# Package 'JMR'

July 8, 2021

**Title** Recurrent Event Survival Analysis and Joint Modeling

**Version** 0.0.0.9000

**Description** While the main focus of this package is recurrent event data analysis and joint modeling, it also includes fitting regression models for right censored (non-recurrent) time-to-event data. Available options are Weibull, log-logistic, log-normal, exponentiated Weibull and generalized gamma accelerated failure time models, and Weibull and generalized log-logistic proportional hazards models. This package also includes residual analysis, data simulation and dynamic predictions. Maximum likelihood method is used to fit recurrent event models, whereas Bayesian approach (implemented in Stan) is used for joint modeling. The rstan package is used to parse, compile, test, estimate and analyze Stan models.

**License** GPL (>= 2)

**Encoding** UTF-8

**LazyData** true

**Biarch** true

**Depends** R (>= 3.4.0),
rstan (>= 2.18.1),
bayesplot (>= 1.7.1),
coda (>= 0.19-3),
numDeriv (>= 2016.8-1.1),
splines (>= 3.6.1),
nlme (>= 3.1-147),
survival (>= 3.1-12),
Rmpfr (>= 0.8-1)

**Imports** methods,
Rcpp (>= 0.12.0),
rstan (>= 2.18.1),
rstantools (>= 2.0.0),
ggplot2 (>= 3.3.0),
splines (>= 3.6.1),
survival (>= 3.1-12),
nlme (>= 3.1-147),
numDeriv (>= 2016.8-1.1),
Rmpfr (>= 0.8-1)

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

**SystemRequirements** GNU make

**RoxygenNote** 7.1.1

## R topics documented:

---

`JMR-package` *The 'JMR' package.*

---

### Description

While the main focus of this package involves recurrent event data analysis and joint modeling, it also includes fitting regression models for right censored (non-recurrent) time-to-event data. For the time-to-event process, available options are Weibull, log-logistic, log-normal, exponentiated Weibull and generalized gamma accelerated failure time models, and Weibull and generalized log-logistic proportional hazards models. This package also includes several utility functions, including functions for residual analysis, data simulation and dynamic predictions. Maximum likelihood method is used to fit recurrent event models, whereas Bayesian approach (implemented in Stan) is used for joint modeling. The 'rstan' package is used to parse, compile, test, estimate and analyze Stan models.

### Author(s)

Shahedul Khan <khan@math.usask.ca>

## References

Cook RJ and Lawless J, The statistical analysis of recurrent events, Springer, 2007.

Cox DR and Oakes D, Analysis of survival data, Chapman and Hall/CRC, 1984.

Gabry J and Mahr T, bayesplot: Plotting for Bayesian Models, R package version 1.7.2, https://mc-stan.org/bayesplot.

Gelman A, Hwang J, and Vehtari A, Understanding predictive information criteria for Bayesian models, Statistics and Computing, 24: 997-1016, 2014.

Henderson R, Diggle P, and Dobson A, Joint modelling of longitudinal measurements and event time data, Biostatistics, 1: 465-480, 2000.

Kalbfleisch JD and Prentice RL, The statistical analysis of failure time data, Wiley, 2002.

Rizopoulos D, JMbayes: Joint Modeling of Longitudinal and Time-to-Event Data under a Bayesian Approach, R package version 0.8-85, 2020, https://cran.r-project.org/web/packages/JMbayes.

Rizopoulos D, The R package JMbayes for fitting joint models for longitudinal and time-to-event data Using MCMC, Journal of Statistical Software, 72(7): 1-45, 2016.

Rizopoulos D, Joint Models for Longitudinal and Time-to-Event Data: With Applications in R, Chapman and Hall/CRC, 2012.

Rizopoulos D and Ghosh P, A Bayesian semiparametric multivariate joint model for multiple longitudinal outcomes and a time-to-event, Statistics in Medicine, 30: 1366-1380, 2011.

Stan Development Team, Stan Modeling Language Users Guide and Reference Manual, 2020, Version https://mc-stan.org.

Stan Development Team, RStan: the R interface to Stan, 2020, Version http://mc-stan.org/.

Therneau T and Grambsch P, Modeling survival data: extending the Cox Model, Springer-Verlag, 2000.

Zhou H and Hanson T, A unified framework for fitting Bayesian semiparametric models to arbitrarily censored survival data, including spatially referenced data, Journal of the American Statistical Association, 113(522), 571-581, 2018.

---

## `etime.reg` *Fit a parametric time-to-event regression model*

---

## Description

Fit a parametric time-to-event regression model for right censored data using the maximum likelihood method. It includes both standard (non-recurrent) survival data analysis and recurrent event data analysis. Available options are Weibull, log-logistic, log-normal, exponentiated Weibull and generalized gamma accelerated failure time (AFT) models, and Weibull and generalized log-logistic proportional hazards (PH) models.

## Usage

```
etime.reg(
  Formula,
  init = NULL,
  Data,
  surv.model = "weibull",
  conf.int = 0.95,
  iter.max = 250
)
```

## Arguments

`Formula` a formula of the form `response ~ model`. The `response` is `c(st,status)` for standard survival analysis with right censored data, and `c(start,stop,status)` for recurrent event data analysis, where

- `st`: survival time for right censored data,
- `status`: censoring indicator,
- `start` and `stop`: the start and end time of each time interval for recurrent event data.

The linear predictor is speficified by the `model` argument of the Formula expression. It consists of a series of terms (covariates) separated by + operators. It also allows `:` and `*` operators to define interaction terms.

