Package 'logitr'

June 14, 2021

Title Logit Models w/Preference & WTP Space Utility Parameterizations

Version 0.2.0

Description Estimation of multinomial (MNL) and mixed logit (MXL) models in R. Models can be estimated using ``Preference'' space or ``Willingness-to-pay'' (WTP) space utility parameterizations. Weighted models can also be estimated. An option is available to run a multi-start optimization loop with random starting points in each iteration, which is useful for non-convex problems like MXL models or models with WTP space utility parameterizations. The main optimization loop uses the 'nloptr' package to minimize the negative log-likelihood function. Additional functions are available for computing and comparing WTP from both preference space and WTP space models and for predicting expected choices and choice probabilities for a set (or multiple sets) of alternatives based on an estimated model. MXL models assume uncorrelated heterogeneity covariances and are estimated using maximum simulated likelihood based on the algorithms in Train (2009) ``Discrete Choice Methods with Simulation, 2nd Edition'' <doi:10.1017/CBO9780511805271>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

VignetteBuilder knitr

Depends R (>= 3.5.0)

Suggests dplyr, fastDummies, knitr, rmarkdown, here, ggplot2, testthat

Imports nloptr, stats, randtoolbox, MASS

URL https://github.com/jhelvy/logitr

BugReports https://github.com/jhelvy/logitr/issues

NeedsCompilation no

Author John Helveston [aut, cre, cph]
(<https://orcid.org/0000-0002-2657-9191>),
Connor Forsythe [aut]

Maintainer John Helveston <john.helveston@gmail.com>

Repository CRAN

Date/Publication 2021-06-14 13:50:17 UTC

R topics documented:

`cars_china` *Stated car choice observations by Chinese car buyers*

Description

Data from Helveston et al. (2015) containing 448 stated choice observations from Chinese car buyers and 384 stated choice observations from US car buyers. Conjoint surveys were fielded in 2012 in four major Chinese cities (Beijing, Shanghai, Shenzhen, and Chengdu), online in the US on Amazon Mechanical Turk, and in person at the Pittsburgh Auto show. Participants were asked to select a vehicle from a set of three alternatives. Each participant answered 15 choice questions.

Usage

```
data(cars_china)
```

Format

Variable	Description
`id`	individual identifiers
`obsnum`	identifier for unique choice observation
`choice`	dummy code for choice (`1` or `0`)
`hev`	dummy code for HEV vehicle type (`1` or `0`)
`phev10`	dummy code for PHEV vehicle type w/10 mile electric driving range (`1` or `0`)
`phev20`	dummy code for PHEV vehicle type w/20 mile electric driving range (`1` or `0`)
`phev40`	dummy code for PHEV vehicle type w/40 mile electric driving range (`1` or `0`)
`bev75`	dummy code for BEV vehicle type w/75 mile electric driving range (`1` or `0`)
`bev100`	dummy code for BEV vehicle type w/100 mile electric driving range (`1` or `0`)
`bev150`	dummy code for BEV vehicle type w/150 mile electric driving range (`1` or `0`)

phevFastcharge	dummy code for whether PHEV vehicle had fast charging capability (1 or 0)
bevFastcharge	dummy code for whether BEV vehicle had fast charging capability (1 or 0)
price	price of vehicle ($USD)
opCost	operating cost of vehicle (US cents / mile)
accelTime	0-60 mph acceleration time (seconds)
american	dummy code for whether American brand (1 or 0)
japanese	dummy code for whether Japanese brand (1 or 0)
chinese	dummy code for whether Chinese brand (1 or 0)
skorean	dummy code for whether S. Korean brand (1 or 0)
weights	weights for each individual computed so that the sample age and income demographics matched with thos

Source

Raw data downloaded from this repo

References

Helveston, J. P., Liu, Y., Feit, E. M., Fuchs, E. R. H., Klampfl, E., & Michalek, J. J. (2015). "Will Subsidies Drive Electric Vehicle Adoption? Measuring Consumer Preferences in the U.S. and China." Transportation Research Part A: Policy and Practice, 73, 96–112. doi: 10.1016/j.tra.2015.01.002

Examples

```
data(cars_china)

head(cars_china)
```

---

cars_us *Stated car choice observations by US car buyers*

Description

Data from Helveston et al. (2015) containing 448 stated choice observations from Chinese car buyers and 384 stated choice observations from US car buyers. Conjoint surveys were fielded in 2012 in four major Chinese cities (Beijing, Shanghai, Shenzhen, and Chengdu), online in the US on Amazon Mechanical Turk, and in person at the Pittsburgh Auto show. Participants were asked to select a vehicle from a set of three alternatives. Each participant answered 15 choice questions.

