Package ‘fic’

March 27, 2025

Title Focused Information Criteria for Model Comparison

Version 1.0.1

Date 2025-03-24

Description
Compares how well different models estimate a quantity of interest (the ``focus'') so that different models may be preferred for different purposes. Comparisons within any class of models fitted by maximum likelihood are supported, with shortcuts for commonly-used classes such as generalised linear models and parametric survival models. The methods originate from Claeskens and Hjort (2003) <doi:10.1198/016214503000000819> and Claeskens and Hjort (2008, ISBN:9780521

Maintainer Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

Depends R (>= 2.10)

Imports stats, numDeriv, mvtnorm, ggplot2, scales, survival, tensor, abind, mgcv

URL https://github.com/chjackson/fic

BugReports https://github.com/chjackson/fic/issues

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Suggests knitr, rmarkdown, testthat, msm(>= 1.6.6), flexsurv, sn, gapminder, GGally

VignetteBuilder knitr

NeedsCompilation no

Author Christopher Jackson [cre, aut] (package design and programming), Gerda Claeskens [aut] (method development and design advice), Howard Thom [ctb]

Repository CRAN

Date/Publication 2025-03-27 17:20:02 UTC

Contents

`fic-package` *Focused Information Criteria for Model Comparison*

Description

For a full explanation of the methods behind the **fic** package and worked examples of using it, see the main package vignette "Different models for different purposes: focused model comparison in R"

Author(s)

Maintainer: Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk> (package design and programming)

Authors:

- Gerda Claeskens <gerda.claeskens@kuleuven.be> (method development and design advice)

Other contributors:

- Howard Thom <howard.thom@bristol.ac.uk> [contributor]

See Also

Useful links:

- https://github.com/chjackson/fic
- Report bugs at https://github.com/chjackson/fic/issues

`all_inds.default` *Form indicator matrix describing all submodels of a general linear wide model*

Description

Form indicator matrix describing all possible submodels of a general linear wide model, where the submodels are defined by selected covariates.

Usage

```
## Default S3 method:
all_inds(wide, inds0 = NULL, auxpars = NULL, ...)

all_inds(wide, inds0, ...)

## S3 method for class 'lm'
all_inds(wide, inds0, ...)

## S3 method for class 'glm'
all_inds(wide, inds0, ...)

## S3 method for class 'coxph'
all_inds(wide, inds0, ...)
```

Arguments

`wide`	A fitted model of standard R format, such that `terms(wide)` returns information about the terms of the model formula. Models outside standard R packages may not support this.
`inds0`	Narrow model indicators, in format described in `fic`.
`auxpars`	Names of parameters in the wide model other than the covariate effects being selected from. By default, for linear and generalised linear models this is `c("(Intercept)")`, and for Cox regression this is omitted.
`...`	Other arguments. Currently unused.

Value

A matrix in the format required by the `inds` argument of `fic()`, representing all possible submodels of the wide model.

The number of rows is the number of models, and the number of columns is the number of parameters in the wide model. The r, s entry of the matrix is a 1 if the rth submodel includes parameter s, and 0 otherwise.

If a factor is included (excluded) from the submodel, then all corresponding parameters are included (excluded).

Examples

```
bwt.glm <- glm(low ~ lwtkg + age + smoke, data=birthwt, family="binomial")
all_inds(bwt.glm, inds0=c(1,0,0,0))

# note no intercept term in Cox models, so inds0 has two elements here
library(survival)
wide <- coxph(Surv(years, death==1) ~ sex + thick_centred, data=melanoma)
all_inds(wide, inds0=c(0,0))
```

---

birthwt *Risk factors associated with low infant birth weight*

Description

Risk factors associated with low infant birth weight

Usage

```
birthwt
```

Format

A data frame with 189 rows and 19 variables. The first 10 columns are included in the dataset of the same name in the **MASS** package. The remaining 9 columns are defined in Claeskens and Hjort (2008), and are included in this dataset for convenience.

low indicator of birth weight less than 2.5 kg

age mother's age in years

lwt mother's weight in pounds at last menstrual period

race mother's race ('1' = white, '2' = black, '3' = other)

smoke smoking status during pregnancy

ptl number of previous premature labours

ht history of hypertension

ui presence of uterine irritability

ftv number of physician visits during the first trimester

bwt birth weight in grams

smokeui Binary indicator for both smoking and uterine irritation

smokeage Interaction between age and binary (0/1) smoking status, that is, `age` for smokers and zero for non-smokers.

intercpt Intercept term (all 1)

raceother Binary indicator for `race` "other"

raceblack Binary indicator for `race` "black"

ftv2p Binary indicator for `ftv`, number of physician visits during the first trimester, 2 or more

ftv1 Binary indicator for `ftv` 1

ptd Binary indicator for `ptl`, number of previous premature labours, 1 or more

lwtkg Weight measured in kg, as used in Claeskens and Hjort. Note `lwt`, as used in **MASS**, is in pounds.

Source

MASS package (Venables, W. N. and Ripley, B. D. (2002) Modern Applied Statistics with S. Fourth edition. Springer); originally from Hosmer, D.W. and Lemeshow, S. (1989) Applied Logistic Regression. New York: Wiley

References

Claeskens, G., & Hjort, N. L. (2008). Model selection and model averaging (Vol. 330). Cambridge: Cambridge University Press.

`expand_inds` *Form 'fic' model indicator argument in presence of factors*

Description

Given a model indicator `inds` identifying terms in a regression model, convert this to the format needed for `fic` by converting indicators for regression terms to indicators for inclusion of parameters. Only required if there are factors.

Usage

```
expand_inds(inds, wide)
```

Arguments

inds	Matrix or vector of indicators for which parameters are included in the submodel or submodels to be assessed. A matrix should be supplied if there are multiple submodels. This should have number of rows equal to the number of submodels, and number of columns equal to the total number of parameters in the wide model. It contains 1s in the positions where the parameter is included in the submodel, and 0s in positions where the parameter is excluded. This should always be 1 in the positions defining the narrow model, as specified in inds0.
wide	Fitted model object containing the wide model.

Details

If a regression term is a factor, then the 0 or 1 indicating its inclusion/exclusion is replicated to a length given by the number of factor levels minus 1, that is, the number of parameters pertaining to that factor.

This function only works for classes of models for which the model.matrix function is understood and returns objects with an "assign" attribute. This includes all the commonly-used models in base R.

Examples

```
# Five terms in this model: intercept and four covariates,
# but the covariate "ftv" is a factor with 3 levels,
# so there are six parameters
bwt.glm <- glm(low ~ lwtkg + age + smoke + ftv, data=birthwt, family="binomial")

## Convert indicator for terms to indicator for parameters
inds <- rbind(c(1,1,1,0,0),
              c(1,1,1,1,1))
expand_inds(inds, bwt.glm)
```

---

fic.coxph *Focused information criteria for Cox proportional hazard regression models*

Description

Focused model comparison for Cox models fitted with coxph from the survival package. Built-in focuses include the hazard ratio, survival and cumulative hazard.

Usage

```
## S3 method for class 'coxph'
fic(
  wide,
  inds,
  inds0 = NULL,
  gamma0 = 0,
  focus,
  X = NULL,
  t = NULL,
  sub = "auto",
  tidy = TRUE,
  ...
)
```

Arguments

`wide`	Fitted model object containing the wide model.
`inds`	Matrix or vector of indicators for which parameters are included in the submodel or submodels to be assessed. A matrix should be supplied if there are multiple submodels. This should have number of rows equal to the number of submodels, and number of columns equal to the total number of parameters in the wide model. It contains 1s in the positions where the parameter is included in the submodel, and 0s in positions where the parameter is excluded. This should always be 1 in the positions defining the narrow model, as specified in `inds0`.
`inds0`	Vector of indicators specifying the narrow model, in the same format as `inds`. If this is omitted, the narrow model is assumed to be defined by the first row of `inds` (if `inds` is a matrix), or `inds` itself if this is a vector.
`gamma0`	Vector of special values taken by the parameters $gamma$ which define the narrow model. This defaults to 0, as in covariate selection, where "excluded" coefficients are fixed to 0. This should either be a scalar, assumed to be the same for all parameters fixed in the narrow model, or a vector of length equal to the number of parameters from the wide model which are fixed in the narrow model, that is, the number of entries of `inds0` which are zero.
`focus`	Three built-in focus quantities are supported: `"hr"` for the hazard ratio between individuals with covariate values of `X` and 0. `"survival"` for the survival probability at time or times given in `t`. `"cumhaz"` for the cumulative hazard at time or times given in `t`. Alternatively, a list of three R functions can be supplied, with components named `"focus"`, `"deriv"` and `"dH"` respectively giving the focus, derivative with respect to the log hazard ratios, and derivative with respect to the cumulative hazards. Each function should have arguments `par`, `H0`, `X` and `t`, giving the log hazard ratios, baseline cumulative hazard, covariate values and time points

at which the focus function should be evaluated. See the section "User-defined focuses" below for a little more information.

`X` Covariate values to evaluate the focus at. See `fic`, argument `...`, for the required format.

`t` times to evaluate the focus at. Only relevant for survival and cumulative hazard focuses, as the hazard ratio is constant through time.

`sub` If `"auto"` (the default) then the submodels are fitted automatically within this function. If `NULL` they are not fitted, and focus estimates are not returned with the results.

`tidy` If `TRUE` the results are returned as a data frame with variables to indicate the submodels, focuses and corresponding result statistics. If `FALSE`, the results are returned as a three-dimensional array, with dimensions indexed by the submodels, result statistics and focuses respectively.

`...` Other arguments to the focus function can be supplied here.

