRUE,
  algorithm = c("modifiedfedorov", "coordinateexchange"),
  trace = FALSE,
  startdesignstotest = 12,
  orthoarrayascandmodfed = FALSE,
  externaltrainingdesign,
  externaltestdesign,
  codingofexternal = c("dummy", "none"),
  efficiencyofexternaltraining = "",
  efficiencyofexternaltest = "",
  csvseparator = ";",
  seed = 1,
  temp = TRUE
)
```

## Arguments

| | |
|---|---|
| `lvls` | Defines the number of features and levels, equivalent to `B_`, but here through the corresponding .rds file coming from `B_`. |
| `setstraining` | An integer defining the number of choice sets (used for estimation). |
| `altstraining` | An integer defining the number of alternatives per choice set (used for estimation). |
| `setstest` | See `setstraining`, but for predictive testing. |
| `altstest` | See `altstraining`, but for predictive testing. |
| `prior` | A Bayesian prior .rds file, i.e., a preference matrix as generated by `B_`. |
| `zeroprior` | If `TRUE`, the entries of `prior` are set to zero. |
| `bayesian` | If `FALSE`, column means of `prior` are taken as (non-Bayesian) prior. |
| `createdesign` | If `FALSE`, external designs can be imported for preparation. |
| `algorithm` | Algorithm to generate choice designs by maximizing D-efficiency. Can either be `"modifiedfedorov"` or `"coordinateexchange"`. Default is `"modifiedfedorov"`. |
| `trace` | If `TRUE`, tracing of the algorithm is enabled. |
| `startdesignstotest` | An integer defining the number of random start designs to test. |
| `orthoarrayascandmodfed` | If `TRUE`, an orthogonal array is employed as the candidate set instead of the full factorial in case of the modified Fedorov algorithm. |

`externaltrainingdesign`
: Path to external design .csv file (used for estimation) with each row being an alternative (long-format), row names as first column and column names as first row.

`externaltestdesign`
: See `externaltrainingdesign`, but for predictive testing.

`codingofexternal`
: Choose `"dummy"` if external designs are dummy-coded and `"none"` to convert the designs to dummy-coding (first level becomes the reference category) if no coding scheme has been applied (i.e., each column corresponds to a feature and the entries are integers representing the levels). Default is `"dummy"`.

`efficiencyofexternaltraining`
: May be used to store efficiency measures of the external design (as character).

`efficiencyofexternaltest`
: See `efficiencyofexternaltraining`.

`csvseparator`
: Value separator for external design .csv files.

`seed`
: Addresses `set.seed()`.

`temp`
: If `FALSE`, output files are not saved into the session's temporary but the current working directory.

## Value

Automatically stores output in .rds files to be re-imported by subsequent functions or via `readRDS()`.

## References


- Dressler JHR, Kurz P, Steiner WJ (2025). "Computing Nash Equilibria for Product Design based on Hierarchical Bayesian Mixed Logit Models." *arXiv preprint (Econometrics)*.
- Traets F, Sanchez DG, Vandebroek M (2020). "Generating Optimal Designs for Discrete Choice Experiments in R: The idefix Package." *Journal of Statistical Software*, 96(3), 1–41.


## Examples

```
oldwd=getwd()
setwd(tempdir())
extrain=matrix(c(11,5,2,12,1,3,13,3,5,14,4,4,15,2,1,
                 21,2,5,22,4,3,23,5,1,24,1,2,25,3,4,
                 31,5,3,32,4,5,33,1,2,34,3,4,35,2,1,
                 41,5,2,42,4,1,43,2,3,44,3,5,45,1,4,
                 51,3,1,52,5,3,53,2,4,54,4,5,55,1,2,
                 61,2,5,62,5,2,63,1,1,64,4,4,65,3,3,
                 71,2,2,72,5,3,73,1,4,74,3,1,75,4,5,
                 81,1,2,82,2,3,83,3,4,84,5,5,85,4,1,
                 91,4,5,92,2,2,93,5,4,94,3,1,95,1,3,
                 101,2,3,102,5,5,103,4,2,104,1,4,105,3,1,
                 111,2,4,112,5,1,113,1,5,114,4,3,115,3,2,
                 121,5,4,122,3,5,123,4,2,124,2,1,125,1,3,
                 131,4,2,132,5,5,133,1,1,134,2,4,135,3,3,
                 141,3,2,142,5,1,143,1,5,144,2,3,145,4,4,
```