Usage

```
data(cars_us)
```

Format

Variable	Description
id	individual identifiers
obsnum	identifier for unique choice observation
choice	dummy code for choice (1 or 0)
hev	dummy code for HEV vehicle type (1 or 0)
phev10	dummy code for PHEV vehicle type w/10 mile electric driving range (1 or 0)
phev20	dummy code for PHEV vehicle type w/20 mile electric driving range (1 or 0)
phev40	dummy code for PHEV vehicle type w/40 mile electric driving range (1 or 0)
bev75	dummy code for BEV vehicle type w/75 mile electric driving range (1 or 0)
bev100	dummy code for BEV vehicle type w/100 mile electric driving range (1 or 0)
bev150	dummy code for BEV vehicle type w/150 mile electric driving range (1 or 0)
phevFastcharge	dummy code for whether PHEV vehicle had fast charging capability (1 or 0)
bevFastcharge	dummy code for whether BEV vehicle had fast charging capability (1 or 0)
price	price of vehicle ($USD)
opCost	operating cost of vehicle (US cents / mile)
accelTime	0-60 mph acceleration time (seconds)
american	dummy code for whether American brand (1 or 0)
japanese	dummy code for whether Japanese brand (1 or 0)
chinese	dummy code for whether Chinese brand (1 or 0)
skorean	dummy code for whether S. Korean brand (1 or 0)
weights	weights for each individual computed so that the sample age and income demographics matched with thos

Source

Raw data downloaded from this repo

References

Helveston, J. P., Liu, Y., Feit, E. M., Fuchs, E. R. H., Klampfl, E., & Michalek, J. J. (2015). "Will Subsidies Drive Electric Vehicle Adoption? Measuring Consumer Preferences in the U.S. and China." Transportation Research Part A: Policy and Practice, 73, 96–112. doi: 10.1016/j.tra.2015.01.002

Examples

```
data(cars_us)

head(cars_us)
```

---

coef.logitr *Get the model coefficients*

Description

Returns the coefficients of an estimated model of the 'logitr' class.

Usage

```
## S3 method for class 'logitr'
coef(object, ...)
```

Arguments

object	The output of a model estimated using the logitr() function.
...	other arguments

Value

A vector of the coefficients from a model estimated using the logitr() function.

Examples

```
# Run a MNL model in the Preference Space:
data(yogurt)

mnl_pref <- logitr(
  data = yogurt,
  choiceName = "choice",
  obsIDName = "obsID",
  parNames = c("price", "feat", "dannon", "hiland", "yoplait")
)

# Get the model coefficients:
coef(mnl_pref)
```

---

dummyCode *Add dummy-coded variables to data frame.*

Description

This function is depreciated. Use fastDummies::dummy_cols() instead.

Usage

```
dummyCode(df, vars)
```

Arguments

df	A data frame.
vars	The variables in the data frame for which you want to create new dummy coded variables.

Value

A a dataframe with new dummy-coded variables added.

getCoefTable *Get the coefficient summary table as a data frame*

Description

Returns a data frame of the coefficient summary table of a model estimated using the `logitr()` function.

Usage

```
getCoefTable(object)
```

Arguments

`object`	The output of a model estimated model using the `logitr()` function.

Value

Returns a data frame of the coefficient summary table of a model estimated using the `logitr()` function.

Examples

```
library(logitr)

# Run a MNL model in the preference space
mnl_pref <- logitr(
  data = yogurt,
  choiceName = "choice",
  obsIDName = "obsID",
  parNames = c("price", "feat", "dannon", "hiland", "yoplait")
)

# Get the coefficient summary table as a data frame
getCoefTable(mnl_pref)
```

logitr *The main function for estimating logit models*

Description

Use this function to estimate multinomial (MNL) and mixed logit (MXL) models with "Preference" space or "Willingness-to-pay" (WTP) space utility parameterizations. The function includes an option to run a multistart optimization loop with random starting points in each iteration, which is useful for non-convex problems like MXL models or models with WTP space utility parameterizations. The main optimization loop uses the `nloptr()` function to minimize the negative log-likelihood function.