- Example 1: `c(st,status) ~ z1 + z2 + z3`
- Example 2: `c(start,stop,status) ~ z1 + z2 * z3`

`init` initial values for the parameters (optional). It is a matrix with each row has one set of initial values. If there are $p$ regression coefficients $\beta_1$, $\beta_2$, ..., $\beta_p$, then each row has initial values with the following sequence: $\beta_1$, $\beta_2$, ..., $\beta_p$, log($\kappa$), log($\gamma$), log($\rho$) for the exponentiated Weibull, generalized gamma and generalized log-logistic models, and $\beta_1$, $\beta_2$, ..., $\beta_p$, log($\kappa$), log($\rho$) for the Weibull, log-logistic and log-normal models, where $\kappa$ and $\gamma$ are the shape parameters and $\rho$ is the rate parameter (see below). That is, multiple sets of initial values can be given. For example, for the exponentiated Weibull model, `init = rbind(c(rep(0,p),rep(0,3)),c(rep(0,p),rep(0.5,3)))` gives two sets of initial values.

`Data` a data frame in which to interpret the variables named in the Formula.

`surv.model` assumed distribution for survival times. Available options are given as follows.

- AFT Models: `"weibull"`, `"lnormal"`, `"llogistic"`, `"eweibull"` and `"ggamma"` for Weibull, log-normal, log-logistic, exponentiated Weibull and generalized gamma distributions, respectively.
- PH Models: `"weibullph"` and `"gllph"` for Weibull and generalized log-logistic distributions, respectively.

Default is `"weibull"`. The probability density functions of these distributions are:

- Weibull: $f(t) = \kappa\rho(\rho t)^{\kappa-1}\,\exp\{-(\rho t)^{\kappa}\}$
- Log-logistic: $f(t) = \kappa\rho(\rho t)^{\kappa-1}/[1+(\rho t)^{\kappa}]^2$
- Log-normal: $f(t) = \kappa\,\exp\{-(\kappa\log(\rho t))^2/2\}/[t(2\pi)^{1/2}]$
- Generalized gamma: $f(t) = \kappa\rho(\rho t)^{\kappa\gamma-1}\,\exp\{-(\rho t)^{\kappa}\}/\Gamma(\gamma)$
- Exponentiated Weibull: $f(t) = \kappa\gamma\rho(\rho t)^{\kappa-1}(1-\exp\{-(\rho t)^{\kappa}\})^{\gamma-1}\,\exp\{-(\rho t)^{\kappa}\}$
- Generalized log-logistic: $f(t) = \kappa\rho(\rho t)^{\kappa-1}/[1+(\gamma t)^{\kappa}]^{(\rho/\gamma)^{\kappa}+1}$

`conf.int` confidence level (default is 0.95).

`iter.max` maximum number of iterations for optimization (default is 250).

## Details

The AFT and the PH models are specified using the hazard functions $h(t;z) = h_0[t\,\exp(-z'\beta)]\,\exp(-z'\beta)$ and $h(t;z) = h_0(t)\,\exp(z'\beta)$, respectively, where $h_0(.)$ is the baseline hazard function of the assumed distribution, $z$ is the $p$ x 1 vector of covariates, and $\beta$ is the corresponding vector of regression coefficients (see Section 2.3 of Kalbfleisch and Prentice, and Section 3.2 of Cook and Lawless). The baseline hazard functions of the distributions are:

- Weibull: $h(t) = \kappa\rho(\rho t)^{\kappa-1}$
- Log-logistic: $h(t) = \kappa\rho(\rho t)^{\kappa-1}/[1 + (\rho t)^{\kappa}]$
- Log-normal: $h(t) = f(t)/S(t)$, whete $S(t)$ is the survivor function of the log-normal distribution
- Generalized gamma: $h(t) = f(t)/S(t)$, whete $S(t)$ is the survivor function of the generalized gamma distribution
- Exponentiated Weibull: $h(t) = \kappa\gamma\rho(\rho t)^{\kappa-1}(1-\exp\{-(\rho t)^{\kappa}\})^{\gamma-1}\exp\{-(\rho t)^{\kappa}\}/[1-(1-\exp\{-(\rho t)^{\kappa}\})^{\gamma}]$
- Generalized log-logistic: $h(t) = \kappa\rho(\rho t)^{\kappa-1}/[1 + (\gamma t)^{\kappa}]$

For recurrent event analysis, each line of data for a given subject must include the start time and stop time for each interval of follow-up.

### Value

`model`: the survival model.

`data summary`: number of observations, number of events, number of censored observations, and number of predictors.

`fit`: estimate, standard error, z, p value, and confidence interval.

`exp(coef)` and `exp(-est)`: estimate and confidence interval.

`fit criteria`: log-likelihood, deviance, and AIC.

`optimizer`: nlminb or optim.

`cov`: covariance matrix.

`st`: survival times (for recurrent event, an n x 2 matrix for start and stop times).

`status`: censoring indicator.

`design.mat`: design matrix.

### Author(s)

Shahedul Khan <khan@math.usask.ca>

### References

Cook RJ and Lawless J, The statistical analysis of recurrent events, Springer, 2007.

Kalbfleisch JD and Prentice RL, The statistical analysis of failure time data, Wiley, 2002.

### See Also

`LR.test`, `etime.resid`, `survreg`, `coxph`.

### Examples

```
# Example 1: Recurrent Event Analysis
  library(frailtypack)
  data(readmission)
  fit.gg <- etime.reg(c(t.start, t.stop, event) ~ sex + chemo + charlson,
      surv.model = "ggamma",  Data = readmission)
  fit.gg
  fit.gg$cov
```

```
# Example 2: Non-recurrent Data (Right Censored)
  library(survival)
  fit.ew <- etime.reg(c(time, status) ~ karno + diagtime + age + prior + celltype + trt,
      Data = veteran, surv.model = "eweibull")
  fit.ew
  fit.ew$design.mat
```

## `etime.resid` *Cox-Snell residuals*

### Description

Cox-Snell residuals of the `etime.reg` fit.

### Usage

```
etime.resid(
  fit,
  plot = FALSE,
  conf.int = 0.95,
  xlim = NULL,
  ylim = NULL,
  xlab = NULL,
  ylab = NULL,
  main = NULL
)
```

### Arguments

`fit` etime.reg fit.

`plot` returns residual plot if `TRUE` (default is `FALSE`).

`conf.int` confidence level for the pointwise confidence intervals of the residuals (if `plot = TRUE`).

