Package 'expertsurv'

Title Incorporate Expert Opinion with Parametric Survival Models

Version 1.2.0

Description Enables users to incorporate expert opinion with parametric survival analysis using a Bayesian or frequentist approach. Expert Opinion can be provided on the survival probabilities at certain time-point(s) or for the difference in mean survival between two treatment arms.Please reference its use as Cooney, P., White, A. (2023) <doi:10.1177/0272989X221150212>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Biarch true

Depends R (>= 3.5.0), survival

Imports abind, dplyr, flexsurv (>= 2.0), ggplot2, loo, magrittr (>= 2.0), methods, Rcpp (>= 0.12.0), RcppParallel (>= 5.0.1), Rdpack, rlang, rms, rstan (>= 2.18.1), rstantools (>= 2.2.0), scales, SHELF, stats, stringr, tibble, tidyselect, utils

LinkingTo BH (>= 1.66.0), Rcpp (>= 0.12.0), RcppEigen (>= 0.3.3.3.0), RcppParallel (>= 5.0.1), rstan (>= 2.18.1), StanHeaders (>= 2.18.0)

SystemRequirements GNU make, C++17

Suggests rmarkdown, knitr, rjags, R2jags, xlsx, shiny, shinyWidgets, shinycssloaders, shinyjs, shinyMatrix, shinybusy, ggpubr

VignetteBuilder knitr

RdMacros Rdpack

NeedsCompilation yes

Author Philip Cooney [aut, cre] (<https://orcid.org/0000-0001-9649-0195>),
Arthur White [ths] (<https://orcid.org/0000-0002-7268-5163>)

Maintainer Philip Cooney <phcooney@tcd.ie>

Repository CRAN

Date/Publication 2023-08-16 13:44:37 UTC

R topics documented:

---

expertsurv-package   *Incorporating Expert Opinion with Parametric Survival Models*

Description

Contains functions to include expert opinion with the parametric models commonly used in health economic modelling. Theoretical details are described elsewhere (Cooney and White 2023). Borrows many function from the survHE package (Baio 2020).

Details

Package:	expertsurv
Type:	Package
Version:	1.0.0
Date:	2023-01-27
License:	MIT + file LICENSE
LazyLoad:	yes

Integrate expert opinions on survival and mean differences in survival with common parametric survival models using either a Bayesian or frequentist framework.

Author(s)

Philip Cooney Package Creator, Maintainer

Arthur White Thesis Supervisor

References

P Cooney (2023). expertsurv: Incorporating expert opinion into parametric survival models.

Baio G (2020). "survHE: Survival Analysis for Health Economic Evaluation and Cost-Effectiveness Modeling." *Journal of Statistical Software*, **95**(14), 1–47. doi:10.18637/jss.v095.i14.

Cooney P, White A (2023). "Direct Incorporation of Expert Opinion into Parametric Survival Models to Inform Survival Extrapolation." *Medical Decision Making*, **1**(1), 0272989X221150212. doi:10.1177/0272989X221150212, PMID: 36647200, https://doi.org/10.1177/0272989X221150212 , https://doi.org/10.1177/0272989X221150212.

Examples

```
#Define expert opinion
require("dplyr")
param_expert_example1 <- list()
#1 timepoint and 2 experts with equal weight,
#first a normal distribution, second a non-standard t-distribution with
#3 degrees of freedom

param_expert_example1[[1]] <- data.frame(dist = c("norm","t"),
                             wi = c(0.5,0.5), # Ensure Weights sum to 1
                             param1 = c(0.1,0.12),
                             param2 = c(0.05,0.05),
                              param3 = c(NA,3))


timepoint_expert <- 14

data2 <- data %>% rename(status = censored) %>% mutate(time2 = ifelse(time > 10, 10, time),
                 status2 = ifelse(time> 10, 0, status))

example1  <- fit.models.expert(formula=Surv(time2,status2)~1,data=data2,
                             distr=c("wph", "gomp"),
                             method="mle",
                             pool_type = "log pool",
                             opinion_type = "survival",
                             times_expert = timepoint_expert,
                             param_expert = param_expert_example1)

 #Visualize the goodness of fit
 model.fit.plot(example1, type = "aic")
 #Visualize the survival curve
 plot(example1, add.km = TRUE, t = 0:30)
```

---

cred_int   *Credible interval for pooled distribution*

Description

Returns the interval based on defined quantiles. The approach used only provides an approximate (although quite accurate) integral.