Usage

```
logitr(
  data,
  choiceName,
  obsIDName,
  parNames,
  priceName = NULL,
  randPars = NULL,
  randPrice = NULL,
  modelSpace = "pref",
  weightsName = NULL,
  clusterName = NULL,
  robust = FALSE,
  options = list()
)
```

Arguments

data	The choice data, formatted as a data.frame object.
choiceName	The name of the column that identifies the choice variable.
obsIDName	The name of the column that identifies each choice observation.
parNames	The names of the parameters to be estimated in the model. Must be the same as the column names in the data argument. For WTP space models, do not include price in parNames.
priceName	The name of the column that identifies the price variable. Only required for WTP space models. Defaults to NULL.
randPars	A named vector whose names are the random parameters and values the distribution: 'n' for normal or 'ln' for log-normal. Defaults to NULL.
randPrice	The random distribution for the price parameter: 'n' for normal or 'ln' for log-normal. Only used for WTP space MXL models. Defaults to NULL.
modelSpace	Set to 'wtp' for WTP space models. Defaults to "pref".
weightsName	The name of the column that identifies the weights to be used in model estimation. Optional. Defaults to NULL.
clusterName	The name of the column that identifies the cluster groups to be used in model estimation. Optional. Defaults to NULL.
robust	Determines whether or not a robust covariance matrix is estimated. Defaults to FALSE. Specification of a clusterName or weightsName will override the user setting and set this to 'TRUE' (a warning will be displayed in this case). Replicates the functionality of Stata's cmcmmixlogit.
options	A list of options.

Details

The following options control the detailed behavior of the optimization algorithm. They must be provided as a named list to the options argument, e.g. options = list(...).

Argument	Description
numMultiStarts	Number of times to run the optimization loop, each time starting from a different random starting point f
keepAllRuns	Set to TRUE to keep all the model information for each multistart run. If TRUE, the logitr() function wil
startParBounds	Set the lower and upper bounds for the starting parameters for each optimization run, which are generat
startVals	A vector of values to be used as starting values for the optimization. Only used for the first run if numMul
useAnalyticGrad	Set to FALSE to use numerically approximated gradients instead of analytic gradients during estimation (
scaleInputs	By default each variable in data is scaled to be between 0 and 1 before running the optimization routine
standardDraws	By default, a new set of standard normal draws are generated during each call to logitr (the same draws
numDraws	The number of Halton draws to use for MXL models for the maximum simulated likelihood.
printLevel	The print level of the nloptr optimization loop. Use nloptr::nloptr.print.options() for more deta
xtol_rel	The relative x tolerance for the nloptr optimization loop. Use nloptr::nloptr.print.options() for
xtol_abs	The absolute x tolerance for the nloptr optimization loop. Use nloptr::nloptr.print.options() fo
ftol_rel	The relative f tolerance for the nloptr optimization loop. Use nloptr::nloptr.print.options() for
ftol_abs	The absolute f tolerance for the nloptr optimization loop. Use nloptr::nloptr.print.options() fo
maxeval	The maximum number of function evaluations for the nloptr optimization loop. Use nloptr::nloptr.
algorithm	The optimization algorithm that nloptr uses.

Value

The function returns a list object containing the following objects.

Value	Description
coef	The model coefficients at convergence.
standErrs	The standard errors of the model coefficients at convergence.
logLik	The log-likelihood value at convergence.
nullLogLik	The null log-likelihood value (if all coefficients are 0).
gradient	The gradient of the log-likelihood at convergence.
hessian	The hessian of the log-likelihood at convergence.
covariance	The covariance matrix at convergence.
numObs	The number of observations.
numParams	The number of model parameters.
startPars	The starting values used.
multistartNumber	The multistart run number for this model.
time	The user, system, and elapsed time to run the optimization.
iterations	The number of iterations until convergence.
message	A more informative message with the status of the optimization result.
status	An integer value with the status of the optimization (positive values are successes). Use statusCodes
modelSpace	The model space ('pref' or 'wtp').
priceName	The name of the column that identifies the price variable.
parNames	The names of the parameters to be estimated in the model.
randPars	A named vector whose names are the random parameters and values the distribution: 'n' for normal d
parSetup	A summary of the distributional assumptions on each model parameter ("f"="fixed", "n"="normal dis
weightsUsed	TRUE or FALSE for whether weights were used in the model.
clusterName	The name of the column used to identify the cluster groups used in model estimation.
numClusters	The number of clusters.
robust	TRUE or FALSE for whether or not a robust covariance matrix was estimated.
standardDraws	The draws used during maximum simulated likelihood (for MXL models).
randParSummary	A summary of any random parameters (for MXL models).