The built-in focus functions `prob_logistic` and `mean_normal` take an argument `X` giving covariate values defining the focus. This can either be a matrix or a vector, or a list or data frame that can be coerced into a matrix.

If just one focus is needed, then `X` can be a vector of length equal to the number of parameters in the wide model.

To compute focused model comparison statistics for multiple focuses defined by the same focus function evaluated at multiple covariate values, `X` should be a matrix, with number of columns equal to the number of parameters in the wide model, and number of rows equal to the number of alternative focuses.

For a typical regression model, the first parameter will denote an intercept, so the first value of `X` should be 1, and the remaining values should correspond to covariates whose coefficients form parameters of the wide model. See the examples in the vignette.

Arguments to the focus function other than `X` can also be supplied as a matrix, vector, list or data frame in the same way. An exception is when the argument is supplied as a vector, this is assumed to refer to multiple focuses. For example, suppose the focus function defines the quantile of a distribution, and takes an argument `focus_p`, then calling `fic(...,focus_p=c(0.1, 0.9))` indicates two alternative focuses defined by the 0.1 and 0.9 quantiles.

Details

Stratified Cox models are not currently supported.

User-defined focuses

Each function should have four arguments:

`par` Vector of estimated coefficients, the log hazard ratios in the Cox model.

`H0` Cumulative hazard estimate at a set of times, in the form of the output from `basehaz`. The function `get_H0` can be used on this estimate to obtain the estimate at any other times by interpolation.

`X` Matrix of covariates, with `ncov` rows and `npar` columns, where `ncov` is the number of alternative covariate values definining alternative focuses we want to compare models for, and `npar` is the number of coefficients in the model.

`t` Vector of times defining alternative focus quantities (such as the survival)

For examples, examine the source for the built-in functions

`fic:::cox_hr`,`fic:::cox_hr_deriv`,`fic:::cox_hr_dH` for the hazard ratio between `X` and `0`

`fic:::cox_cumhaz`,`fic:::cox_cumhaz_deriv`,`fic:::cox_cumhaz_dH` for the cumulative hazard

`fic:::cox_survival`,`fic:::cox_survival_deriv`,`fic:::cox_survival_dH` for the survival.

Examples

```
## Example of focused covariate selection in a Cox model
## For more information see the main package vignette.

library(survival)
wide <- coxph(Surv(years, death==1) ~ sex + thick_centred + infilt +
              epith + ulcer + depth + age, data=melanoma)

## Define models to be selected from
## Sex included in all models
## Select between models including all combinations of other covariates
inds0 <- expand_inds(c(1,0,0,0,0,0,0), wide)
combs <- all_inds(wide, inds0)

## Covariate values defining focus
newdata <- with(melanoma,
                data.frame(sex = c("female","male"),
                           thick_centred = tapply(thick_centred, sex, mean),
                           infilt=4, epith=1, ulcer=1, depth=2,
                           age = tapply(age, sex, mean)))
X <- newdata_to_X(newdata, wide, intercept=FALSE)

## Focus is 5-year survival for these covariate values
ficall <- fic(wide, inds=combs, inds0=inds0, focus="survival", X=X, t=5)
ggplot_fic(ficall)
summary(ficall)
```

`fic.default` *Focused information criteria: main user interface*

Description

Focused information criteria for general models. These methods estimate the bias and variance of estimates of a quantity of interest (the "focus") when smaller submodels are used in place of a "wide" model that is assumed to generate the data but may not give precise enough estimates.

Usage

```
## Default S3 method:
fic(
  wide,
  inds,
  inds0 = NULL,
  gamma0 = 0,
  focus = NULL,
  focus_deriv = NULL,
  wt = NULL,
  sub = NULL,
  fns = NULL,
  FIC = FALSE,
  B = 0,
  loss = loss_mse,
  tidy = TRUE,
  ...
)

fic(wide, ...)
```

Arguments

wide	Fitted model object containing the wide model.
inds	Matrix or vector of indicators for which parameters are included in the submodel or submodels to be assessed. A matrix should be supplied if there are multiple submodels. This should have number of rows equal to the number of submodels, and number of columns equal to the total number of parameters in the wide model. It contains 1s in the positions where the parameter is included in the submodel, and 0s in positions where the parameter is excluded. This should always be 1 in the positions defining the narrow model, as specified in inds0.
inds0	Vector of indicators specifying the narrow model, in the same format as inds. If this is omitted, the narrow model is assumed to be defined by the first row of inds (if inds is a matrix), or inds itself if this is a vector.
gamma0	Vector of special values taken by the parameters $gamma$ which define the narrow model. This defaults to 0, as in covariate selection, where "excluded" coefficients are fixed to 0. This should either be a scalar, assumed to be the same for all parameters fixed in the narrow model, or a vector of length equal to the number of parameters from the wide model which are fixed in the narrow model, that is, the number of entries of inds0 which are zero.
focus	An R function with: • first argument named par, denoting a vector of parameters, of the same length as in wide model

- other arguments defining alternative focuses. These are supplied through the `...` argument to `fic`. In the built-in examples, there is an argument named `X`, denoting alternative covariate values. The required format is documented below.

The function should return the focus quantity of interest. If additional arguments are supplied which are vectors or matrices, e.g. `X`, then these are assumed to represent multiple focuses, and `focus` should return a vector giving the focus for `par` and each row of `X`. Otherwise `focus` should return a scalar giving the focus value at `par`.

Not required if `focus_deriv` is specified.

Alternatively, `focus` can be a character string naming a built-in focus function supplied by the **fic** package. Currently these include:

`"prob_logistic"`, the probability of the outcome in a logistic regression model

`"mean_normal"` the mean outcome in a normal linear regression model

See `focus_fns` for the functions underlying these built-in focuses.

`focus_deriv` Vector of partial derivatives of the focus function with respect to the parameters in the wide model. This is not usually needed, as it can generally be computed automatically and accurately from the function supplied in `focus`, using numerical differentiation.

`wt` Vector of weights to apply to different covariate values in `X`. This should have length equal to the number of alternative values for the covariates, that is, the number of alternative focuses of interest. The covariate-specific focused model comparison statistics are then supplemented by averaged statistics for a population defined by this distribution of covariate values. If this argument is omitted, the values are assumed to have equal weight when computing the average. The weights are not normalised, though the interpretation is unclear if the weights don't sum to one.

`sub` List of fitted model objects corresponding to each submodel to be assessed.

For some classes of models with built in methods for `fic`, e.g. `fic.glm`, the submodels are fitted automatically by default, so this argument does not need to be supplied.

Otherwise, this argument can be omitted, but it is required if you want the estimate of the focus function under each submodel to be included in the results, which is usually the case.

`fns` Named list of functions to extract the quantities from the fitted model object that are required to calculate the focused model comparison statistics. By default this is

`list(coef=coef, nobs=nobs, vcov=vcov)`

Suppose the fitted model object is called `mod`. This default list assumes that

- `coef(mod)` returns the vector of parameter estimates,
- `vcov(mod)` returns the covariance matrix for the parameter estimates,
- `nobs(mod)` returns the number of observations used in the model fit. Only required if the 'classic' FIC is required, and not required to compute the mean square error of the focus.

If one or more of these functions does not work for `mod`, then the defaults can be changed. For example, suppose the functions `coef()`, `nobs()` and `vcov()` are not understood (or return something different) for your class of model objects, but the parameters are stored in `mod$estimates`, the number of observations is in `mod$data$nobs`, and the covariance matrix is in `mod$cov`, then the `fns` argument should be set to

```
list( coef = function(x){x$estimates}, nobs = function(x){x$data$nobs},
vcov = function(x){x$cov} )
```

If less than three components are specified in `fns`, then the missing components are assumed to take their default values.

`FIC` If `TRUE`, then the Focused Information Criterion is returned with the results alongside the mean squared error and its components. This is done for built-in model classes, but optional for user-defined model classes, since it requires knowledge of the sample size `n` as well as the estimates and covariance matrix under the wide model.

`B` If `B` is 0 (the default) the standard analytic formulae for the focused model comparison statistics are used with mean square error loss. If `B`>0, then a parametric bootstrap method is used with `B` bootstrap samples, and the loss specified in the `loss` argument. More details of this approach are given in the package vignette "Focused model comparison with bootstrapping and alternative loss functions".

`loss` A function returning an estimated loss for a submodel estimate under the sampling distribution of the wide model. Only applicable when using bootstrapping. This should have two arguments `sub` and `wide`. `sub` should be a scalar giving the focus estimate from a submodel. `wide` should be a vector with a sample of focus estimates from the wide model, e.g. generated by a bootstrap method. By default this is a function calculating the root mean square error of the submodel estimate. An example is given in the vignette "Focused model comparison with bootstrapping and alternative loss functions".

`tidy` If `TRUE` the results are returned as a data frame with variables to indicate the submodels, focuses and corresponding result statistics. If `FALSE`, the results are returned as a three-dimensional array, with dimensions indexed by the submodels, result statistics and focuses respectively.

`...` Other arguments to the focus function can be supplied here.

The built-in focus functions `prob_logistic` and `mean_normal` take an argument `X` giving covariate values defining the focus. This can either be a matrix or a vector, or a list or data frame that can be coerced into a matrix.

If just one focus is needed, then `X` can be a vector of length equal to the number of parameters in the wide model.

To compute focused model comparison statistics for multiple focuses defined by the same focus function evaluated at multiple covariate values, `X` should be a matrix, with number of columns equal to the number of parameters in the wide model, and number of rows equal to the number of alternative focuses.