Usage

```
cred_int(plt_obj, val = "linear pool", interval = c(0.025, 0.975))
```

Arguments

plt_obj	A plot object from `plot_expert_opinion`.
val	The name of the opinion for which the interval will be generated.
interval	A vector of the upper and lower probabilities. Default is the standard 95% interval

Value

Credible interval based on the pooled distribution

Examples

```
param_expert_example1 <- list()
param_expert_example1[[1]] <- data.frame(dist = c("norm","t"),
    wi = c(0.5,0.5), # Ensure Weights sum to 1
    param1 = c(0.1,0.12),
    param2 = c(0.005,0.005),
    param3 = c(NA,3))
plot_opinion1<- plot_expert_opinion(param_expert_example1[[1]],
            weights = param_expert_example1[[1]]$wi)
cred_int(plot_opinion1,val = "linear pool", interval = c(0.025, 0.975))
```

---

data   *A fictional survival trial taken directly from survHE.*

Description

A dataset containing fictional data from a trial, where the main outcome is in terms of time-to-event and censoring indicator and with additional covariates.

Usage

```
data
```

Format

A data frame with 367 rows and 8 variables:

ID_patient The individual level identifier

time The observed time at which the event happens

censored An indicator to describe whether the event is fully observed or censored

arm An indicator for the treatment arm, with 0 = control and 1 = active treatment

sex An indicator for the individual's sex, with 0 = male and 1 = female

age A numeric variable with the individual's age

imd A categorical variable representing a measure of area-level social deprivation

ethnic A categorical variable representing the individual's ethnic group, as measured from a Census

References

Baio G (2020). "survHE: Survival Analysis for Health Economic Evaluation and Cost-Effectiveness Modeling." *Journal of Statistical Software*, **95**(14), 1–47. doi:10.18637/jss.v095.i14.

---

elicit_surv	*Elicit survival judgements interactively and estimate survival models*

Description

Opens up a web browser (using the shiny package), from which you can specify judgements and fit distributions for multiple timepoints and experts. Plots of the fitted density functions are provided overlayed on the survival data (where appropriate).

Usage

```
elicit_surv()
```

Details

Once the elicitation is complete the analysis can be run. Click "Download R objects" to download the expertsurv object generated from the analysis. Click "Download report" to generate a report including plots and parameter values for the parametric survival models.

Author(s)

Philip Cooney phcooney@tcd.ie

Examples

```
## Not run:

elicit_surv()


## End(Not run)
```

---

fit.models.expert   *Fitting Parametric Survival models with Expert Opinion*

Description

Implementation of survival models with expert opinion on the survival probabilities or expected difference in survival. Function is equivalent to the `fit.models` in `survHE` except for the inclusion of the "expert_type" and "param_expert" arguments. Worked examples can be found in the README file. Note that the default method is "hmc", however, the user may use "mle" (method "inla" is not included).

Usage

```
fit.models.expert(
  formula = NULL,
  data,
  distr = NULL,
  method = "hmc",
  expert_type = "survival",
  param_expert = NULL,
  ...
)
```

Arguments

formula	As per `fit.models` in `survHE`
data	As per `fit.models` in `survHE`
distr	As per `fit.models` in `survHE`. Note Generalized F model is not available for method = "hmc" nor Royston-Parmar available with opinion on the mean survival.
method	As per `fit.models` in `survHE`. (except for the inla method). It should be noted that a few of the models are fit using JAGS, however, for consistency we still use "hmc".
expert_type	Either "survival", which indicates expert opinion on the survival function or "mean" (actually anything that does not contain "survival") which represents a belief on difference in survival.
param_expert	A list containing a dataframe for each timepoint (if applicable). Each dataframe should have columns with the following names and each row representing an expert:

   - **dist**: Names of the distribution assigned to each expert which may be "norm", "t", "lnorm", "gamma", "beta".
   - **wi**: Weight of the expert, must sum to 1.
   - **param1**: First parameter of the distribution (e.g. mean for norm distribution). Parameters as per `SHELF` package.
   - **param2**: Second parameter of the distribution.

- **param3**: Third parameter of the distribution (NA expect for degrees of freedom for t distribution)

...   Other arguments may be required depending on the example. See README for details.

Value

An object of class `expertsurv` which contains the parameters of the models estimated with expert opinion.