multistartSummary	A summary of the log-likelihood values for each multistart run (if more than one multistart was used).
options	A list of all the model options.

Examples

```
## Not run:

# For more detailed examples, visit
# https://jhelvy.github.io/logitr/articles/

library(logitr)

# Run a MNL model in the Preference Space:
mnl_pref <- logitr(
  data = yogurt,
  choiceName = "choice",
  obsIDName = "obsID",
  parNames = c("price", "feat", "dannon", "hiland", "yoplait")
)

# Run a MNL model in the WTP Space:
mnl_wtp <- logitr(
  data = yogurt,
  choiceName = "choice",
  obsIDName = "obsID",
  parNames = c("feat", "dannon", "hiland", "yoplait"),
  priceName = "price",
  modelSpace = "wtp"
)

## End(Not run)
```

---

predictChoices *Predict choices*

Description

Returns the expected choices for a set of one or more alternatives based on the results from an estimated model.

Usage

```
predictChoices(model, alts, obsIDName = NULL)
```

Arguments

model	The output of a model estimated model using the `logitr()` function. Include if you want to compare true choices from actual observations (e.g. hold outs) to the predicted choices.

alts	A data frame of a set of alternatives for which to predict choices. Each row is an alternative and each column an attribute corresponding to parameter names in the estimated model.
obsIDName	The name of the column that identifies each set of alternatives. Required if simulating results for more than one set of alternatives. Defaults to NULL (for a single set of alternatives).

Value

A data frame with the predicted choices for each alternative in alts.

Examples

```
## Not run:
# Run a MNL model in the Preference Space:
library(logitr)

mnl_pref <- logitr(
  data = yogurt,
  choiceName = "choice",
  obsIDName = "obsID",
  parNames = c("price", "feat", "brand")
)

# You can predict choices for any set of alternative, such as hold out
# samples or within-sample. For this example I will predict choices on
# the full yogurt data set, which was used to estimate the model.

# Run the simulation using the preference space MNL model:
choices_mnl_pref <- predictChoices(
  model      = mnl_pref,
  alts       = yogurt,
  obsIDName  = "obsID"
)

head(choices_mnl_pref)

# Compute the accuracy
chosen <-  subset(choices, choice == 1)
chosen$correct <- chosen$choice == chosen$choice_predict
sum(chosen$correct) / nrow(chosen)

## End(Not run)
```

---

predictProbs *Predict expected choice probabilities*

Description

Returns the expected choice probabilities for a single set or multiple sets of alternatives based on the results from an estimated model.

Usage

```
predictProbs(
  model,
  alts,
  obsIDName = NULL,
  computeCI = TRUE,
  alpha = 0.025,
  numDraws = 10^4
)
```

Arguments

model	The output of a model estimated model using the logitr() function.
alts	A data frame of a set of alternatives for which to predict choice probabilities. Each row is an alternative and each column an attribute corresponding to parameter names in the estimated model.
obsIDName	The name of the column that identifies each set of alternatives. Required if simulating results for more than one set of alternatives. Defaults to NULL (for a single set of alternatives).
computeCI	Should a confidence interval be computed? Defaults to TRUE.
alpha	The sensitivity of the computed confidence interval. Defaults to alpha = 0.025, reflecting a 95% CI.
numDraws	The number of draws to use in simulating uncertainty for the computed confidence interval.

Value

A data frame with the estimated choice probabilities for each alternative in alts.