`xlim` x limits (a vector of length 2 of the form `c(minimum,maximum)`) of the plot (optional).

`ylim` y limits (a vector of length 2 of the form `c(minimum,maximum)`) of the plot (optional).

`main` main title (character expression) of the plot (optional).

### Details

If `plot = TRUE`, this function returns both residuals and a residual plot. The plot shows the unit slop line (in grey), residuals (solid circles) and pointwise confidence intervals (dashed lines). The plot of the Cox-Snell (hazard-based) residuals should be roughly a straight line with unit slope when the model is adequate. See Section 6.2 of Lawless and Section 3.7.3 of Cook and Lawless for details.

### Value

`residuals` Cox-Snell residuals.

`plot` residual plot if `plot=TRUE`.

### Author(s)

Shahedul Khan <khan@math.usask.ca>

### References

Cook RJ and Lawless J, The statistical analysis of recurrent events, Springer, 2007.

Lawless J, Statistical models and methods for lifetime data, Wiley, 2003.

### See Also

`etime.reg`, `LR.test`.

### Examples

```
# Example 1: Recurrent Event Analysis
  library(frailtypack)
  data(readmission)
  fit.gg <- etime.reg(c(t.start, t.stop, event) ~ sex + chemo + charlson,
      surv.model = "ggamma", Data = readmission)
  fit.wph <- etime.reg(c(t.start, t.stop, event) ~ sex + chemo + charlson,
      surv.model = "weibullph", Data = readmission)
  etime.resid(fit.wph, plot = TRUE)
  par(mfrow = c(1, 2))
  etime.resid(fit.wph, plot = TRUE, xlim = c(0, 5.25),
    ylim = c(0, 5.25),  main = "Weibull PH")
  etime.resid(fit.gg, plot = TRUE, xlim = c(0, 5.25),
     ylim = c(0, 5.25),  main = "Generalized Gamma AFT")

# Example 2: Non-recurrent Data (Right Censored)
  library(survival)
  fit.ln <- etime.reg(c(time, status) ~ karno + diagtime + age + prior + celltype + trt,
      Data = veteran, surv.model = "lnormal")
  etime.resid(fit.ln, plot = TRUE, main = "Log-normal AFT")
```

---

## `jm.icriteria` *Information criteria (DIC and WAIC)*

---

### Description

Computes DIC and WAIC for a Bayesian joint model fit. It uses a `jm.reg` fit as its argument.

### Usage

```
jm.icriteria(jmfit, posterior.mean = TRUE)
```

### Arguments

| | |
|---|---|
| `jmfit` | a `jm.reg` fit with `method = "MCMC"`. |
| `posterior.mean` | if `TRUE` (default), the posterior mean is used to compute DIC, otherwise posterior median is used. Note that this argument is used only for DIC calculation. |

## Details

This function uses the posterior draws from class "stanfit". See Gelman et al. (2014) for a description of DIC and WAIC from a Bayesian perspective.

## Value

`DIC`: the values of log predictive density (lpd) or log-likelihood given a point estimate of the fitted model, effective number of parameters (p), and DIC. If `posterior.mean = TRUE`, the posterior mean is used as a point estimate, otherwise posterior median is used.

`WAIC`: the values of log pointwise predictive density (lppd) evaluated using the posterior simulations, effective number of parameters (p), and WAIC.

## Author(s)

Shahedul Khan <khan@math.usask.ca>

## References

Gelman A, Hwang J, and Vehtari A, Understanding predictive information criteria for Bayesian models, Statistics and Computing, 24: 997-1016, 2014.

## See Also

`jm.plots`, `jm.reg`, `jm.resid`

## Examples

```
# Example: pbc data
lme.fit <- lme(log(bilirubin) ~ drug + ns(futime, 2),
  data = pbc.long, random = ~ ns(futime, 2) | id)
surv.fit <- coxph(Surv(st, status2) ~ drug * age, data = pbc.surv, x = TRUE)
jmfit.ll2 <- jm.reg(surv.fit = surv.fit, lme.fit = lme.fit,
    surv.model = "llogistic", timevar = "futime", form = "riz")
jm.icriteria(jmfit.ll2)
jmfit.wph2 <- jm.reg(surv.fit = surv.fit, lme.fit = lme.fit,
    surv.model = "weibullph", timevar = "futime", form = "riz")
jm.icriteria(jmfit.wph2)
```

# `jm.lppredict` *Marginal predictions for the longitudinal outcome of a joint model*

## Description

Given a `jm.reg` fit and a data frame, this function produces marginal predictions (population-level) for the longitudinal outcome.

## Usage

```
jm.lppredict(jmfit, newdata)
```

## Arguments

jmfit
: a jm.reg fit with method = "MCMC".

newdata
: a data frame in which to look for variables with which to predict. The data frame must have columns that correspond by name to the original regressors used in the longitudinal part of the joint model fit.

## Details

Posterior summaries of X(s)$\alpha$ (marginal predictions) are obtained from the MCMC samples, where X(s) is the fixed-effects design matrix and $\alpha$ is the fixed-effects coefficient vector for the longitudinal model.

## Value

This function returns posterior summaries of X(s)$\alpha$ (marginal predictions).