For a typical regression model, the first parameter will denote an intercept, so the first value of `X` should be 1, and the remaining values should correspond to covariates whose coefficients form parameters of the wide model. See the examples in the vignette.

Arguments to the focus function other than `X` can also be supplied as a matrix, vector, list or data frame in the same way. An exception is when the argument is supplied as a vector, this is assumed to refer to multiple focuses. For example, suppose the focus function defines the quantile of a distribution, and takes an argument `focus_p`, then calling `fic(...,focus_p=c(0.1, 0.9))` indicates two alternative focuses defined by the 0.1 and 0.9 quantiles.

Value

The returned data frame or array contains the following components, describing characteristics of the defined submodel. See the package vignette for full, formal definitions, and Chapter 6 of Claeskens and Hjort, 2008.

`rmse`	The root mean square error of the estimate of the focus quantity. Defined as the square root of (squared unadjusted bias plus variance). This is an asymptotically unbiased estimator, but may occasionally be indeterminate if the estimate of the squared bias plus variance is negative.
`rmse.adj`	The root mean square error, based on a bias estimator which is adjusted to avoid negative squared bias. Defined on page 157 of Claeskens and Hjort as the sum of the variance and the squared adjusted bias.
`bias`	The estimated bias of the focus quantity, adjusted to avoid negative squared bias. This is defined as the square root of the quantity $sqb3(S)$ from page 152 of Claeskens and Hjort, multiplied by the sign of the unadjusted bias.
`se`	The estimated standard error (root variance) of the focus quantity. Defined on page 157.
`FIC`	The focused information criterion (equation 6.1 from Claeskens and Hjort), if `FIC=TRUE` was supplied.

The object returned by `fic` also has the following attributes, which can be extracted with the `attr` function.

`iwide`	Index of the wide model in the vector of submodels, or `NULL` if the wide model is not included.
`inarr`	Index of the narrow model in the vector of submodels, or `NULL` if the wide model is not included.
`sub`	List of fitted submodel objects.
`parnames`	Vector of names of parameters in the wide model.
`inds`	Submodel indicators, as supplied in the `inds` argument.

References

Claeskens, G., & Hjort, N. L. (2008). Model selection and model averaging (Vol. 330). Cambridge: Cambridge University Press.

Claeskens, G., & Hjort, N. L. (2003). The focused information criterion. Journal of the American Statistical Association, 98(464), 900-916.

Examples

```
wide.glm <- glm(low ~ lwtkg + age + smoke + ht + ui + smokeage + smokeui,
                data=birthwt, family=binomial)
inds <- rbind(
              narrow = c(1,1,0,0,0,0,0,0),
              mod1 = c(1,1,1,1,0,0,0,0),
              wide = c(1,1,1,1,1,1,1,1)
)
vals.smoke <-    c(1, 58.24, 22.95, 1, 0, 0, 22.95, 0)
vals.nonsmoke <- c(1, 59.50, 23.43, 0, 0, 0, 0, 0)
X <- rbind("Smokers"=vals.smoke, "Non-smokers"=vals.nonsmoke)

fic(wide=wide.glm, inds=inds, focus="prob_logistic", X=X)

focus <- function(par, X)plogis(X %*% par)
fic(wide=wide.glm, inds=inds, focus=focus, X=X)   # equivalent
```

`fic.flexsurvreg` *Focused information criteria for flexible parametric survival models*

Description

Focused information criteria for parametric survival models fitted with the **flexsurv** package.

Usage

```
## S3 method for class 'flexsurvreg'
fic(
  wide,
  inds,
  inds0 = NULL,
  gamma0 = 0,
  focus = NULL,
  focus_deriv = NULL,
  wt = NULL,
  sub = NULL,
  B = 0,
  loss = loss_mse,
  ...
)
```

Arguments

`wide`	Object of class `"flexsurvreg"` containing the wide model. These are returned, for example, by the `flexsurvreg` and the `flexsurvspline` functions.

`inds` Matrix or vector of indicators for which parameters are included in the submodel or submodels to be assessed.

A matrix should be supplied if there are multiple submodels. This should have number of rows equal to the number of submodels, and number of columns equal to the total number of parameters in the wide model. It contains 1s in the positions where the parameter is included in the submodel, and 0s in positions where the parameter is excluded. This should always be 1 in the positions defining the narrow model, as specified in `inds0`.

`inds0` Vector of indicators specifying the narrow model, in the same format as `inds`. If this is omitted, the narrow model is assumed to be defined by the first row of `inds` (if `inds` is a matrix), or `inds` itself if this is a vector.

`gamma0` Vector of special values taken by the parameters $gamma$ which define the narrow model.

This defaults to 0, as in covariate selection, where "excluded" coefficients are fixed to 0.

This should either be a scalar, assumed to be the same for all parameters fixed in the narrow model, or a vector of length equal to the number of parameters from the wide model which are fixed in the narrow model, that is, the number of entries of `inds0` which are zero.

`focus` An R function with:

- first argument named `par`, denoting a vector of parameters, of the same length as in wide model
- other arguments defining alternative focuses. These are supplied through the `...` argument to `fic`. In the built-in examples, there is an argument named `X`, denoting alternative covariate values. The required format is documented below.

The function should return the focus quantity of interest. If additional arguments are supplied which are vectors or matrices, e.g. `X`, then these are assumed to represent multiple focuses, and `focus` should return a vector giving the focus for `par` and each row of `X`. Otherwise `focus` should return a scalar giving the focus value at `par`.

Not required if `focus_deriv` is specified.

Alternatively, `focus` can be a character string naming a built-in focus function supplied by the **fic** package. Currently these include:

`"prob_logistic"`, the probability of the outcome in a logistic regression model

`"mean_normal"` the mean outcome in a normal linear regression model

See `focus_fns` for the functions underlying these built-in focuses.

`focus_deriv` Vector of partial derivatives of the focus function with respect to the parameters in the wide model. This is not usually needed, as it can generally be computed automatically and accurately from the function supplied in `focus`, using numerical differentiation.

`wt` Vector of weights to apply to different covariate values in `X`. This should have length equal to the number of alternative values for the covariates, that is, the number of alternative focuses of interest. The covariate-specific focused model

comparison statistics are then supplemented by averaged statistics for a population defined by this distribution of covariate values. If this argument is omitted, the values are assumed to have equal weight when computing the average. The weights are not normalised, though the interpretation is unclear if the weights don't sum to one.

`sub` List of objects of class `flexsurvreg` containing the submodels to be assessed. Optional. Only required if you want the estimate of the focus function under the submodels to be included in the results.

`B` If `B` is 0 (the default) the standard analytic formulae for the focused model comparison statistics are used with mean square error loss. If `B`>0, then a parametric bootstrap method is used with `B` bootstrap samples, and the loss specified in the `loss` argument. More details of this approach are given in the package vignette "Focused model comparison with bootstrapping and alternative loss functions".

`loss` A function returning an estimated loss for a submodel estimate under the sampling distribution of the wide model. Only applicable when using bootstrapping. This should have two arguments `sub` and `wide`. `sub` should be a scalar giving the focus estimate from a submodel. `wide` should be a vector with a sample of focus estimates from the wide model, e.g. generated by a bootstrap method. By default this is a function calculating the root mean square error of the submodel estimate. An example is given in the vignette "Focused model comparison with bootstrapping and alternative loss functions".

`...` Other arguments to the focus function can be supplied here.

The built-in focus functions `prob_logistic` and `mean_normal` take an argument `X` giving covariate values defining the focus. This can either be a matrix or a vector, or a list or data frame that can be coerced into a matrix.

If just one focus is needed, then `X` can be a vector of length equal to the number of parameters in the wide model.

To compute focused model comparison statistics for multiple focuses defined by the same focus function evaluated at multiple covariate values, `X` should be a matrix, with number of columns equal to the number of parameters in the wide model, and number of rows equal to the number of alternative focuses.

For a typical regression model, the first parameter will denote an intercept, so the first value of `X` should be 1, and the remaining values should correspond to covariates whose coefficients form parameters of the wide model. See the examples in the vignette.

Arguments to the focus function other than `X` can also be supplied as a matrix, vector, list or data frame in the same way. An exception is when the argument is supplied as a vector, this is assumed to refer to multiple focuses. For example, suppose the focus function defines the quantile of a distribution, and takes an argument `focus_p`, then calling `fic(...,focus_p=c(0.1, 0.9))` indicates two alternative focuses defined by the 0.1 and 0.9 quantiles.

Details

Any situation where all models being compared are special cases of a single "wide" model are supported. Examples include covariate selection, selection between models for the baseline hazard/survival with different levels of flexibility (e.g. comparing exponential, Weibull and general-

ized gamma). Some of these are illustrated in the **fic** package vignette "Examples of focused model comparison: parametric survival models".

The choice between flexsurvspline models with different numbers of knots is not supported, unless perhaps if the knot locations are defined manually so that models are nested within each other, but this has not been investigated.

Examples

```
## Simulated example from the "fic" package vignette on
## parametric survival modelling.
## See this vignette for more details and more examples.

set.seed(1)

if (requireNamespace("flexsurv", quietly=TRUE)){

## Simulate from an exponential
y <- rexp(50); cen <- rep(1,50)

## Fit wide generalized gamma, and compare
## exponential, weibull and generalized gamma models
indmat <- rbind(exp    = c(1,0,0),
                weib   = c(1,1,0),
                ggamma = c(1,1,1))
gge <- flexsurv::flexsurvreg(survival::Surv(y, cen) ~ 1, dist="gengamma")

## Focus is restricted mean survival over 8 time units
focus <- function(par){
   flexsurv::rmst_gengamma(8, par[1], exp(par[2]), par[3])
}

## Weibull model actually has lowest FIC and RMSE even though it's
## not true: extra variability is deemed worth alleviating the
## risk of bias.

fic(gge, inds=indmat, gamma0=c(0,1), focus=focus)

}
```

`fic.glm` *Focused information criteria for generalized linear models*

Description

Focused information criteria for generalized linear models fitted with glm. Used to compare models with different covariates (more generally, different linear terms).