```
                151,2,5,152,4,3,153,5,4,154,3,2,155,1,1),
              nrow=75,ncol=3,byrow=TRUE)
  write.csv2(extrain,"extrain.csv",row.names=FALSE)
extest=matrix(c(11,1,1,12,4,2,13,5,3,14,2,5,15,3,4,
                21,1,5,22,3,3,23,5,1,24,2,4,25,4,2,
                31,2,1,32,5,5,33,3,4,34,4,3,35,1,2,
                41,1,1,42,3,3,43,5,5,44,2,2,45,4,4,
                51,5,4,52,1,3,53,3,1,54,4,2,55,2,5),
              nrow=25,ncol=3,byrow=TRUE)
  write.csv2(extest,"extest.csv",row.names=FALSE)
C_choicedes(setstraining=15,altstraining=5,setstest=5,createdesign=FALSE,
  externaltrainingdesign="./extrain.csv",externaltestdesign="./extest.csv",
  codingofexternal="none")
setwd(oldwd)
```

---

`D_responsesim` *Error and response simulation*

---

## Description

Median relative Gumbel error tuning procedure to add noise to the individual deterministic portion of utility across alternatives and choice sets, and simulation of choices based on the resulting total utilities, and/or preparation of data for model estimation.

## Usage

```
D_responsesim(
  simulatedindividualbetas = "./B_simulatedindividualbetas.rds",
  trainingdesign = "./C_trainingdesign.rds",
  testdesign = "./C_testdesign.rds",
  MRGEtarget,
  learningrate = 0.5,
  stoppingcriterion = 10^(-5),
  maxiterations = 10^4,
  transform = TRUE,
  simulate = TRUE,
  seed = 1,
  temp = TRUE
)
```

## Arguments

| | |
|---|---|
| `simulatedindividualbetas` | Path to .rds file with synthetic utilities (as generated by `B_`). |
| `trainingdesign` | Path to .rds file with choice design for estimation (as generated by `C_`). |
| `testdesign` | Path to .rds file with choice design for predictive testing (as generated by `C_`). |
| `MRGEtarget` | Percentage indicating the desired magnitude of the Median relative Gumbel error (MRGE). |

| | |
|---|---|
| learningrate | Learning rate for the MRGE tuning. |
| stoppingcriterion | Tolerance for the MRGE tuning. |
| maxiterations | Maximum iterations for the MRGE tuning. |
| transform | If TRUE, choice design and responses are stored in an additional way such that they can be processed by the R package *bayesm* used for model estimation. |
| simulate | If FALSE, error and response simulation can be skipped to import and prepare external (experimental or differently simulated) data for model estimation. |
| seed | Addresses set.seed(). |
| temp | If FALSE, output files are not saved into the session's temporary but the current working directory. |

## Value

Automatically stores output in .rds files to be re-imported by subsequent functions or via readRDS().

## References


- Dressler JHR, Kurz P, Steiner WJ (2025). "Computing Nash Equilibria for Product Design based on Hierarchical Bayesian Mixed Logit Models." *arXiv preprint (Econometrics)*.


## Examples

```
D_responsesim(MRGEtarget=0.3)
```

---

# E_hbmxlest *Hierarchical Bayesian estimation of mixed logit models*

---

## Description

Hierarchical Bayesian estimation of mixed logit model employing the hybrid Gibbs sampler with a random walk Metropolis Hastings step implemented by the R package *bayesm*.

## Usage