Examples

```
require("dplyr")
#Expert Opinion as a normal distribution centered on 0.1 with sd 0.005
param_expert_example1 <- list()
param_expert_example1[[1]] <- data.frame(dist = c("norm"),
                                          wi = c(1), # Ensure Weights sum to 1
                                          param1 = c(0.1),
                                          param2 = c(0.05),
                                          param3 = c(NA))
timepoint_expert <- 14 # Expert opinion at t = 14

data2 <- data %>% rename(status = censored) %>% mutate(time2 = ifelse(time > 10, 10, time),
status2 = ifelse(time> 10, 0, status))

example1  <- fit.models.expert(formula=Surv(time2,status2)~1,data=data2,
                               distr=c("wei", "gomp"),
                               method="mle",
                               opinion_type = "survival",
                               times_expert = timepoint_expert,
                               param_expert = param_expert_example1)

plot(example1, add.km = TRUE, t = seq(0:20)) #Plot Survival
model.fit.plot(example1, type = "aic")  #Plot AIC
```

---

make.surv   *Engine for Probabilistic Sensitivity Analysis on the survival curves*

Description

Creates the survival curves for the fitted model(s) - Original code from `survHE`

Usage

```
make.surv(fit, mod = 1, t = NULL, newdata = NULL, nsim = 1, ...)
```

Arguments

`fit`	the result of the call to the `fit.models` function, containing the model fitting (and other relevant information)
`mod`	the index of the model. Default value is 1, but the user can choose which model fit to visualise, if the call to fit.models has a vector argument for distr (so many models are fitted & stored in the same object)
`t`	the time vector to be used for the estimation of the survival curve
`newdata`	a list (of lists), specifiying the values of the covariates at which the computation is performed. For example `list(list(arm=0),list(arm=1))` will create two survival curves, one obtained by setting the covariate `arm` to the value 0 and the other by setting it to the value 1. In line with `flexsurv` notation, the user needs to either specify the value for *all* the covariates or for none (in which case, `newdata=NULL`, which is the default). If some value is specified and at least one of the covariates is continuous, then a single survival curve will be computed in correspondence of the average values of all the covariates (including the factors, which in this case are expanded into indicators).
`nsim`	The number of simulations from the distribution of the survival curves. Default at `nsim=1`, in which case uses the point estimate for the relevant distributional parameters and computes the resulting survival curve
`...`	Additional options

Value

A list with survival times for the fitted models

Author(s)

Gianluca Baio

References

Baio G (2020). "survHE: Survival Analysis for Health Economic Evaluation and Cost-Effectiveness Modeling." *Journal of Statistical Software*, **95**(14), 1–47. doi:10.18637/jss.v095.i14.

See Also

`psa.plot` (for example)

`model.fit.plot`	*Graphical representation of the measures of model fitting based on Information Criteria*

Description

Plots a summary of the model fit for all the models fitted.

Usage

```
model.fit.plot(..., type = "dic")
```

Arguments

...	Optional inputs. Must include an expertsurv object.
type	should the DIC, WAIC, PML be plotted (AIC, BIC also allowed but only valid for frequentist approach).

Value

A plot with the relevant model fitting statistics plotted in order of fit.

Examples

```
require("dplyr")
param_expert_example1 <- list()
param_expert_example1[[1]] <- data.frame(dist = c("norm"),
                                        wi = c(1), # Ensure Weights sum to 1
                                        param1 = c(0.1),
                                        param2 = c(0.05),
                                        param3 = c(NA))
timepoint_expert <- 14 # Expert opinion at t = 14


data2 <- expertsurv::data %>% rename(status = censored) %>%
mutate(time2 = ifelse(time > 10, 10, time),
status2 = ifelse(time> 10, 0, status))
example1  <- fit.models.expert(formula=Surv(time2,status2)~1,data=data2,
                            distr=c("wei", "gomp"),
                            method="mle",
                            pool_type = "linear pool",
                            opinion_type = "survival",
                            times_expert = timepoint_expert,
                            param_expert = param_expert_example1)


model.fit.plot(example1, type = "aic")
```

---

plot.expertsurv	*Plot survival curves for the models fitted using* fit.models

---

Description

Plots the results of model fit.