Usage

```
## S3 method for class 'glm'
fic(
  wide,
  inds,
  inds0 = NULL,
  gamma0 = 0,
  focus = NULL,
  focus_deriv = NULL,
  wt = NULL,
  sub = "auto",
  B = 0,
  loss = loss_mse,
  ...
)
```

Arguments

wide	Fitted model object containing the wide model.
inds	Matrix or vector of indicators for which parameters are included in the submodel or submodels to be assessed. A matrix should be supplied if there are multiple submodels. This should have number of rows equal to the number of submodels, and number of columns equal to the total number of parameters in the wide model. It contains 1s in the positions where the parameter is included in the submodel, and 0s in positions where the parameter is excluded. This should always be 1 in the positions defining the narrow model, as specified in inds0.
inds0	Vector of indicators specifying the narrow model, in the same format as inds. If this is omitted, the narrow model is assumed to be defined by the first row of inds (if inds is a matrix), or inds itself if this is a vector.
gamma0	Vector of special values taken by the parameters $gamma$ which define the narrow model. This defaults to 0, as in covariate selection, where "excluded" coefficients are fixed to 0. This should either be a scalar, assumed to be the same for all parameters fixed in the narrow model, or a vector of length equal to the number of parameters from the wide model which are fixed in the narrow model, that is, the number of entries of inds0 which are zero.
focus	An R function with: • first argument named par, denoting a vector of parameters, of the same length as in wide model • other arguments defining alternative focuses. These are supplied through the ... argument to fic. In the built-in examples, there is an argument named X, denoting alternative covariate values. The required format is documented below.

The function should return the focus quantity of interest. If additional arguments are supplied which are vectors or matrices, e.g. `X`, then these are assumed to represent multiple focuses, and `focus` should return a vector giving the focus for `par` and each row of `X`. Otherwise `focus` should return a scalar giving the focus value at `par`.

Not required if `focus_deriv` is specified.

Alternatively, `focus` can be a character string naming a built-in focus function supplied by the **fic** package. Currently these include:

`"prob_logistic"`, the probability of the outcome in a logistic regression model

`"mean_normal"` the mean outcome in a normal linear regression model

See `focus_fns` for the functions underlying these built-in focuses.

`focus_deriv` Vector of partial derivatives of the focus function with respect to the parameters in the wide model. This is not usually needed, as it can generally be computed automatically and accurately from the function supplied in `focus`, using numerical differentiation.

`wt` Vector of weights to apply to different covariate values in `X`. This should have length equal to the number of alternative values for the covariates, that is, the number of alternative focuses of interest. The covariate-specific focused model comparison statistics are then supplemented by averaged statistics for a population defined by this distribution of covariate values. If this argument is omitted, the values are assumed to have equal weight when computing the average. The weights are not normalised, though the interpretation is unclear if the weights don't sum to one.

`sub` If `"auto"` (the default) then the submodels are fitted automatically within this function. If `NULL` they are not fitted, and focus estimates are not returned with the results.

`B` If `B` is 0 (the default) the standard analytic formulae for the focused model comparison statistics are used with mean square error loss. If `B`>0, then a parametric bootstrap method is used with `B` bootstrap samples, and the loss specified in the `loss` argument. More details of this approach are given in the package vignette "Focused model comparison with bootstrapping and alternative loss functions".

`loss` A function returning an estimated loss for a submodel estimate under the sampling distribution of the wide model. Only applicable when using bootstrapping. This should have two arguments `sub` and `wide`. `sub` should be a scalar giving the focus estimate from a submodel. `wide` should be a vector with a sample of focus estimates from the wide model, e.g. generated by a bootstrap method. By default this is a function calculating the root mean square error of the submodel estimate. An example is given in the vignette "Focused model comparison with bootstrapping and alternative loss functions".

`...` Other arguments to the focus function can be supplied here.

The built-in focus functions `prob_logistic` and `mean_normal` take an argument `X` giving covariate values defining the focus. This can either be a matrix or a vector, or a list or data frame that can be coerced into a matrix.

If just one focus is needed, then `X` can be a vector of length equal to the number of parameters in the wide model.

To compute focused model comparison statistics for multiple focuses defined by the same focus function evaluated at multiple covariate values, `X` should be a matrix, with number of columns equal to the number of parameters in the wide model, and number of rows equal to the number of alternative focuses.

For a typical regression model, the first parameter will denote an intercept, so the first value of `X` should be 1, and the remaining values should correspond to covariates whose coefficients form parameters of the wide model. See the examples in the vignette.

Arguments to the focus function other than `X` can also be supplied as a matrix, vector, list or data frame in the same way. An exception is when the argument is supplied as a vector, this is assumed to refer to multiple focuses. For example, suppose the focus function defines the quantile of a distribution, and takes an argument `focus_p`, then calling `fic(...,focus_p=c(0.1, 0.9))` indicates two alternative focuses defined by the 0.1 and 0.9 quantiles.

Details

The model parameters include the intercept, followed by the coefficients of any covariates. Any "dispersion" parameter is excluded from the parameters indicated by `inds` and `inds0`.

Only covariate selection problems are supported in this function. To compare between models with a fixed and unknown dispersion, `glm` would have to be replaced by maximum likelihood estimation routines written by hand, along the lines described in the "skew-normal models" vignette.

The focus function can however depend on the value of the dispersion parameter. The focus function should then have an argument called `dispersion`. See the example of a gamma GLM below, where the focus is the mean outcome for some covariate value.

Examples of covariate selection in logistic regression are given in the main package vignette.

Examples

```
# Gamma regression with one binary covariate

# Simulated data
set.seed(1)
simx <- rbinom(1000, 1, 0.5)
mean0 <- 1.1
simy <- rgamma(1000, shape=2, scale=exp(log(mean0) + 0.2*simx))

mod <- glm(simy ~ simx, family=Gamma(link="log"))

# Check the parameter estimates are close to true
# values used for simulation
(shape <- 1 / summary(mod)$dispersion)
coef(mod)[2] # log mean ratio associated with covariate. true value 0.2
exp(coef(mod)[1] - log(shape))  # mean with x=0, true value 1.1
exp(coef(mod)[1] + coef(mod)[2] - log(shape))  # mean with x=1

focus_mean <- function(par, X, dispersion){
  exp(X %*% par - log(1/dispersion))
}
```

```
X <- rbind("x0" = c(1,0), "x1" = c(1,1))
inds <- rbind("no_covariate"=c(1,0), "covariate"=c(1,1))

fic(mod, inds=inds, focus=focus_mean, X=X)

# The focus need not depend on X or the dispersion
focus_base_scale <- function(par, dispersion){
  exp(par[1])
}
fic(mod, inds=inds, focus=focus_base_scale)

# ...equivalently,
focus_base_scale <- function(par){
  exp(par[1])
}
fic(mod, inds=inds, focus=focus_base_scale)
```

`fic.lm` *Focused information criteria for linear models*

Description

Focused information criteria for linear models fitted with `lm`. Typically used to compare models with different covariates (more generally, different linear terms).

Usage

```
## S3 method for class 'lm'
fic(
  wide,
  inds,
  inds0 = NULL,
  gamma0 = 0,
  focus = NULL,
  focus_deriv = NULL,
  wt = NULL,
  sub = "auto",
  B = 0,
  loss = loss_mse,
  ...
)
```

Arguments

`wide`	Fitted model object containing the wide model.
`inds`	Matrix or vector of indicators for which parameters are included in the submodel or submodels to be assessed. A matrix should be supplied if there are multiple submodels. This should have number of rows equal to the number of submodels, and number of columns equal to the total number of parameters in the wide model. It contains 1s in the positions where the parameter is included in the submodel, and 0s in positions where the parameter is excluded. This should always be 1 in the positions defining the narrow model, as specified in `inds0`.
`inds0`	Vector of indicators specifying the narrow model, in the same format as `inds`. If this is omitted, the narrow model is assumed to be defined by the first row of `inds` (if `inds` is a matrix), or `inds` itself if this is a vector.
`gamma0`	Vector of special values taken by the parameters $gamma$ which define the narrow model. This defaults to 0, as in covariate selection, where "excluded" coefficients are fixed to 0. This should either be a scalar, assumed to be the same for all parameters fixed in the narrow model, or a vector of length equal to the number of parameters from the wide model which are fixed in the narrow model, that is, the number of entries of `inds0` which are zero.
`focus`	An R function with: • first argument named `par`, denoting a vector of parameters, of the same length as in wide model • other arguments defining alternative focuses. These are supplied through the `...` argument to `fic`. In the built-in examples, there is an argument named `X`, denoting alternative covariate values. The required format is documented below. The function should return the focus quantity of interest. If additional arguments are supplied which are vectors or matrices, e.g. `X`, then these are assumed to represent multiple focuses, and `focus` should return a vector giving the focus for `par` and each row of `X`. Otherwise `focus` should return a scalar giving the focus value at `par`. Not required if `focus_deriv` is specified. Alternatively, `focus` can be a character string naming a built-in focus function supplied by the **fic** package. Currently these include: `"prob_logistic"`, the probability of the outcome in a logistic regression model `"mean_normal"` the mean outcome in a normal linear regression model See `focus_fns` for the functions underlying these built-in focuses.
`focus_deriv`	Vector of partial derivatives of the focus function with respect to the parameters in the wide model. This is not usually needed, as it can generally be computed automatically and accurately from the function supplied in `focus`, using numerical differentiation.

wt — Vector of weights to apply to different covariate values in X. This should have length equal to the number of alternative values for the covariates, that is, the number of alternative focuses of interest. The covariate-specific focused model comparison statistics are then supplemented by averaged statistics for a population defined by this distribution of covariate values. If this argument is omitted, the values are assumed to have equal weight when computing the average. The weights are not normalised, though the interpretation is unclear if the weights don't sum to one.

sub — If "auto" (the default) then the submodels are fitted automatically within this function. If NULL they are not fitted, and focus estimates are not returned with the results.