```
E_hbmxlest(
  trainingdesign = "./C_trainingdesign.rds",
  bayesmdata = "./D_bayesmdatatraining.rds",
  mcmciterations,
  nthiterationtoprint = 5000,
  moncomponents = 1,
  nthdrawtokeep = 1,
  signresfirstatt = c("none", "negative", "positive"),
  compresslist = TRUE,
  seed = 1,
  temp = TRUE
)
```

### Arguments

| | |
|---|---|
| `trainingdesign` | Path to .rds file with choice design for estimation (as generated by `C_`). |
| `bayesmdata` | Path to .rds file with choice designs incl. responses prepared for *bayesm* (as generated by `D_`). |
| `mcmciterations` | An integer defining the length of the Markov chain. |
| `nthiterationtoprint` | An integer defining the progress update frequency of the sampler. |
| `moncomponents` | An integer defining the mixture of normals components. To be left at default for now. |
| `nthdrawtokeep` | An integer defining pre-thinning of the Markov chain. |
| `signresfirstatt` | For sign-restricting the draws of the first feature. Can either be `"none"`, `"negative"` or `"positive"`. Default is `"none"`. |
| `compresslist` | If `TRUE`, size of the output is reduced. |
| `seed` | Addresses `set.seed()`. |
| `temp` | If `FALSE`, output files are not saved into the session's temporary but the current working directory. |

### Value

Automatically stores output in .rds files to be re-imported by subsequent functions or via `readRDS()`.

### References


- Dressler JHR, Kurz P, Steiner WJ (2025). "Computing Nash Equilibria for Product Design based on Hierarchical Bayesian Mixed Logit Models." *arXiv preprint (Econometrics)*.
- Rossi PE (2025). *bayesm: Bayesian Inference for Marketing/ Micro-Econometrics*. R Package.
- Rossi PE, Allenby GM, McCulloch R (2005). *Bayesian Statistics and Marketing*. John Wiley & Sons.


### Examples

```
E_hbmxlest(mcmciterations=1000)
```

---

## `F_designeval` *Design assessment*

---

### Description

Computes measures for choice design evaluation.

## Usage

```
F_designeval(
  trainingdesign = "./C_trainingdesign.rds",
  testdesign = "./C_testdesign.rds",
  burninpercentage = 0.9,
  mcmciterations,
  nthiterationtoprint = 5000,
  moncomponents = 1,
  nthdrawtokeep = 1,
  recoverycheckfortestdesign = TRUE,
  seed = 1,
  temp = TRUE
)
```

## Arguments

| | |
|---|---|
| `trainingdesign` | Path to .rds file with choice design for estimation (as generated by `C_`). |
| `testdesign` | Path to .rds file with choice design for predictive testing (as generated by `C_`). |
| `burninpercentage` | Percentage defining burn-in iterations of the Markov chain to be discarded before calculating recovery measures. |
| `mcmciterations` | An integer defining the length of the Markov chain. |
| `nthiterationtoprint` | An integer defining the progress update frequency of the sampler. |
| `moncomponents` | An integer defining the mixture of normals components. To be left at default for now. |
| `nthdrawtokeep` | An integer defining pre-thinning of the Markov chain. |
| `recoverycheckfortestdesign` | If `FALSE`, model estimation for `testdesign` based on random responses is prevented. |
| `seed` | Addresses `set.seed()`. |
| `temp` | If `FALSE`, output files are not saved into the session's temporary but the current working directory. |

## Value

Automatically stores output in .rds files to be re-imported by subsequent functions or via `readRDS()`.

## References


- Dressler JHR, Kurz P, Steiner WJ (2025). "Computing Nash Equilibria for Product Design based on Hierarchical Bayesian Mixed Logit Models." *arXiv preprint (Econometrics)*.


## Examples

```
F_designeval(mcmciterations=1000)
```

G_modeleval *Convergence and model assessment*

## Description

Computes various measures for model evaluation and prepares final model.

## Usage