## Author(s)

Shahedul Khan <khan@math.usask.ca>

## See Also

jm.lspredict, jm.surv, jm.reg

## Examples

```
# Example: pbc data
lme.fit <- lme(log(bilirubin) ~ drug + ns(futime, 2),
  data = pbc.long, random = ~ ns(futime, 2) | id)
surv.fit <- coxph(Surv(st, status2) ~ drug * age, data = pbc.surv, x = TRUE)
jmfit.gg2 <-  jm.reg(surv.fit = surv.fit, lme.fit = lme.fit,
    surv.model = "ggamma", timevar = "futime", form = "riz")
jm.summary(jmfit.gg2)
newdata <-  expand.grid(futime = seq(min(pbc.long$futime), max(pbc.long$futime),
    len = 25), drug = 0:1)
pred <- jm.lppredict(jmfit = jmfit.gg2, newdata = newdata)
pred
# Plot of the predicted values
futime <- pred[[1]][, 1]
post.mean <- pred[[1]][, 2]
lower <- pred[[1]][, 4]
upper <- pred[[1]][, 8]
drug <- factor(newdata$drug, levels = c(0, 1), labels = c("placebo", "D-penicillamine"))
# Use the xyplot function from package 'lattice'
library(lattice)
xyplot(post.mean + lower + upper ~ futime | drug,
  type = "l", col = rep(1, 3), lty = c(1, 3, 3),
  lwd = 2,  ylab = "log(bilirubin)",  xlab = "futime")
```

## `jm.lspredict` *Subject-specific predictions for the longitudinal outcome of a joint model*

### Description

Given a `jm.reg` fit and a data frame, this function produces subject-specific predictions for the longitudinal outcome.

### Usage

```
jm.lspredict(
  jmfit,
  newdata,
  n.sample = NULL,
  ptime = NULL,
  plot = TRUE,
  posterior.mean = TRUE,
  xlim = NULL,
  ylim = NULL,
  xlab = NULL,
  ylab = NULL,
  main = NULL,
  adapt_delta = 0.9,
  max_treedepth = 15,
  warmup = 200,
  seed = sample.int(.Machine$integer.max, 1)
)
```

### Arguments

`jmfit` a `jm.reg` fit with `method = "MCMC"`.

`newdata` a data frame in which to look for variables with which to predict. The data frame must have columns that correspond by name to the original variables used to fit the joint model.

`n.sample` an integer denoting how many MCMC samples to use. Default is `min(200,n.iter)`, where n.iter is the number of MCMC iterations (for all chains combined).

`ptime` a numeric vector of times at which predictions are to be computed. If `NULL`, `ptime = seq(ptime0,floor(max.st),length=25)` is used, where `ptime0` is the last time the subject provided a longitudinal measurement, and `max.st` is `floor(max(event times))`.

`plot` if `TRUE`, a plot of the predictions is produced.

`posterior.mean` if `TRUE`, posterior means of the predictions are used in the plot, otherwise posterior medians are used.

`xlim` x limits (a vector of length 2 of the form `c(minimum,maximum)`) of the plot (optional).

`ylim` y limits (a vector of length 2 of the form `c(minimum,maximum)`) of the plot (optional).

`xlab` a label (character expression) for the x axis (optional).

| | |
|---|---|
| `ylab` | a label (character expression) for the y axis (optional). |
| `main` | main title (character expression) of the plot (optional). |
| `adapt_delta` | the target average proposal acceptance probability during Stan's adaptation period. |
| `max_treedepth` | a positive integer specifying the maximum treedepth. |
| `warmup` | a positive integer specifying the number of burnin iterations. |
| `seed` | the seed for random number generation. |

### Details

This function computes subject-specific predictions for the longitudinal outcome based on the joint model. This is accomplished with a Monte Carlo simulation scheme, similar to the one described in `jm.surv`. Let $\theta^{(m)}$ be a realization taken randomly from the posterior simulations of the joint model parameters $(m = 1, 2, ...,$ `n.sample`$)$. Also, let $b^{(m)}$ be a realization of the random-effects vector b from its posterior distribution given $\theta^{(m)}$ (see the MCMC scheme described in `jm.surv`). Posterior summaries of $x_i'(s)\alpha + w_i'(s)b_i$ (subject-specific prediction) at time $s$ are derived from the MCMC sample $\{(\theta^{(m)}, b^{(m)}), m = 1, 2, ...,$ `n.sample`$\}$.

### Value

Posterior summaries of the subject-specific predictions. A plot of the predictions is produced if `plot = TRUE`, with the predictions displayed by the solid curve, the pointwise Bayesian intervals displayed by the dashed curves, and `ptime[1]` displayed by the dashed vertical line. Note that with the default definition of `ptime`, `ptime[1]` is the last time the subject provided a longitudinal measurement.

### Author(s)

Shahedul Khan <khan@math.usask.ca>

### References

Rizopoulos D, The R package JMbayes for fitting joint models for longitudinal and time-to-event data Using MCMC, Journal of Statistical Software, 72(7): 1-45, 2016.

### See Also

`jm.lppredict`, `jm.reg`, `jm.surv`

### Examples

```
# Example: pbc data
lme.fit <- lme(log(bilirubin) ~ drug + ns(futime, 2), data = pbc.long,
  random = ~ ns(futime, 2) | id)
surv.fit <- coxph(Surv(st, status2) ~ drug * age, data = pbc.surv, x = TRUE)
jmfit.wph2 <- jm.reg(surv.fit = surv.fit, lme.fit = lme.fit,
    surv.model = "weibullph", timevar = "futime", form = "riz")
jm.summary(jmfit.wph2)
newdata2 <- data.frame(pbc.long[pbc.long$id == 2, ])
newdata6 <- data.frame(pbc.long[pbc.long$id == 6, ])
par(mfrow = c(1, 2))
pred2 <- jm.lspredict(jmfit = jmfit.wph2, newdata = newdata2)
pred6 <- jm.lspredict(jmfit = jmfit.wph2, newdata = newdata6)
```

```
# For comparison, use the same xlim and ylim in the two plots.
par(mfrow = c(1, 2))
pred2 <-  jm.lspredict(jmfit = jmfit.wph2, newdata = newdata2,
    xlim = c(0, 14),  ylim = c(-2.25, 4))
pred6 <- jm.lspredict(jmfit = jmfit.wph2, newdata = newdata6,
    xlim = c(0, 14), ylim = c(-2.25, 4))
```

## jm.plots *MCMC plots for posterior analysis*

### Description

This function produces plots for posterior analysis. The 'bayesplot' package is used for plotting. Available options are pair plots, rhat values as either points or a histogram, ratios of effective sample size to total sample size as either points or a histogram, trace plots, density plots, and plots of uncertainty intervals. See the 'bayesplot' package for detail.