Usage

```
## S3 method for class 'expertsurv'
plot(...)
```

Arguments

... Must include at least one result object saved as the call to the `fit.models` function. Nay include other optional parameters. These include whether the KM curve should be added `add.km` and whether the user specifies a profile of covariates (in the list `newdata`). Other possibilities are additional (mainly graphical) options. These are: `xlab` = a string with the label for the x-axis (default = "time") `ylab` = a string with the label for the y-axis (default = "Survival") `lab.profile` = a (vector of) string(s) indicating the labels associated with the strata defining the different survival curves to plot. Default to the value used by the Kaplan Meier estimate given in `fit.models`. `newdata` = a list (of lists) providing the values for the relevant covariates If NULL, then will use the mean values for the covariates if at least one is a continuous variable, or the combination of the categorical covariates. `xlim` = a vector determining the limits for the x-axis `colors` = a vector of characters defining the colours in which to plot the different survival curves `lab.profile` = a vector of characters defining the names of the models fitted `add.km` = TRUE (whether to also add the Kaplan Meier estimates of the data) `annotate` = FALSE (whether to also add text to highlight the observed vs extrapolated data) `legend.position` = a vector of proportions to place the legend. Default to 'c(.75,.9)', which means 75% across the x-axis and 90% across the y-axis `legend.title` = suitable instructions to format the title of the legend; defaults to 'element_text(size=15,face="bold")' but there may be other arguments that can be added (using 'ggplot' facilities) `legend.text` = suitable instructions to format the text of the legend; defaults to 'element_text(colour="black", size=14, face="plain")' but there may be other arguments that can be added (using 'ggplot' facilities)

Value

A ggplot2 object of the survival curves.

Author(s)

Gianluca Baio

References

Baio G (2020). "survHE: Survival Analysis for Health Economic Evaluation and Cost-Effectiveness Modeling." *Journal of Statistical Software*, **95**(14), 1–47. doi:10.18637/jss.v095.i14.

See Also

`fit.models`, `write.surv`

Examples

```
require("dplyr")
param_expert_example1 <- list()
param_expert_example1[[1]] <- data.frame(dist = c("norm","t"),
                                         wi = c(0.5,0.5), # Ensure Weights sum to 1
                                         param1 = c(0.1,0.12),
                                         param2 = c(0.15,0.5),
                                         param3 = c(NA,3))

timepoint_expert <- 14
data2 <- data %>% rename(status = censored) %>% mutate(time2 = ifelse(time > 10, 10, time),
                                             status2 = ifelse(time> 10, 0, status))
example1_mle <- fit.models.expert(formula=Surv(time2,status2)~1,data=data2,
                                  distr=c("wph", "exp"),
                                  method="mle",
                                  pool_type = "log pool",
                                 opinion_type = "survival",
                                  times_expert = timepoint_expert,
                                  param_expert = param_expert_example1)

#Warning! 50 iterations is far too few, however, it has been done so that the
#example can be run for CRAN
example1_bayes <- fit.models.expert(formula=Surv(time2,status2)~1,data=data2,
                                    distr=c("wph", "exp"),
                                    method="hmc",
                                    iter = 50,
                                    pool_type = "log pool",
                                    opinion_type = "survival",
                                    times_expert = timepoint_expert,
                                    param_expert = param_expert_example1)

plot(MLE=example1_mle,Bayesian=example1_bayes, add.km = TRUE, t = 0:30)
```

---

plot_expert_opinion   *Plotting Pooled Expert Opinion*

Description

Returns a ggplot with the individual expert opinions along with the pooled distributions (both linear and logarithmic).

Usage

```
plot_expert_opinion(
  object,
  xl_plt = NULL,
  xu_plt = NULL,
  weights = NULL,
```

```
    St_indic = 0
)
```

Arguments

`object`	Either a object of class elicitation (from SHELF) or a dataframe with parameters of the distribution (see Example below).
`xl_plt`	Optionally set the lower bound for the plot
`xu_plt`	Optionally set the upper bound for the plot
`weights`	A vector with the weight of each expert. If omitted, set to equal weights.
`St_indic`	Set to 1 if you want to truncate the distributions to be between 0 and 1.

Value

A ggplot with pooled distributions.

Examples

```
expert_df <- data.frame(dist = c("norm","t"), #Distribution Name
                        wi = c(1/3,2/3), #Expert weights
                        param1 = c(0.3,0.40), #Parameter 1
                        param2 = c(0.05,0.05),# Parameter 2
                        param3 = c(NA,3)) #Parameter 3: Only t-distribution
plot_expert_opinion(expert_df , weights = expert_df$wi)
```

`print.expertsurv`   *Print a summary of the survival model(s) fitted by* `fit.models`

Description

Prints the summary table for the model(s) fitted, with the estimate of the parameters - ported from `survHE`.

Usage

```
## S3 method for class 'expertsurv'
print(x, mod = 1, ...)
```

Arguments

`x`	the `expertsurv` object (the output of the call to `fit.models`)
`mod`	is the index of the model. Default value is 1, but the user can choose which model fit to visualise, if the call to fit.models has a vector argument for distr (so many models are fitted & stored in the same object)
`...`	additional options, including: `digits` = number of significant digits to be shown in the summary table (default = 6) `original` = a flag to say whether the *original* table from either `flexsurv` or `rstan`/`JAGS` should be printed

Value

Printed message (no object returned) providing estimates of the survival models.