```
G_modeleval(
  experimental = FALSE,
  simulatedindividualbetas = "./B_simulatedindividualbetas.rds",
  trainingdesign = "./C_trainingdesign.rds",
  testdesign = "./C_testdesign.rds",
  trainingdesignwithsimuresponses = "./D_trainingdesignwithsimuresponses.rds",
  testdesignwithsimuresponses = "./D_testdesignwithsimuresponses.rds",
  bayesmdata = "./D_bayesmdatatraining.rds",
  model = "./E_model.rds",
  burninpercentage,
  chainlengthpsrf,
  secondmodelforgelman = TRUE,
  seedsecondmodel = 2,
  removereversals = TRUE,
  removepositives = TRUE,
  drawspostremoval,
  thinning,
  alphanormality = 0.05,
  crediblelevel = 0.95,
  saveplot = FALSE,
  show = 5,
  seed = 1,
  temp = TRUE
)
```

## Arguments

| | |
|---|---|
| experimental | If TRUE, an internal model based on external data or an external model can be evaluated. |
| simulatedindividualbetas | Path to .rds file with synthetic utilities (as generated by B_). |
| trainingdesign | Path to .rds file with choice design for estimation (as generated by C_). |
| testdesign | Path to .rds file with choice design for predictive testing (as generated by C_). |
| trainingdesignwithsimuresponses | Path to .rds file with choice designs incl. responses for estimation (as generated by D_). |

`testdesignwithsimuresponses`
: Path to .rds file with choice designs incl. responses for predictive testing (as generated by `D_`).

`bayesmdata`
: Path to .rds file with choice designs incl. responses prepared for *bayesm* (as generated by `D_`).

`model`
: Path to .rds file with model (as generated by `E_`).

`burninpercentage`
: Percentage defining burn-in iterations of the Markov chain to be discarded. If < 1, it is interpreted as a percentage, and if >= 1, it is interpreted as an absolute number of burn-in iterations to be discarded.

`chainlengthpsrf`
: An integer defining the number of iterations after burn-in to be used for computation of the (M)PSRF.

`secondmodelforgelman`
: If `FALSE`, simulation of an independent second chain for convergence assessment is skipped.

`seedsecondmodel`
: Seed for simulation of independent second chain.

`removereversals`
: If `TRUE`, draws (remaining after burn-in) that violate monotonicity of the first feature (incorrect order) are removed.

`removepositives`
: If `TRUE`, draws (remaining after burn-in) that violate monotonicity of the first feature (incorrect signs) are removed. Only considered if `removereversals = TRUE`.

`drawspostremoval`
: An integer defining the desired number of remaining draws after removal before thinning.

`thinning`
: An integer defining thinning of the remaining draws.

`alphanormality`
: Significance level for frequentist normality tests.

`crediblelevel`
: Probability coverage of the Bayesian credible intervals for the parameter recovery and predictive accuracy measures.

`saveplot`
: If `TRUE`, the last respondent's primary chain trace plots for each parameter as well as very basic visualizations of the final posterior means and draws are saved.

`show`
: An integer defining the number of draws/respondents (taken in equal distance from first to last) to show if `saveplot = TRUE`.

`seed`
: Addresses `set.seed()`.

`temp`
: If `FALSE`, output files are not saved into the session's temporary but the current working directory.

## Value

Automatically stores output in .rds files to be re-imported by subsequent functions or via `readRDS()`.

## References


- Dressler JHR, Kurz P, Steiner WJ (2025). "Computing Nash Equilibria for Product Design based on Hierarchical Bayesian Mixed Logit Models." *arXiv preprint (Econometrics)*.


## Examples

```
G_modeleval(burninpercentage=100,chainlengthpsrf=900,drawspostremoval=10,thinning=5)
```

---

`H_precomputeM` *Pre-computations for Nash competition*

---

## Description

Pre-computes product configurations, unit contributions, utilities, line configurations and competitive scenarios with total contributions.

## Usage