### Usage

```
jm.plots(
  jmfit,
  pairs = TRUE,
  rhat = FALSE,
  neff = FALSE,
  trace = FALSE,
  density = FALSE,
  uncertainty.intervals = FALSE
)
```

### Arguments

| | |
|---|---|
| `jmfit` | a `jm.reg` fit with `method = "MCMC"`. |
| `pairs` | if `TRUE`, produce pair plots. |
| `rhat` | if `TRUE`, produce a plot for rhats. |
| `neff` | if `TRUE`, produce a plot for ratios of effective sample size to total sample size. |
| `trace` | if `TRUE`, produce trace plots. |
| `density` | if `TRUE`, produce density plots. |
| `uncertainty.intervals` | if `TRUE`, produce uncertainty intervals computed from parameter draws. |

### Details

The 'bayesplot' package is used for plotting. See the 'bayesplot' package for detail.

### Value

Plots for posterior analysis.

### Author(s)

Shahedul Khan <khan@math.usask.ca>

## References

Gabry J and Mahr T, bayesplot: Plotting for Bayesian Models, R package version 1.7.2, https://mc-stan.org/bayesplot

## See Also

jm.reg, jm.resid

## Examples

```
# Example: AIDS data from package JM
  library(JM)
  data(aids.id)
  data(aids)
  surv.fit <- coxph(Surv(Time, death) ~ drug + gender + prevOI + AZT,
          data = aids.id, x = TRUE)
  lme.fit <- lme(CD4 ~ obstime + obstime:drug + gender + prevOI + AZT,
      random =  ~ obstime | patient, data = aids)
  # Weibull PH with form = "riz"
  jmfit.wph0 <- jm.reg(surv.fit = surv.fit, lme.fit = lme.fit,
      surv.model = "weibullph", timevar = "obstime", form = "riz")
  jm.plots(jmfit.wph0, pairs = TRUE, rhat = TRUE, neff = TRUE,
    trace = TRUE, density = TRUE, uncertainty.intervals = TRUE)
```

# jm.reffects *Posterior summaries of the random effects from a joint model fit*

## Description

Returns posterior summary of the random effects. Note that the random effects are shared between the longitudinal and the survival components, and the link between these two processes via the random effects is commonly known as *latent association*. The formulation of the joint model is described in jm.reg.

## Usage

```
jm.reffects(jmfit, rhat.plot = FALSE, neff.plot = FALSE)
```

## Arguments

| | |
|---|---|
| jmfit | a jm.reg fit with method = "MCMC". |
| rhat.plot | if TRUE, produce a plot for rhats. |
| neff.plot | if TRUE, produce a plot for ratios of effective sample size to total sample size. |

## Details

The random effects are monitored in MCMC simulations in Stan. The summary method in rstan is used to produce the posterior summaries of the random effects.

## Value

b: posterior summaries of the random effects.

plots: plots if rhat.plot = TRUE and/or neff.plot = TRUE.

### Author(s)

Shahedul Khan <khan@math.usask.ca>

### See Also

`jm.reg`, `jm.summary`

### Examples

```
# Example: pbc data
lme.fit <- lme(log(bilirubin) ~ drug + ns(futime, 2),
  data = pbc.long, random = ~ ns(futime, 2) | id)
surv.fit <- coxph(Surv(st, status2) ~ drug * age, data = pbc.surv, x = TRUE)
# Bayesian estimation
jmfit.wph2 <- jm.reg(surv.fit = surv.fit, lme.fit = lme.fit,
    surv.model = "weibullph", timevar = "futime", form = "riz")
jm.reffects(jmfit.wph2, rhat.plot = TRUE, neff.plot = TRUE)
```

---

## `jm.reg` *Fit joint model*

---

### Description

Bayesian fit of a joint model. The time-to-event process is described using a parametric survival model (both AFT and PH are available), and the longitudinal process is modeled using the linear mixed-effects model. It is assumed that the association between the two submodels is induced by the longitudinal value. This version does not allow association induced by the random intercepts and/or slopes. The Markov Chain Monte Carlo (MCMC) for Bayesian inference is implemented via the 'rstan' package (Stan Development Team, 2020), which provides the R interface to Stan.

### Usage

```
jm.reg(
  surv.fit,
  lme.fit,
  surv.model = "weibullph",
  fixed.model = "nsimple",
  rand.model = "nsimple",
  timevar,
  form = "henderson",
  method = "MCMC",
  inits = NULL,
  warmup = 1000,
  iter = 3500,
  chains = 2,
  thin = 1,
  adapt_delta = 0.8,
  max_treedepth = 10,
  control = list()
)
```

### Arguments

`surv.fit` an object of class `coxph` from the 'survival' package, representing the Cox PH fit. It is required to specify `x = TRUE` in `coxph`.

`lme.fit` an object of class `lme` from the 'nlme' package, representing the linear mixed-effects model fit.

`surv.model` the survival model to be used to describe the event process. Available options are given as follow.

- AFT Models: `"weibull"`, `"lnormal"`, `"llogistic"`, `"eweibull"` and `"ggamma"` for Weibull, log-normal, log-logistic, exponentiated Weibull and generalized gamma distributions, respectively.
- PH Models: `"weibullph"` and `"gllph"` for Weibull and generalized log-logistic distributions, respectively.

Default is `"weibullph"`. The parameterization of these distributions is described in `etime.reg`.

`fixed.model` a character string describing the form of the fixed-effects component in `lme` (this argument is only used when an AFT model is considered to describe the event process). For computational efficiency, we recommend to use `fixed.model = "simple"` if the form is $x'(s)\alpha = \alpha_0+\alpha_1 s$ or $\alpha_0+\alpha_1 s+\alpha_2 x_2+...+\alpha_q x_q$, where $s$ denotes the time points at which the longitudinal measurements are recorded, and $x_2, ..., x_q$ are time-independent covariates; otherwise use `fixed.model = "nsimple"` (default). For example, use `fixed.model = "nsimple"` if $x'(s)\alpha$ includes an interaction term involving $s$ or includes `ns(s,df)`, where df is the degrees of freedom for a natural cubic spline. Note that `fixed.model = "nsimple"` also works if the form of the fixed-effects component is "simple" as described above.