The model parameters include the intercept, followed by the coefficients of any covariates. The standard deviation is excluded.

Only covariate selection problems are supported in this function. To compare between models with a fixed and unknown standard deviation, hand-written maximum likelihood estimation routines would be needed, along the lines described in the "skew-normal models" vignette.

The focus can depend on the standard deviation. The focus function should then have an argument sigma.

See the vignette "Using the fic R package for focused model comparison: linear regression" for some examples.

B — If B is 0 (the default) the standard analytic formulae for the focused model comparison statistics are used with mean square error loss. If B>0, then a parametric bootstrap method is used with B bootstrap samples, and the loss specified in the loss argument. More details of this approach are given in the package vignette "Focused model comparison with bootstrapping and alternative loss functions".

loss — A function returning an estimated loss for a submodel estimate under the sampling distribution of the wide model. Only applicable when using bootstrapping. This should have two arguments sub and wide. sub should be a scalar giving the focus estimate from a submodel. wide should be a vector with a sample of focus estimates from the wide model, e.g. generated by a bootstrap method. By default this is a function calculating the root mean square error of the submodel estimate. An example is given in the vignette "Focused model comparison with bootstrapping and alternative loss functions".

... — Other arguments to the focus function can be supplied here.

The built-in focus functions prob_logistic and mean_normal take an argument X giving covariate values defining the focus. This can either be a matrix or a vector, or a list or data frame that can be coerced into a matrix.

If just one focus is needed, then X can be a vector of length equal to the number of parameters in the wide model.

To compute focused model comparison statistics for multiple focuses defined by the same focus function evaluated at multiple covariate values, X should be a matrix, with number of columns equal to the number of parameters in the wide model, and number of rows equal to the number of alternative focuses.

For a typical regression model, the first parameter will denote an intercept, so the first value of X should be 1, and the remaining values should correspond

to covariates whose coefficients form parameters of the wide model. See the examples in the vignette.

Arguments to the focus function other than `X` can also be supplied as a matrix, vector, list or data frame in the same way. An exception is when the argument is supplied as a vector, this is assumed to refer to multiple focuses. For example, suppose the focus function defines the quantile of a distribution, and takes an argument `focus_p`, then calling `fic(...,focus_p=c(0.1, 0.9))` indicates two alternative focuses defined by the 0.1 and 0.9 quantiles.

Examples

```
## Covariate selection in Motor Trend cars data
## See the "fic" package vignette on linear models for more details

wide.lm <- lm(mpg ~ am + wt + qsec + disp + hp, data=mtcars)

## Select between all submodels
ncovs_wide <- length(coef(wide.lm)) - 1
inds0 <- c(1, rep(0, ncovs_wide))
inds <- all_inds(wide.lm, inds0)

## Two focuses: mean MPG for automatic and manual transmission,
## given mean values of the other covariates
cmeans <- colMeans(model.frame(wide.lm)[,c("wt","qsec","disp","hp")])
X <- rbind(
  "auto"   = c(intercept=1, am=0, cmeans),
  "manual" = c(intercept=1, am=1, cmeans)
)
ficres <- fic(wide.lm, inds=inds, focus=mean_normal, X=X)
summary(ficres)
ggplot_fic(ficres)
```

`fic.msm` *Focused information criteria for multi-state models for panel data*

Description

Focused information criteria for multi-state models fitted with `msm` from the **msm** package.

Usage

```
## S3 method for class 'msm'
fic(
  wide,
  inds,
  inds0 = NULL,
  gamma0 = 0,
```

```
  focus = NULL,
  focus_deriv = NULL,
  wt = NULL,
  sub = NULL,
  B = 0,
  loss = loss_mse,
  ...
)
```

Arguments

wide	Object returned by msm containing the wide model.
inds	Matrix or vector of indicators for which parameters are included in the submodel or submodels to be assessed. A matrix should be supplied if there are multiple submodels. This should have number of rows equal to the number of submodels, and number of columns equal to the total number of parameters in the wide model. It contains 1s in the positions where the parameter is included in the submodel, and 0s in positions where the parameter is excluded. This should always be 1 in the positions defining the narrow model, as specified in inds0.
inds0	Vector of indicators specifying the narrow model, in the same format as inds. If this is omitted, the narrow model is assumed to be defined by the first row of inds (if inds is a matrix), or inds itself if this is a vector.
gamma0	Vector of special values taken by the parameters $gamma$ which define the narrow model. This defaults to 0, as in covariate selection, where "excluded" coefficients are fixed to 0. This should either be a scalar, assumed to be the same for all parameters fixed in the narrow model, or a vector of length equal to the number of parameters from the wide model which are fixed in the narrow model, that is, the number of entries of inds0 which are zero.
focus	An R function with: • first argument named par, denoting a vector of parameters, of the same length as in wide model • other arguments defining alternative focuses. These are supplied through the ... argument to fic. In the built-in examples, there is an argument named X, denoting alternative covariate values. The required format is documented below. The function should return the focus quantity of interest. If additional arguments are supplied which are vectors or matrices, e.g. X, then these are assumed to represent multiple focuses, and focus should return a vector giving the focus for par and each row of X. Otherwise focus should return a scalar giving the focus value at par. Not required if focus_deriv is specified. Alternatively, focus can be a character string naming a built-in focus function supplied by the **fic** package. Currently these include:

"prob_logistic", the probability of the outcome in a logistic regression model

"mean_normal" the mean outcome in a normal linear regression model

See focus_fns for the functions underlying these built-in focuses.

focus_deriv — Vector of partial derivatives of the focus function with respect to the parameters in the wide model. This is not usually needed, as it can generally be computed automatically and accurately from the function supplied in focus, using numerical differentiation.

wt — Vector of weights to apply to different covariate values in X. This should have length equal to the number of alternative values for the covariates, that is, the number of alternative focuses of interest. The covariate-specific focused model comparison statistics are then supplemented by averaged statistics for a population defined by this distribution of covariate values. If this argument is omitted, the values are assumed to have equal weight when computing the average. The weights are not normalised, though the interpretation is unclear if the weights don't sum to one.

sub — List of objects returned by msm containing the submodels to be assessed. Optional. Only required if you want the estimate of the focus function under the submodel to be included in the results.

B — If B is 0 (the default) the standard analytic formulae for the focused model comparison statistics are used with mean square error loss. If B>0, then a parametric bootstrap method is used with B bootstrap samples, and the loss specified in the loss argument. More details of this approach are given in the package vignette "Focused model comparison with bootstrapping and alternative loss functions".

loss — A function returning an estimated loss for a submodel estimate under the sampling distribution of the wide model. Only applicable when using bootstrapping. This should have two arguments sub and wide. sub should be a scalar giving the focus estimate from a submodel. wide should be a vector with a sample of focus estimates from the wide model, e.g. generated by a bootstrap method. By default this is a function calculating the root mean square error of the submodel estimate. An example is given in the vignette "Focused model comparison with bootstrapping and alternative loss functions".

... — Other arguments to the focus function can be supplied here.

The built-in focus functions prob_logistic and mean_normal take an argument X giving covariate values defining the focus. This can either be a matrix or a vector, or a list or data frame that can be coerced into a matrix.

If just one focus is needed, then X can be a vector of length equal to the number of parameters in the wide model.

To compute focused model comparison statistics for multiple focuses defined by the same focus function evaluated at multiple covariate values, X should be a matrix, with number of columns equal to the number of parameters in the wide model, and number of rows equal to the number of alternative focuses.

For a typical regression model, the first parameter will denote an intercept, so the first value of X should be 1, and the remaining values should correspond to covariates whose coefficients form parameters of the wide model. See the examples in the vignette.

Arguments to the focus function other than `X` can also be supplied as a matrix, vector, list or data frame in the same way. An exception is when the argument is supplied as a vector, this is assumed to refer to multiple focuses. For example, suppose the focus function defines the quantile of a distribution, and takes an argument `focus_p`, then calling `fic(...,focus_p=c(0.1, 0.9))` indicates two alternative focuses defined by the 0.1 and 0.9 quantiles.

Details

This might be used for covariate selection, or comparing models with different constraints on the covariate effects or intensities. An example is given in the **fic** package vignette "Examples of focused model comparison: multi-state models". Note in particular in this example how the parameters are ordered in the `inds` argument, and how the various **msm** output functions can be used as focuses.