Examples

```
## Not run:
# Run a MNL model in the Preference Space:
library(logitr)

mnl_pref <- logitr(
  data = yogurt,
  choiceName = "choice",
  obsIDName = "obsID",
  parNames = c("price", "feat", "brand")
)

# Create a set of alternatives for which to predict choice probabilities.
# Each row is an alternative and each column an attribute.
# In this example, I just use two of the choice observations from the
# yogurt dataset:
alts <- subset(yogurt, obsID %in% c(42, 13),
               select = c('obsID', 'price', 'feat', 'brand'))
alts
```

```
# Predict choice probabilities using the estimated preference space MNL
# model:
predictProbs(mnl_pref, alts, obsIDName = "obsID")

## End(Not run)
```

`recodeData` *Returns a list of the design matrix* `X` *and updated* `parNames` *and* `randPars` *to include any dummy-coded categorical or interaction variables.*

Description

Recodes the data and returns a list of the encoded design matrix (`X`) as well as two vectors (`parNames` and `randPars`) with discrete (categorical) variables and interaction variables added to `X`, `parNames`, and `randPars`.

Usage

```
recodeData(data, parNames, randPars)
```

Arguments

`data`	The choice data, formatted as a `data.frame` object.
`parNames`	The names of the parameters to be estimated in the model. Must be the same as the column names in the `data` argument. For WTP space models, do not include price in `parNames`.
`randPars`	A named vector whose names are the random parameters and values the distribution: `'n'` for normal or `'ln'` for log-normal. Defaults to `NULL`.

Value

A list of the design matrix (`X`) and two vectors (`parNames` and `randPars`) with discrete (categorical) variables and interaction variables added.

Examples

```
data(yogurt)

result <- recodeData(
    data = yogurt,
    parNames = c("price", "feat", "brand", "price*brand"),
    randPars = c(feat = "n", brand = "n")
)

result$parNames
result$randPars
head(result$X)
```

`simulateShares` *Simulate expected shares*

Description

This function has been depreciated since logitr version 0.1.4. Use `predictProbs()` instead.

Usage

```
simulateShares(
  model,
  alts,
  obsIDName = NULL,
  priceName = NULL,
  computeCI = TRUE,
  alpha = 0.025,
  numDraws = 10^4
)
```

Arguments

`model`	The output of a model estimated model using the `logitr()` function.
`alts`	A data frame of a set of alternatives for which to simulate shares. Each row is an alternative and each column an attribute corresponding to parameter names in the estimated model.
`obsIDName`	The name of the column that identifies each set of alternatives. Required if simulating results for more than one set of alternatives. Defaults to `NULL` (for a single set of alternatives).
`priceName`	The name of the parameter that identifies price. Only required for WTP space models. Defaults to `NULL`.
`computeCI`	Should a confidence interval be computed? Defaults to `TRUE`.
`alpha`	The sensitivity of the computed confidence interval. Defaults to `alpha = 0.025`, reflecting a 95% CI.
`numDraws`	The number of draws to use in simulating uncertainty for the computed confidence interval.

Value

A data frame with the estimated shares for each alternative in `alts`.

statusCodes *View a description the nloptr status codes*

Description

Prints a description of the status codes from the nloptr optimization routine.

Usage

```
statusCodes()
```

Details

Code	Description
1	Generic success return value.
2	Optimization stopped because stopval was reached.
3	Optimization stopped because ftol_rel or ftol_abs was reached.
4	Optimization stopped because xtol_rel or xtol_abs was reached.
5	Optimization stopped because maxeval was reached.
6	Optimization stopped because maxtime was reached.
-1	Generic failure code.
-2	Invalid arguments (e.g. lower bounds are bigger than upper bounds,
an unknown algorithm was specified, etc.).	
-3	Ran out of memory.
-4	Halted because roundoff errors limited progress (in this case, the optimization st
-5	Halted because of a forced termination: the user called `nlopt_force_stop(opt`

Value

No return value; prints a summary of the `nloptr` status codes to the console.

Examples

```
statusCodes()
```

summary.logitr *View summary of estimated model*

Description

Prints a summary of a model estimated using the `logitr()` function

Usage

```
## S3 method for class 'logitr'
summary(object, ...)
```

Arguments

object	The output of a model estimated model using the logitr() function.
...	other arguments

Value

No return value; prints a summary of the model results to the console.