`rand.model` a character string describing the form of the random-effects component in `lme` (this argument is only used when an AFT model is considered to describe the event process). For computational efficiency, we recommend to use `rand.model = "simple"` if the form is $w'(s)b = b_0 + b_1 s$, where $b_0$ and $b_1$ are random intercept and slope, respectively; otherwise use `rand.model = "nsimple"` (default). For example, use `rand.model = "nsimple"` if $w'(s)b$ includes `ns(s,df)`. Note that `rand.model = "nsimple"` also works if the form of the random-effects component is "simple" as described above.

`timevar` the name of the time variable in the linear mixed-effects model (a character string).

`form` a character string to describe the formulation of the joint model. Available options are `form = "henderson"` (default, can be abbreviated as "hen") for the formulation proposed by Henderson et al. (2000), and `form = "rizopoulos"` (can be abbreviated as "riz") for the formulation described by Rizopoulos (2012). See Details.

`method` estimation method to be used (a character string). Available options are `method = "MCMC"` for Bayesian inference (implemented vis the 'rstan' package), and `method = "2stage"` for the two-stage estimation method. Note that the two-stage approach may produce biased results as demonstrated by many authors, and hence it is not recommended. Since the MCMC approach is computatinally intensive and time consuming, the two-stage method is included for a quick but crude estimation of the parameters.

`inits` initial values of the parameters for MCMC; see Details.

`warmup` a positive integer specifying the number of burnin iterations per chain for MCMC (default is 1000).

`iter` a positive integer specifying the number of iterations for each chain including warmup (default is 3500).

`chains` a positive integer specifying the number of Markov chains (default is 2).

`thin` thinning interval for monitors (default is 1, the recommended value).

`adapt_delta` the target average proposal acceptance probability during Stan's adaptation period (default is 0.8). In general, you should not need to change `adapt_delta` unless you see a warning message about divergent transitions. If there is such a warning message, increase `adapt_delta` from the default to a value closer to 1 (e.g., from 0.90 to 0.99). See the Stan manual for detail.

`max_treedepth` a positive integer specifying the maximum treedepth (see the Stan manual). If there is a warning about transitions exceeding the maximum treedepth, try increasing the `max_treedepth` parameter (e.g., `max_treedepth = 15`).

`control` a list of control values (see Details).

## Details

The MCMC algorithm for Bayesian inference is implemented in Stan. The `coxph` and `lme` fits (the arguments `surv.fit` and `lme.fit`) are used to organize the data to be used in Stan. The event process can be modeled by one of Weibull AFT, log-normal AFT, log-logistic AFT, exponentiated Weibull AFT, generalized gamma AFT, Weibull PH and generalized log-logistic PH models, and the longitudinal process is characterized by the linear mixed-effects model. The event time distribution is characterized by the parameters $\rho$, $\kappa$ and $\gamma$ as described in `etime.reg`.

**Longitudinal Process:** We model the longitudinal response $y_{ij}$ at time $s_{ij}$ by the relationship $y_{ij} = \mu_i(s_{ij}) + U_i(s_{ij}) + \epsilon_{ij}$, where $\mu_i(s_{ij})$ is the mean response, $U_i(s_{ij})$ incorporates subject-specific random effects, and $\epsilon_{ij} \sim N(0, \sigma^2)$ is a sequence of mutually independent measurement errors. We assume that the mean response at time $s$ is characterized by a linear model $\mu_i(s) = x_i'(s)\alpha$, where $x_i(s)$ is a vector of covariates (possibly time-dependent) and $\alpha$ is the corresponding vector of regression coefficients (fixed effects). For $U_i(s)$, we assume a linear random effects model $U_i(s) = w_i'(s)b_i$, where $w_i(s)$ is a vector of covariates and $b_i \sim N(0, \Sigma_b)$ is the corresponding vector of random effects.

**AFT Event Process:** The event intensity process at time $t$ can be expressed as $\lambda_i(t) = \lambda_0[g_i(t)]$ exp$[-z_i'\beta - V_i(t)]$, where $\lambda_0(.)$ is the baseline intensity function of the assumed distribution (see `etime.reg`), $g_i(t) = \int_0^t$exp$[-z_i'\beta - V_i(u)]du$, $z_i$ is a vector of baseline covariates, and $\beta$ is the corresponding vector of regression coefficients (the specificantion of $V_i(t)$ is described below). With this formulation, the survivor function can be written as $S_i(t) = S_0[g_i(t)]$, where $S_0(.)$ is the baseline survivor function of the assumed distribution (see Cox and Oakes (1984) for detail).

**PH Event Process:** The event intensity process at time $t$ can be expressed as $\lambda_i(t) = \lambda_0(t)$ exp$[z_i'\beta + V_i(t)]$ (the specificantion of $V_i(t)$ is described below). With this formulation, the survivor function can be written as $S_i(t)$ =exp$[-\Lambda_i(t)]$, where $\Lambda_i(t) = \int_0^t \lambda_0(u)$exp$[z_i'\beta + V_i(u)]du$.

**Association Structure:** In our implementation, dependence between the longitudinal and the time-to-event sub-models is captured through $V_i(t)$. In Rizopoulos (2012) (see also the 'JM' package), $V_i(t)$ is defined as $V_i(t) = \phi[\mu_i(t) + U_i(t)]$, where $\phi$ is the measure of association (regression coefficient for the time-dependent covariate of the event time process) induced by the fitted longitudinal values. On the other hand, Henderson et al. (2000) (see also the 'joineR' package) proposed $V_i(t) = \phi U_i(t)$. Both these formulations have been implemented, with `form = "henderson"` and `form = "rizopoulos"` for the formulations described by Henderson et al. (2000) and Rizopoulos (2012), respectively.