```
H_precomputeM(
  trainingdesign = "./C_trainingdesign.rds",
  estimatedindividualbetas = "./G_estimatedindividualbetasdraws.rds",
  pricecostmatrix,
  pricecostmatrixfull,
  basecostfix = 0,
  basecostvar = TRUE,
  choicerule = c("first", "logit"),
  ruletosolvefcutilitytie = c("splitting", "sampling"),
  prodsperline,
  competitors,
  addnullparameters = TRUE,
  progressupdatedelay = 120,
  compressmatrix = FALSE,
  seed = 1,
  temp = TRUE
)
```

## Arguments

| | |
|---|---|
| `trainingdesign` | Path to .rds file with choice design for estimation (as generated by `C_`). Only extracts `$resps`, `$betas` and `$lvls`. |
| `estimatedindividualbetas` | Path to .rds file with final preference array of draws, point estimates/posterior means (as generated by `G_`) or simulated/true utilities (as generated by `B_`). |
| `pricecostmatrix` | Path to .csv file with price and cost levels, comprising only the features that have been defined in `B_` as being part of the conjoint choice experiment. The .csv has |

to have row names as first column, column names as first row, as many rows as there are features + 1, as many columns as there are (max) levels + 1, and price as first feature. Undefined cells (if the number of levels differs across features) must be NA, and a zero indicates that the level has no cost.

`pricecostmatrixfull`
: See `pricecostmatrix`, but containing all features relevant to the contribution margin.

`basecostfix`
: Numeric value defining fixed base cost.

`basecostvar`
: If `TRUE`, levels of the features included in the full but not partial price cost matrix are randomly sampled to add their corresponding costs to the fixed base cost.

`choicerule`
: Choice rule to be applied. Can either be `"first"` or `"logit"`. Default is `"first"`.

`ruletosolvefcutilitytie`
: Tie-breaking strategy for the case of multiple first choices being present within a complete competitive scenario. Can either be `"splitting"` or `"sampling"`. Default is `"splitting"`.

`prodsperline`
: An integer defining the number of products per line.

`competitors`
: An integer defining the number of competitors.

`addnullparameters`
: If `FALSE`, expansion of the preference array by zeros for the reference categories is suppressed.

`progressupdatedelay`
: Numeric value defining update frequency of progress .txt file in minutes.

`compressmatrix`
: If `TRUE`, size of the output is reduced.

`seed`
: Addresses `set.seed()`.

`temp`
: If `FALSE`, output files are not saved into the session's temporary but the current working directory.

## Value

Automatically stores output in .rds files to be re-imported by subsequent functions or via `readRDS()`.

## References


- Dressler JHR, Kurz P, Steiner WJ (2025). "Computing Nash Equilibria for Product Design based on Hierarchical Bayesian Mixed Logit Models." *arXiv preprint (Econometrics)*.


## Examples

```
oldwd=getwd()
setwd(tempdir())
pcmpartial=matrix(c(299,599,899,1199,1499,25,30,33,44,54),
  nrow=2,ncol=5,byrow=TRUE)
  write.csv(pcmpartial,"pcmpartial.csv")
pcmfull=matrix(c(299,599,899,1199,1499,25,30,33,44,54,10,11,12,65,79),
  nrow=3,ncol=5,byrow=TRUE)
  write.csv(pcmfull,"pcmfull.csv")
```

```
H_precomputeM(pricecostmatrix="./pcmpartial.csv",
  pricecostmatrixfull="./pcmfull.csv",prodsperline=1,competitors=2)
setwd(oldwd)
```

---

`I_preoptandnashgame` *Pre-optimization and Nash competition*

---

## Description

Pre-computes optima of competitive scenarios matrix and uses them for fast simulation of the Nash games.

## Usage

```
I_preoptandnashgame(
  scenariosmatrix = "./H_scenariosmatrixfirstdraws.rds",
  scenariosinfo = "./H_scenariosinfofirstdraws.rds",
  products = "./H_products.rds",
  lines = "./H_lines.rds",
  maxrounds = 20,
  progressupdatedelay = 120,
  temp = TRUE
)
```

## Arguments

| | |
|---|---|
| `scenariosmatrix` | Path to .rds file with pre-computed competitive scenarios matrix (as generated by `H_`). |
| `scenariosinfo` | Path to .rds file with additional information generated by `H_`. |
| `products` | Path to .rds file with pre-computed product configurations (as generated by `H_`). |
| `lines` | Path to .rds file with pre-computed line configurations (as generated by `H_`). |
| `maxrounds` | Even integer defining upper limit of game rounds. |
| `progressupdatedelay` | Numeric value defining update frequency of progress .txt file in minutes. |
| `temp` | If `FALSE`, output files are not saved into the session's temporary but the current working directory. |

## Value

Automatically stores output in .rds files to be re-imported via `readRDS()`.

## References


- Dressler JHR, Kurz P, Steiner WJ (2025). "Computing Nash Equilibria for Product Design based on Hierarchical Bayesian Mixed Logit Models." *arXiv preprint (Econometrics)*.

## Examples

```
I_preoptandnashgame()
```

# Index