Examples

```
## Covariate selection in psoriatic arthritis model.
## See the "fic" package vignette on multi-state models for
## more details and examples.

if (requireNamespace("msm",quietly=TRUE)){

Qind <- rbind(c(0, 1, 0, 0),
              c(0, 0, 1, 0),
              c(0, 0, 0, 1),
              c(0, 0, 0, 0))
psor.wide.msm <- msm::msm(state ~ months, subject=ptnum, data=msm::psor,
                     qmatrix = Qind,  gen.inits=TRUE,
                     covariates = ~ollwsdrt+hieffusn)
inds <- rbind(
    c(1,1,1,0,0,0,0,0,0),
    c(1,1,1,0,0,0,0,0,1),
    c(1,1,1,0,0,0,0,1,1),
    c(1,1,1,0,0,0,1,1,1),
    c(1,1,1,0,0,1,1,1,1),
    c(1,1,1,0,1,1,1,1,1),
    c(1,1,1,1,1,1,1,1,1)
)
focus_tlos <- function(par){
    x.new <- msm::updatepars.msm(psor.wide.msm, par)
    msm::totlos.msm(x.new, covariates=0, tot=10)["State 4"]
}
fres <- fic(wide=psor.wide.msm, inds=inds, focus=focus_tlos)
fres

}
```

`fic.survreg` *Focused information criteria for parametric survival models*

Description

Focused information criteria for parametric survival models fitted with the **survival** package.

Usage

```
## S3 method for class 'survreg'
fic(
  wide,
  inds,
  inds0 = NULL,
  gamma0 = 0,
  focus = NULL,
  focus_deriv = NULL,
  wt = NULL,
  sub = NULL,
  B = 0,
  loss = loss_mse,
  ...
)
```

Arguments

`wide`	Object returned by the `survreg` function, containing the wide model.
`inds`	Matrix or vector of indicators for which parameters are included in the submodel or submodels to be assessed. A matrix should be supplied if there are multiple submodels. This should have number of rows equal to the number of submodels, and number of columns equal to the total number of parameters in the wide model. It contains 1s in the positions where the parameter is included in the submodel, and 0s in positions where the parameter is excluded. This should always be 1 in the positions defining the narrow model, as specified in `inds0`.
`inds0`	Vector of indicators specifying the narrow model, in the same format as `inds`. If this is omitted, the narrow model is assumed to be defined by the first row of `inds` (if `inds` is a matrix), or `inds` itself if this is a vector.
`gamma0`	Vector of special values taken by the parameters $gamma$ which define the narrow model. This defaults to 0, as in covariate selection, where "excluded" coefficients are fixed to 0. This should either be a scalar, assumed to be the same for all parameters fixed in the narrow model, or a vector of length equal to the number of parameters from the wide model which are fixed in the narrow model, that is, the number of entries of `inds0` which are zero.

`focus` An R function with:

- first argument named `par`, denoting a vector of parameters, of the same length as in wide model
- other arguments defining alternative focuses. These are supplied through the `...` argument to `fic`. In the built-in examples, there is an argument named `X`, denoting alternative covariate values. The required format is documented below.

The function should return the focus quantity of interest. If additional arguments are supplied which are vectors or matrices, e.g. `X`, then these are assumed to represent multiple focuses, and `focus` should return a vector giving the focus for `par` and each row of `X`. Otherwise `focus` should return a scalar giving the focus value at `par`.

Not required if `focus_deriv` is specified.

Alternatively, `focus` can be a character string naming a built-in focus function supplied by the **fic** package. Currently these include:

`"prob_logistic"`, the probability of the outcome in a logistic regression model

`"mean_normal"` the mean outcome in a normal linear regression model

See `focus_fns` for the functions underlying these built-in focuses.

`focus_deriv` Vector of partial derivatives of the focus function with respect to the parameters in the wide model. This is not usually needed, as it can generally be computed automatically and accurately from the function supplied in `focus`, using numerical differentiation.

`wt` Vector of weights to apply to different covariate values in `X`. This should have length equal to the number of alternative values for the covariates, that is, the number of alternative focuses of interest. The covariate-specific focused model comparison statistics are then supplemented by averaged statistics for a population defined by this distribution of covariate values. If this argument is omitted, the values are assumed to have equal weight when computing the average. The weights are not normalised, though the interpretation is unclear if the weights don't sum to one.

`sub` List of fitted model objects of class `survreg` containing the submodels to be assessed. Optional. Only required if you want the estimate of the focus function under the submodels to be included in the results.

`B` If `B` is 0 (the default) the standard analytic formulae for the focused model comparison statistics are used with mean square error loss. If `B`>0, then a parametric bootstrap method is used with `B` bootstrap samples, and the loss specified in the `loss` argument. More details of this approach are given in the package vignette "Focused model comparison with bootstrapping and alternative loss functions".

`loss` A function returning an estimated loss for a submodel estimate under the sampling distribution of the wide model. Only applicable when using bootstrapping. This should have two arguments `sub` and `wide`. `sub` should be a scalar giving the focus estimate from a submodel. `wide` should be a vector with a sample of focus estimates from the wide model, e.g. generated by a bootstrap method. By default this is a function calculating the root mean square error of the submodel estimate. An example is given in the vignette "Focused model comparison with bootstrapping and alternative loss functions".

`...` Other arguments to the focus function can be supplied here.

The built-in focus functions `prob_logistic` and `mean_normal` take an argument `X` giving covariate values defining the focus. This can either be a matrix or a vector, or a list or data frame that can be coerced into a matrix.

If just one focus is needed, then `X` can be a vector of length equal to the number of parameters in the wide model.

To compute focused model comparison statistics for multiple focuses defined by the same focus function evaluated at multiple covariate values, `X` should be a matrix, with number of columns equal to the number of parameters in the wide model, and number of rows equal to the number of alternative focuses.

For a typical regression model, the first parameter will denote an intercept, so the first value of `X` should be 1, and the remaining values should correspond to covariates whose coefficients form parameters of the wide model. See the examples in the vignette.

Arguments to the focus function other than `X` can also be supplied as a matrix, vector, list or data frame in the same way. An exception is when the argument is supplied as a vector, this is assumed to refer to multiple focuses. For example, suppose the focus function defines the quantile of a distribution, and takes an argument `focus_p`, then calling `fic(...,focus_p=c(0.1, 0.9))` indicates two alternative focuses defined by the 0.1 and 0.9 quantiles.

Details

Any situation where all models being compared are special cases of a single "wide" model are supported. Examples include covariate selection, selection between models for the baseline hazard/survival with different levels of flexibility (e.g. comparing exponential and Weibull). An example of the latter is in the **fic** package vignette "Examples of focused model comparison: parametric survival models".

Parameters `par` of the focus function should be on the scale reported by the `icoef` component of the results of `survreg`, that is, with any positive-valued parameters log transformed.

Examples

```
library(survival)

## Fit exponential and Weibull models and plot fitted survival curves
ex <-  survreg(Surv(futime, fustat) ~ 1, data=ovarian, dist="exponential")
we <-  survreg(Surv(futime, fustat) ~ 1, data=ovarian, dist="weibull")

## Plot fitted survival curves, highlighting 1 year survival
plot(survfit(Surv(futime, fustat) ~ 1, data=ovarian))
t <- seq(0, 1200)
lines(t, pweibull(q=t, shape=exp(we$icoef[2]),
                  scale=exp(we$icoef[1]), lower.tail=FALSE))
lines(t, pexp(q=t, rate=1/exp(ex$icoef[1]), lower.tail=FALSE), lty=2)
abline(v=365, col="gray")

## Focused model comparison for focus of 1-year survival probability
indmat <- rbind(exp    = c(1,0),
```

```
                weib   = c(1,1))
surv1yr <- function(par){
    pweibull(q=365, shape=exp(par[2]), scale=exp(par[1]), lower.tail=FALSE)
}
fic(we, inds=indmat, focus=surv1yr, sub=list(ex, we))

## Exponential model has lower expected error, given such a small dataset
```

`fic_core` *Focused information criteria: core calculation functions*

Description

Core FIC calculation functions underlying the user interface in `fic`. `fic_core` just handles one submodel, while `fic_multi` can assess multiple submodels of the same wide model. For `fic_multi`, `inds` and `parsub` can be matrices with one row per submodel, while for `fic_core` they must be vectors.

Usage

```
fic_core(par, J, inds, inds0, gamma0 = 0, n, focus_deriv = NULL)

fic_multi(
  par,
  J,
  inds,
  inds0,
  gamma0 = 0,
  n,
  focus = NULL,
  focus_deriv = NULL,
  wt = NULL,
  parsub = NULL,
  auxpar = NULL,
  auxsub = NULL,
  ...
)
```

Arguments

`par`	Vector of maximum likelihood estimates from the wide model
`J`	Information matrix from the wide model, evaluated at the maximum likelihood estimates (note that this definition differs from Claeskens and Hjort, where `J` is defined as the information divided by the sample size `n`)

`inds` Matrix or vector of indicators for which parameters are included in the submodel or submodels to be assessed.

A matrix should be supplied if there are multiple submodels. This should have number of rows equal to the number of submodels, and number of columns equal to the total number of parameters in the wide model. It contains 1s in the positions where the parameter is included in the submodel, and 0s in positions where the parameter is excluded. This should always be 1 in the positions defining the narrow model, as specified in `inds0`.

`inds0` Vector of indicators specifying the narrow model, in the same format as `inds`. If this is omitted, the narrow model is assumed to be defined by the first row of `inds` (if `inds` is a matrix), or `inds` itself if this is a vector.

`gamma0` Vector of special values taken by the parameters $gamma$ which define the narrow model.

This defaults to 0, as in covariate selection, where "excluded" coefficients are fixed to 0.

This should either be a scalar, assumed to be the same for all parameters fixed in the narrow model, or a vector of length equal to the number of parameters from the wide model which are fixed in the narrow model, that is, the number of entries of `inds0` which are zero.

`n` Number of observations in the data used to fit the wide model.

`focus_deriv` Vector of partial derivatives of the focus function with respect to the parameters in the wide model. This is required by `fic_core`.