Examples

```
# Run a MNL model in the Preference Space with a multistart:
data(yogurt)

mnl_pref <- logitr(
  data = yogurt,
  choiceName = "choice",
  obsIDName = "obsID",
  parNames = c("price", "feat", "dannon", "hiland", "yoplait"),
  options = list(
    numMultiStarts = 5,
    keepAllRuns = TRUE
  )
)

# View a summary of the model:
summary(mnl_pref)
```

---

wtp *Get WTP from a preference space model*

Description

Returns the computed WTP from a preference space model.

Usage

```
wtp(model, priceName)
```

Arguments

model	The output of a "preference space" model estimated using the logitr() function.
priceName	The name of the parameter that identifies price.

Details

Willingness to pay is computed by dividing the estimated parameters of a utility model in the "preference" space by the price parameter. Uncertainty is handled via simulation.

Value

A data frame of the WTP estimates.

Examples

```
# Run a MNL model in the Preference Space:
library(logitr)

mnl_pref <- logitr(
  data = yogurt,
  choiceName = "choice",
  obsIDName = "obsID",
  parNames = c("price", "feat", "dannon", "hiland", "yoplait")
)

# Get the WTP implied from the preference space model
wtp(mnl_pref, priceName = "price")
```

---

wtpCompare *Compare WTP from preference and WTP space models*

Description

Returns a comparison of the WTP between a preference space and WTP space model.

Usage

```
wtpCompare(model_pref, model_wtp, priceName)
```

Arguments

model_pref	The output of a "preference space" model estimated using the `logitr()` function.
model_wtp	The output of a "willingness to pay space" model estimated using the `logitr()` function.
priceName	The name of the parameter that identifies price.

Details

Willingness to pay (WTP) is first computed from the preference space model by dividing the estimated parameters by the price parameter. Then those estimates are compared against the WTP values directly estimated from the "WTP" space model. Uncertainty is handled via simulation.

Value

A data frame comparing the WTP estimates from preference space and WTP space models.

Examples

```
# Run a MNL model in the Preference Space:
library(logitr)

mnl_pref <- logitr(
  data = yogurt,
  choiceName = "choice",
  obsIDName = "obsID",
  parNames = c("price", "feat", "dannon", "hiland", "yoplait")
)

# Get the WTP implied from the preference space model
wtp_mnl_pref <- wtp(mnl_pref, priceName = "price")

# Run a MNL model in the WTP Space:
mnl_wtp <- logitr(
  data = yogurt,
  choiceName = "choice",
  obsIDName = "obsID",
  parNames = c("feat", "dannon", "hiland", "yoplait"),
  priceName = "price",
  modelSpace = "wtp",
  options = list(startVals = wtp_mnl_pref$Estimate)
)

# Compare the WTP between the two spaces:
wtpCompare(mnl_pref, mnl_wtp, priceName = "price")
```

---

yogurt *Choice observations of yogurt purchases by 100 households*

Description

Data from Jain et al. (1994) containing 2,412 choice observations from a series of yogurt purchases by a panel of 100 households in Springfield, Missouri, over a roughly two-year period. The data were collected by optical scanners and contain information about the price, brand, and a "feature" variable, which identifies whether a newspaper advertisement was shown to the customer. There are four brands of yogurt: Yoplait, Dannon, Weight Watchers, and Hiland, with market shares of 34%, 40%, 23% and 3%, respectively.

Usage

```
data(yogurt)
```

Format

Variable	Description
id	individual identifiers
obsID	identifier for unique choice observation
alt	alternative in each choice observation
choice	dummy code for choice (1 or 0)
price	price of yogurt
feat	dummy for whether a newspaper advertisement was shown to the customer (1 or 0)
brand	yogurt brand: "yoplait", "dannon", "hiland", or "weight" (for weight watcher)
dannon	dummy variable for the "dannon" brand (1 or 0)
hiland	dummy variable for the "hiland" brand (1 or 0)
weight	dummy variable for the "weight" brand (1 or 0)
yoplait	dummy variable for the "yoplait" brand (1 or 0)

Source

Raw data downloaded from the package mlogit v0.3-0 by Yves Croissant archive

References

Dipak C. Jain, Naufel J. Vilcassim & Pradeep K. Chintagunta (1994) A Random-Coefficients Logit Brand-Choice Model Applied to Panel Data, Journal of Business & Economic Statistics, 12:3, 317-328, doi: 10.1080/07350015.1994.10524547

Examples

```
data(yogurt)

head(yogurt)
```

Index