Author(s)

Gianluca Baio

References

Baio G (2020). "survHE: Survival Analysis for Health Economic Evaluation and Cost-Effectiveness Modeling." *Journal of Statistical Software*, **95**(14), 1–47. doi:10.18637/jss.v095.i14.

Examples

```
require("dplyr")
param_expert_example1 <- list()
param_expert_example1[[1]] <- data.frame(dist = c("norm","t"),
                                         wi = c(0.5,0.5), # Ensure Weights sum to 1
                                       param1 = c(0.1,0.12),
                                      param2 = c(0.15,0.5),
                                       param3 = c(NA,3))
timepoint_expert <- 14
data2 <- data %>% rename(status = censored) %>% mutate(time2 = ifelse(time > 10, 10, time),
                                                  status2 = ifelse(time> 10, 0, status))
mle = example1 <- fit.models.expert(formula=Surv(time2,status2)~1,data=data2,
                  distr=c("wph", "gomp"),
                  method="mle",
                  pool_type = "log pool",
                  opinion_type = "survival",
                  times_expert = timepoint_expert,
                  param_expert = param_expert_example1)
print(mle)
```

---

psa.plot	*Graphical depiction of the probabilistic sensitivity analysis for the survival curves - ported from* survHE

Description

Plots the survival curves for all the PSA simulations. The function is actually deprecated - similar graphs can be obtained directly using the plot method (with options), which allows a finer depiction of the results.

Usage

```
psa.plot(psa, ...)
```

Arguments

psa   the result of the call to the function `make.surv`

...   Optional graphical parameters, such as: `xlab` = label for the x-axis `ylab` = label
                        for the y-axis `col` = (vector) of colors for the lines to be plotted `alpha` = the
                        level of transparency for the curves (default = 0.2)

Value

ggplot2 object of the survival curve including parameter uncertainty

Author(s)

Gianluca Baio

References

Baio G (2020). "survHE: Survival Analysis for Health Economic Evaluation and Cost-Effectiveness
Modeling." *Journal of Statistical Software*, **95**(14), 1–47. doi:10.18637/jss.v095.i14.

Examples

```
require("dplyr")
param_expert_example1 <- list()
param_expert_example1[[1]] <- data.frame(dist = c("norm","t"),
                                         wi = c(0.5,0.5), # Ensure Weights sum to 1
                                         param1 = c(0.1,0.12),
                                         param2 = c(0.15,0.5),
                                         param3 = c(NA,3))

timepoint_expert <- 14
data2 <- data %>% rename(status = censored) %>% mutate(time2 = ifelse(time > 10, 10, time),
                                                 status2 = ifelse(time> 10, 0, status))
example1 <- fit.models.expert(formula=Surv(time2,status2)~1,data=data2,
                         distr=c("wph", "gomp"),
                         method="mle",
                         pool_type = "log pool",
                         opinion_type = "survival",
                         times_expert = timepoint_expert,
                         param_expert = param_expert_example1)

p.mle = make.surv(example1,mod= 2,t = 1:30, nsim=1000) #Plot the Gompertz model
psa.plot(p.mle , name_labs = "PSA", labs = "Gompertz", col ="blue")
```

---

summary.expertsurv	*Prints a summary table for the distribution the mean survival time for a given model and data*

Description

Calculates the mean survival time as the area under the survival curve - ported from `survHE`

Usage

```
## S3 method for class 'expertsurv'
summary(object, mod = 1, t = NULL, nsim = 1000, ...)
```

Arguments

object	a survHE object (resulting from the call to `fit.models`
mod	the model to be analysed (default = 1)
t	the vector of times to be used in the computation. Default = NULL, which means the observed times will be used. NB: the vector of times should be: i) long enough so that S(t) goes to 0; and ii) dense enough so that the approximation to the AUC is sufficiently precise
nsim	the number of simulations from the survival curve distributions to be used (to compute interval estimates)
...	Additional options

Details

A list comprising of the following elements

Value

mean.surv	A matrix with the simulated values for the mean survival times
tab	A summary table

Author(s)

Gianluca Baio

References

Baio G (2020). "survHE: Survival Analysis for Health Economic Evaluation and Cost-Effectiveness Modeling." *Journal of Statistical Software*, **95**(14), 1–47. doi:10.18637/jss.v095.i14.

See Also

`fit.models.expert`, `make.surv`

Index