If there are multiple submodels, this should be a matrix with number of rows equal to the number of submodels, and number of columns equal to the number of parameters in the wide model. If there is a single submodel, this should be a vector with number of columns equal to the number of parameters in the wide model.

This should take the value given by `gamma0` in positions where the parameter is excluded from the submodel. For example, coefficients of covariates should take the value 0 if the covariate is excluded.

`focus` An R function with:

- first argument named `par`, denoting a vector of parameters, of the same length as in wide model
- other arguments defining alternative focuses. These are supplied through the `...` argument to `fic`. In the built-in examples, there is an argument named `X`, denoting alternative covariate values. The required format is documented below.

The function should return the focus quantity of interest. If additional arguments are supplied which are vectors or matrices, e.g. `X`, then these are assumed to represent multiple focuses, and `focus` should return a vector giving the focus for `par` and each row of `X`. Otherwise `focus` should return a scalar giving the focus value at `par`.

Not required if `focus_deriv` is specified.

Alternatively, `focus` can be a character string naming a built-in focus function supplied by the **fic** package. Currently these include:

"prob_logistic", the probability of the outcome in a logistic regression model

"mean_normal" the mean outcome in a normal linear regression model

See focus_fns for the functions underlying these built-in focuses.

wt — Vector of weights to apply to different covariate values in X. This should have length equal to the number of alternative values for the covariates, that is, the number of alternative focuses of interest. The covariate-specific focused model comparison statistics are then supplemented by averaged statistics for a population defined by this distribution of covariate values. If this argument is omitted, the values are assumed to have equal weight when computing the average. The weights are not normalised, though the interpretation is unclear if the weights don't sum to one.

parsub — Vector of maximum likelihood estimates from the submodel, or a matrix if there are multiple submodels. Only required to return the estimate of the focus quantity alongside the model assessment statistics for the submodel. If omitted, the estimate is omitted.

auxpar — Estimates of auxiliary parameters from the wide model. The only built-in example is the dispersion parameter for GLMs.

auxsub — List of estimates of auxiliary parameters from the submodel. The only built-in example is the dispersion parameter for GLMs.

... — Other arguments to the focus function can be supplied here.

The built-in focus functions prob_logistic and mean_normal take an argument X giving covariate values defining the focus. This can either be a matrix or a vector, or a list or data frame that can be coerced into a matrix.

If just one focus is needed, then X can be a vector of length equal to the number of parameters in the wide model.

To compute focused model comparison statistics for multiple focuses defined by the same focus function evaluated at multiple covariate values, X should be a matrix, with number of columns equal to the number of parameters in the wide model, and number of rows equal to the number of alternative focuses.

For a typical regression model, the first parameter will denote an intercept, so the first value of X should be 1, and the remaining values should correspond to covariates whose coefficients form parameters of the wide model. See the examples in the vignette.

Arguments to the focus function other than X can also be supplied as a matrix, vector, list or data frame in the same way. An exception is when the argument is supplied as a vector, this is assumed to refer to multiple focuses. For example, suppose the focus function defines the quantile of a distribution, and takes an argument focus_p, then calling `fic(...,focus_p=c(0.1, 0.9))` indicates two alternative focuses defined by the 0.1 and 0.9 quantiles.

Value

See fic

See Also

fic

Examples

```
## Lower-level implementation of the example in the main vignette

wide.glm <- glm(low ~ lwtkg + age + smoke + ht + ui + smokeage + smokeui,
                data=birthwt, family=binomial)
mod1.glm <- glm(low ~ lwtkg + age + smoke, data=birthwt, family=binomial)
inds0 <- c(1,1,0,0,0,0,0,0)
inds1 <- c(1,1,1,1,0,0,0,0)
focus_plogis <- function(par, X)plogis(X %*% par)
vals.smoke <-    c(1, 58.24, 22.95, 1, 0, 0, 22.95, 0)
vals.nonsmoke <- c(1, 59.50, 23.43, 0, 0, 0, 0, 0)
X <- rbind(vals.smoke, vals.nonsmoke)
par <- coef(wide.glm)
n <- nrow(birthwt)
J <- solve(vcov(wide.glm))
fic_multi(par=par, J=J,  inds=inds1, inds0=inds0, n=n, focus="prob_logistic",
          X=X, parsub=c(coef(mod1.glm), 0, 0, 0, 0))

## Even lower-level implementation, requiring derivatives of the focus
## These are available analytically in this example, but would normally
## need to be calculated using numerical differentiation

focus_deriv <- prob_logistic_deriv(par=par, X=X)
fic_core(par=par, J=J, inds=inds1, inds0=inds0, gamma0=0, n=n,
         focus_deriv=focus_deriv)
```

`fit_submodels` *Fit submodels of a general linear wide model, defined by a matrix of indicators for inclusion of covariates*

Description

Fit the submodels of a wide model `wide` which are defined by `inds`. This can only be used for covariate selection problems, where the submodels contain different subsets of covariates.

Usage

```
fit_submodels(wide, inds, ...)
```

Arguments

`wide`	Fitted model object containing the wide model.
`inds`	Matrix or vector of indicators for which parameters are included in the submodel or submodels to be assessed. A matrix should be supplied if there are multiple submodels. This should have number of rows equal to the number of submodels, and number of columns equal to the total number of parameters in the wide model. It contains 1s in

the positions where the parameter is included in the submodel, and 0s in positions where the parameter is excluded. This should always be 1 in the positions defining the narrow model, as specified in `inds0`.

`...` Other arguments to the focus function can be supplied here.

The built-in focus functions `prob_logistic` and `mean_normal` take an argument `X` giving covariate values defining the focus. This can either be a matrix or a vector, or a list or data frame that can be coerced into a matrix.

If just one focus is needed, then `X` can be a vector of length equal to the number of parameters in the wide model.

To compute focused model comparison statistics for multiple focuses defined by the same focus function evaluated at multiple covariate values, `X` should be a matrix, with number of columns equal to the number of parameters in the wide model, and number of rows equal to the number of alternative focuses.

For a typical regression model, the first parameter will denote an intercept, so the first value of `X` should be 1, and the remaining values should correspond to covariates whose coefficients form parameters of the wide model. See the examples in the vignette.

Arguments to the focus function other than `X` can also be supplied as a matrix, vector, list or data frame in the same way. An exception is when the argument is supplied as a vector, this is assumed to refer to multiple focuses. For example, suppose the focus function defines the quantile of a distribution, and takes an argument `focus_p`, then calling `fic(...,focus_p=c(0.1, 0.9))` indicates two alternative focuses defined by the 0.1 and 0.9 quantiles.

Details

Requires `wide` to have a component named `call` giving the function call used to produce `wide`. This call should include a `formula` component, which this function updates in order to define and fit the submodel. This should work for most standard linear-type models in common R packages.

Value

List of all fitted submodel objects.

Examples

```
bwt.glm <- glm(low ~ lwtkg + age + smoke,
               data=birthwt, family="binomial")
inds <- rbind(c(1,1,1,0), c(1,1,0,0))
fit_submodels(bwt.glm, inds=inds)
```

focus_fns *Built-in focus functions and their derivatives*

Description

Built-in focus functions and their derivatives

Usage

```
prob_logistic(par, X)

prob_logistic_deriv(par, X)

mean_normal(par, X)

mean_normal_deriv(par, X)
```

Arguments

par	Vector of parameter estimates, including the intercept.
X	Vector or matrix of covariate values, including the intercept. This can either be a vector of length p, or a nxp matrix, where p is the number of covariate effects, and n is the number of alternative sets of covariate values at which the focus function is to be evaluated.

Value

`prob_logistic` returns the probability of the outcome in a logistic regression model, and `mean_normal` returns the mean outcome in a normal linear regression. The `_deriv` functions return the vector of partial derivatives of the focus with respect to each parameter (or matrix, if there are multiple foci).

See Also

fic

Examples

```
## Model and focus from the main vignette
wide.glm <- glm(low ~ lwtkg + age + smoke + ht + ui +
                smokeage + smokeui, data=birthwt, family=binomial)
vals.smoke <-    c(1, 58.24, 22.95, 1, 0, 0, 22.95, 0)
vals.nonsmoke <- c(1, 59.50, 23.43, 0, 0, 0, 0, 0)
X <- rbind("Smokers" = vals.smoke, "Non-smokers" = vals.nonsmoke)
prob_logistic(coef(wide.glm), X=X)
prob_logistic_deriv(coef(wide.glm), X=X)


## Mean mpg for a particular covariate category in the Motor Trend data
```

```
## See the "fic" linear models vignette for more detail
wide.lm <- lm(mpg ~ am + wt + qsec + disp + hp, data=mtcars)
cmeans <- colMeans(model.frame(wide.lm)[,c("wt","qsec","disp","hp")])
X <- rbind(
  "auto"   = c(intercept=1, am=0, cmeans),
  "manual" = c(intercept=1, am=1, cmeans)
)
mean_normal(coef(wide.lm), X)
mean_normal_deriv(coef(wide.lm), X)
```

`get_H0` *Interpolate cumulative hazard function from a fitted Cox model*

Description

Returns the baseline cumulative hazard, at the requested times, from a Cox model fitted by coxph. Linear interpolation is used, assuming the hazard is piecewise constant, thus the cumulative hazard is piecewise linear.

Usage

```
get_H0(H0, t)
```

Arguments

H0	output from basehaz, containing estimates of the baseline cumulative hazard at a series of times.
t	vector of times for which cumulative hazard estimates are required.

Details

This does not extrapolate. If t is outside the observed event times, then NA will be returned.

Value

Fitted cumulative hazard at t.