**Initial Values for MCMC:** Initial values of the parameters can be specified (optional) using the `inits` argument. Use the "inits via list" format as described in the 'rstan' package: set inital values by providing a list equal in length to the number of chains. The elements of this list should themselves be named lists, where each of these named lists has the name of a parameter and is used to specify the initial values for that parameter for the corresponding chain. The form of each named list for a joint model with the survival process described by a two-parameter distribution ($\rho$ and $\kappa$) is

```
list(alpha = a vector, beta = a vector, phi = a scalar, squ_kappa = a scalar,
inv_rho2 = a scalar, b_unscaled = a matrix, rand_cov = a matrix, inv_sigma2 = a scalar),
```

where `alpha` = $\alpha$, `beta` = $\beta$, `phi` = $\phi$, `squ_kappa` = $\kappa^2$, `inv_rho2` = $1/\rho^2$, `b_unscaled` = unscaled random effects (a matrix with the number of columns equal to the number of random effects for each subject, and the number of rows equal to the number of subjects), `rand_cov` = $\Sigma_b$, and `inv_sigma2` = $1/\sigma^2$. Note that `b_unscaled` = `solve(t(chol(rand_cov))) %*% t(b)`. For a joint model with the survival process described by the exponentiated Weibull or the generalized gamma distribution (three parameters: $\rho$, $\kappa$ and $\gamma$), add `squ_gam` (a scalar) in the above list, where `squ_gam` = $\gamma^2$. With the survival process described by the generalized log-logistic PH model, add `inv_gam2` in the above list, where `inv_gam2` = $1/\gamma^2$.

`control`: a list of control values with components:

- `alpha_mu`: $\mu_\alpha$ (a vector) for the prior $\alpha$ ~ $N(\mu_\alpha, \Sigma_\alpha)$, where $\alpha$ is the vector of fixed-effects coefficients of the longitudinal model.
- `alpha_sd`: standard deviations for $\alpha$ (a vector), that is, square roots of the diagonal elements of $\Sigma_\alpha$.
- `beta_mu`: $\mu_\beta$ (a vector) for the prior $\beta$ ~ $N(\mu_\beta, \Sigma_\beta)$, where $\beta$ is the vector of coefficients for the baseline covariates of the survival model (no intercept).
- `beta_sd`: standard deviations for $\beta$ (a vector), that is, square roots of the diagonal elements of $\Sigma_\beta$.
- `phi_mu`: $\mu_\phi$ (a scalar) for the prior $\phi$ ~ $N(\mu_\phi, \sigma_\phi)$, where $\phi$ is the association parameter (regression coefficient for the time-dependent covariate of the event time process).
- `phi_sd`: the value of $\sigma_\phi$ (a scalar).
- `A`: a positive definite matrix for the prior $\Sigma_b$ ~ InvWishart(A, df), where $\Sigma_b$ is the covariance matrix of the random-effects vector $b_i$ with dimension equal to the number of random effects for each $i$.
- `nu`: the value of the df for the inverse Wishart prior InvWishart(A, df).
- `a0`: the shape parameter for the prior $\sigma^{-2}$ ~ gamma(shape, rate).
- `a1`: the rate parameter for the prior $\sigma^{-2}$ ~ gamma(shape, rate).
- `b0`: the shape parameter for the prior $1/\rho^2$ ~ gamma(shape, rate).
- `b1`: the rate parameter for the prior $1/\rho^2$ ~ gamma(shape, rate).
- `c0`: the shape parameter for the prior $\kappa^2$ ~ gamma(shape, rate).
- `c1`: the rate parameter for the prior $\kappa^2$ ~ gamma(shape, rate).
- `d0`: the shape parameter for the prior $\gamma^2$ ~ gamma(shape, rate)
  (for the generalized log-logistic PH, $1/\gamma^2$ ~ gamma(shape, rate)).
- `d1`: the rate parameter for the prior $\gamma^2$ ~ gamma(shape, rate)
  (for the generalized log-logistic PH, $1/\gamma^2$ ~ gamma(shape, rate)).
- `qpoints`: number of quadrature points to approximate the integral involved in the survival model; available options are 3, 5, 7, 15, 21, 31 and 41.
- `seed`: the seed for random number generation in Stan.

## Value

`Survival sub-model`: a description of the survival model used to describe the event process.

`Longitudinal sub-model`: a description the mixed-effects model used to describe the longitudinal process.

`Association structure`: the association structure linking the two processes.

`stan_fit`: posterior summaries of the parameters.

`time`: computational time.

## Author(s)

Shahedul Khan <khan@math.usask.ca>

## References

Cox DR and Oakes D, Analysis of survival data, Chapman and Hall/CRC, 1984.

Henderson R, Diggle P, and Dobson A, Joint modelling of longitudinal measurements and event time data, Biostatistics, 1: 465-480, 2000.

Rizopoulos D, Joint Models for Longitudinal and Time-to-Event Data: With Applications in R, Chapman and Hall/CRC, 2012.

Stan Development Team, Stan Modeling Language Users Guide and Reference Manual, 2020, Version https://mc-stan.org.

Stan Development Team, RStan: the R interface to Stan, 2020, Version http://mc-stan.org/.

## See Also

`jm.icriteria`, `jm.lppredict` , `jm.lspredict`, `jm.plots`, `jm.resid`, `jm.sim`, `jm.summary`, `jm.surv`