Examples

```
library(survival)
wide <- coxph(Surv(years, death==1) ~ sex + thick_centred +
              infilt + epith + ulcer + depth + age, data=melanoma)
basehaz(wide)
get_H0(basehaz(wide), c(0,1,5,10,100))
```

ggplot_fic *Plot focused model comparison statistics: ggplot2 method*

Description

This only works if the focus estimates are available. The focus estimates are plotted against the root MSE. One plot is made for each covariate value defining different focuses. If the wide model estimate is available, this is illustrated as a solid line on the plot, and if the narrow model estimate is available, this is showm as a dashed line.

Usage

```
ggplot_fic(
  x,
  ci = TRUE,
  adj = TRUE,
  legend = TRUE,
  ylab = NULL,
  xlab = NULL,
  xlim = NULL,
  ylim = NULL
)
```

Arguments

x	Output from fic.
ci	Plot interval estimates? (TRUE or FALSE). These are calculated as plus / minus twice the standard error of the submodel focus under the wide model. These are rough estimates of uncertainty intended to illustrate the bias-variance tradeoff, and exclude any uncertainty associated with the choice between models.
adj	The optimal model is the one with the lowest root mean square error (RMSE). If adj=TRUE the RMSE is based on the adjusted bias estimator. Otherwise the standard estimator is used.
legend	Show a legend, identifying a) the line types for the wide and narrow models b) the names of the terms of the wide model. This is used when the inds object supplied to fic was constructed by all_inds, so has row names made out of a string of 0s and 1s that identify the terms included in the submodel. These strings are plotted as text labels against the estimate for each submodel. The legend identifies which 0s and 1s correspond to which model terms.
ylab	y-axis label.
xlab	x-axis label.
xlim	x-axis limits (pair of numbers)
ylim	y-axis limits

See Also

plot.fic, summary.fic

Examples

```
## Example from the main vignette, see there for more details

wide.glm <- glm(low ~ lwtkg + age + smoke + ht + ui + smokeage + smokeui,
                data=birthwt, family=binomial)
vals.smoke <-    c(1, 58.24, 22.95, 1, 0, 0, 22.95, 0)
vals.nonsmoke <- c(1, 59.50, 23.43, 0, 0, 0, 0, 0)
X <- rbind("Smokers" = vals.smoke, "Non-smokers" = vals.nonsmoke)
inds0 <- c(1,1,0,0,0,0,0,0)
combs <- all_inds(wide.glm, inds0)
ficres <- fic(wide = wide.glm, inds = combs, inds0 = inds0,
              focus = prob_logistic, X = X)
ggplot_fic(ficres)
summary(ficres)
```

`melanoma` *Malignant melanoma survival data*

Description

Data originally analysed by Andersen et al (1993) on the survival of 205 patients in Denmark with malignant melanoma, and used by Claeskens and Hjort (2008) to illustrate focused model selection in Cox regression.

Usage

```
melanoma
```

Format

A data frame with 205 rows and the following columns:

ptno Patient identification number

death Survival status: 1 = dead from illness, 2 = censored, 4 = dead from other causes

days Survival time in days

depth Invasion depth: factor with levels 1, 2, 3

infilt Infection infiltration level, a measure of resistance to the tumour: factor with levels 1 (high resistance), 2, 3, 4 (low resistance)

epith Indicator for epithelioid cells present

ulcer Indicator for ulceration

thick Thickness of the tumour in 1/100 mm

sex Sex. Factor with levels `"female"`,`"male"` and reference level `"female"`

age Age in years

years Survival time in years (instead of days)

thick_centred Version of `thick` centred around its mean and rescaled, defined as (`thick` - 292)/100.

Source

The supporting material from Claeskens and Hjort (2008), at https://feb.kuleuven.be/public/u0043181/modelselection/datasets/melanoma_data.txt. Versions of this dataset are also given in the **MASS** and **boot** packages.

References

Claeskens, G., & Hjort, N. L. (2008). Model selection and model averaging (Vol. 330). Cambridge: Cambridge University Press.

Andersen, P. K., Borgan, O., Gill, R. D., & Keiding, N. (2012). Statistical models based on counting processes. Springer.

`newdata_to_X` *Convert data frame of covariate values to a design matrix*

Description

Convert data frame of covariate values to a design matrix

Usage

```
newdata_to_X(newdata, wide, intercept = TRUE)
```

Arguments

`newdata`	Data frame where each row is a vector of covariate values defining an alternative focus quantity.
`wide`	Wide model which includes these covariates.
`intercept`	Include an intercept as the first column.

Details

Numeric values can be supplied for factor levels that are character strings denoting numbers (like `"1"` or `"2"`).

Value

"Design" matrix of covariate values defining alternative focuses, with factors expanded to their contrasts. This is in the form required by the `X` argument of `fic`, with one row per alternative focus. The columns correspond to coefficients in a linear-type model. For the built-in focus functions such as `mean_normal` and `prob_logistic`, these coefficients include an intercept, but user-written focuses may be written in such a way as not to require an intercept (as in the example in the "skew normal" vignette).

Examples

```
bwt.glm <- glm(low ~ lwtkg + age + smoke + ftv, data=birthwt, family="binomial")
newdata <- data.frame(lwtkg=1, age=60, smoke=0, ftv="2+")
newdata_to_X(newdata, bwt.glm)

## See the Cox regression section of the main package vignette for another example.
```

---

plot.fic *Plot focused model comparison statistics: base graphics method*

Description

Plot focused model comparison statistics: base graphics method

Usage

```
## S3 method for class 'fic'
plot(
  x,
  ci = TRUE,
  adj = TRUE,
  xlab = NULL,
  ylab = NULL,
  xlim = NULL,
  ylim = NULL,
  pch = 19,
  mfrow = NULL,
  ...
)
```

Arguments

`x`	Output from `fic`.
`ci`	Plot interval estimates? (`TRUE` or `FALSE`). These are calculated as plus / minus twice the standard error of the submodel focus under the wide model. These are rough estimates of uncertainty intended to illustrate the bias-variance tradeoff, and exclude any uncertainty associated with the choice between models.

adj	The optimal model is the one with the lowest root mean square error (RMSE). If adj=TRUE the RMSE is based on the adjusted bias estimator. Otherwise the standard estimator is used.
xlab	x-axis label.
ylab	y-axis label.
xlim	x-axis limits (pair of numbers)
ylim	y-axis limits
pch	Plot point character, by default 19 (solid circle).
mfrow	Vector of two numbers giving the number of rows and number of columns respectively in the plot grid, if there are multiple focuses.
...	Other options to pass to plot.

Details

If the focus estimates are available, then the focus estimates are plotted against the root MSE. One plot is made for each covariate value defining different focuses. If the wide model estimate is available, this is illustrated as a solid line on the plot, and if the narrow model estimate is available, this is shown as a dashed line.

If the focus estimates are unavailable, then the standard errors of the focus estimate are plotted against the corresponding bias. The plot points are shaded with darkness proportional to the RMSE, with the point of maximum RMSE in black.

The **ggplot2**-based plot method, ggplot_fic, is slightly nicer.

See Also

ggplot_fic, summary.fic

Examples

```
## Example from the main vignette, see there for more details

wide.glm <- glm(low ~ lwtkg + age + smoke + ht + ui + smokeage + smokeui,
                data=birthwt, family=binomial)
vals.smoke <-    c(1, 58.24, 22.95, 1, 0, 0, 22.95, 0)
vals.nonsmoke <- c(1, 59.50, 23.43, 0, 0, 0, 0, 0)
X <- rbind("Smokers" = vals.smoke, "Non-smokers" = vals.nonsmoke)
inds0 <- c(1,1,0,0,0,0,0,0)
combs <- all_inds(wide.glm, inds0)
ficres <- fic(wide = wide.glm, inds = combs, inds0 = inds0,
              focus = prob_logistic, X = X)
plot(ficres)
```

summary.fic *Summarise focused model comparison results*

Description

Summarise focused model comparison results

Usage

```
## S3 method for class 'fic'
summary(object, tidy = TRUE, adj = FALSE, ...)
```

Arguments

object	Object returned by fic representing focused model comparison statistics for a range of models, and potentially also multiple focus quantities.
tidy	If TRUE (the default) then the results describing the optimal model (per focus) are returned as a data frame, with the names of the parameters in the optimal model collapsed into a single string. If FALSE, the results are returned as a list, including a vector of parameter names.
adj	The optimal model is the one with the lowest root mean square error (RMSE). If adj=TRUE the RMSE is based on the adjusted bias estimator. Otherwise the standard estimator is used.
...	Other arguments, currently unused.

Value

A list of two components, one for the optimal model per focus, and one for the range of focus and RMSE estimates over models.

See Also

ggplot_fic, plot.fic for a more detailed visual representation of the focused comparison

Examples

```
## Example from the main vignette, see there for more details

wide.glm <- glm(low ~ lwtkg + age + smoke + ht + ui + smokeage + smokeui,
                data=birthwt, family=binomial)
vals.smoke <-    c(1, 58.24, 22.95, 1, 0, 0, 22.95, 0)
vals.nonsmoke <- c(1, 59.50, 23.43, 0, 0, 0, 0, 0)
X <- rbind("Smokers" = vals.smoke, "Non-smokers" = vals.nonsmoke)
inds0 <- c(1,1,0,0,0,0,0,0)
combs <- all_inds(wide.glm, inds0)
ficres <- fic(wide = wide.glm, inds = combs, inds0 = inds0,
              focus = prob_logistic, X = X)
ggplot_fic(ficres)
```

```
summary(ficres)
```

Index