## Examples

```
# Example 1: AIDS data from package 'JM'
library(JM)
data(aids.id)
data(aids)
surv.fit <- coxph(Surv(Time, death) ~ drug + gender + prevOI + AZT,
        data = aids.id, x = TRUE)
lme.fit <- lme(CD4 ~ obstime + obstime:drug + gender + prevOI + AZT,
    random =  ~ obstime | patient, data = aids)
# Weibull PH with form = "riz"
jmfit.wph0 <- jm.reg(surv.fit = surv.fit, lme.fit = lme.fit,
    surv.model = "weibullph", timevar = "obstime", form = "riz")
jmfit.wph0
# Generalized log-logistic PH with form = "riz"
jmfit.gllph0 <- jm.reg(surv.fit = surv.fit, lme.fit = lme.fit,
    surv.model = "gllph", timevar = "obstime", form = "riz")
jmfit.gllph0
# Weibull AFT with form = "riz"
jmfit.w0 <- jm.reg(surv.fit = surv.fit, lme.fit = lme.fit,
    surv.model = "weibull", timevar = "obstime", form = "riz")
jmfit.w0
# Exponentiated Weibull AFT with form = "hen"
jmfit.ew0 <- jm.reg(surv.fit = surv.fit, lme.fit = lme.fit,
    surv.model = "eweibull", timevar = "obstime", form = "hen")
```

```
jmfit.ew0
# Summary of jmfit.wph0
jm.summary(jmfit.wph0)
# WAIC and DIC
jm.icriteria(jmfit.wph0)
# MCMC Diagnostic plots
jm.plots(jmfit.wph0, pairs = TRUE, rhat = TRUE, neff = TRUE,
  trace = TRUE, density = TRUE, uncertainty.intervals = TRUE)
# Cox-Snell Residuals
jm.resid(jmfit.wph0)

# Example 2: AIDS data from package 'JM'
surv.fit <- coxph(Surv(Time, death) ~ drug + gender + prevOI + AZT,
        data = aids.id, x = TRUE)
lme.fit <- lme(CD4 ~ obstime + drug + gender + prevOI + AZT,
    random =  ~ obstime | patient, data = aids)
# For AFT models, use fixed.model = "simple" and rand.model = "simple"
# (for computational efficiency)
# Log-normal AFT with form = "hen"
jmfit.ln1 <- jm.reg(surv.fit = surv.fit, lme.fit = lme.fit,
    surv.model = "lnormal", fixed.model = "simple",
    rand.model = "simple",  timevar = "obstime", form = "hen")
jm.summary(jmfit.ln1)
# Generalized gamma AFT with form = "riz"
jmfit.gg1 <- jm.reg(surv.fit = surv.fit, lme.fit = lme.fit,
    surv.model = "ggamma",  fixed.model = "simple",
    rand.model = "simple",  timevar = "obstime",  form = "riz")
jm.summary(jmfit.gg1)
# For Weibull PH, use the default fixed.model = "nsimple" and rand.simple = "nsimple".
jmfit.wph1 <- jm.reg(surv.fit = surv.fit, lme.fit = lme.fit,
    surv.model = "weibullph", timevar = "obstime", form = "hen")
jm.summary(jmfit.wph1)

# Example 3: pbc data
lme.fit <- lme(log(bilirubin) ~ drug + ns(futime, 2),
  data = pbc.long, random = ~ ns(futime, 2) | id)
surv.fit <- coxph(Surv(st, status2) ~ drug * age, data = pbc.surv, x = TRUE)
# Two-stage estimation
jmfit.gg2stage <- jm.reg(surv.fit = surv.fit, lme.fit = lme.fit,
    surv.model = "ggamma", method = "2stage", timevar = "futime", form = "riz")
jmfit.gg2stage
# Bayesian estimation
jmfit.ll2 <- jm.reg(surv.fit = surv.fit, lme.fit = lme.fit,
    surv.model = "llogistic", timevar = "futime", form = "riz")
jm.summary(jmfit.ll2)
jmfit.wph2 <- jm.reg(surv.fit = surv.fit, lme.fit = lme.fit,
    surv.model = "weibullph", timevar = "futime", form = "riz")
jm.summary(jmfit.wph2)
```

## jm.resid *Cox-Snell residuals for a jm.reg fit*

### Description

Returns posterior summaries of the Cox-Snell residuals for a `jm.reg` fit. It also produces a residual plot.

## Usage

```
jm.resid(
  jmfit,
  posterior.mean = TRUE,
  n.sample = NULL,
  xlim = NULL,
  ylim = NULL,
  xlab = NULL,
  ylab = NULL,
  main = NULL,
  seed = sample.int(.Machine$integer.max, 1)
)
```

## Arguments

| | |
|---|---|
| `jmfit` | a `jm.reg` fit with `method = "MCMC"`. |
| `posterior.mean` | returns posterior means of the residuals if `posterior.mean = TRUE`, otherwise returns posterior medians of the residuals. |
| `n.sample` | an integer denoting how many MCMC samples to use. Default is `min(200,n.iter)`, where n.iter is the number of MCMC iterations (for all chains combined). |
| `xlim` | x limits (a vector of length 2 of the form `c(minimum,maximum)`) of the plot (optional). |
| `ylim` | y limits (a vector of length 2 of the form `c(minimum,maximum)`) of the plot (optional). |
| `xlab` | a label (character expression) for the x axis (optional). |
| `ylab` | a label (character expression) for the y axis (optional). |
| `main` | main title (character expression) of the plot (optional). |
| `seed` | the seed for random number generation to draw `n.sample` samples. Default is `sample.int(.Machine$integer.max,1)`. |

## Details

A random sample of size `n.sample` is drawn from the posterior simulations, and the Cox-Snell residuals are computed for each sample. Following Rizopoulos and Ghosh (2011), the estimate of the posterior expectation is obtained as the MCMC sample mean (or median) of the residuals at each observed time point. A plot of residual vs. $-\log S_{KM}$(redidual) is then produced (solid circles), where $S_{KM}$(redidual) is the Kaplan-Meier estimate of the survivor function. Uncertainty in the plot can be assessed through the `n.sample` sets of residuals obtained from the posterior simulations (Zhou and Hanson, 2018). These are shown in the plot as grey lines. The plot also shows the unit slop line (in black). The plot of the Cox-Snell residuals should be roughly a straight line with unit slope when the survival sub-model is adequate.

## Value

`Cox-Snell Residuals:` posterior expectation of the residuals and $-\log S_{KM}$(rediduals).

`plot:` see Details.

## Author(s)

Shahedul Khan <khan@math.usask.ca>

### References

Rizopoulos D and Ghosh P, A Bayesian semiparametric multivariate joint model for multiple longitudinal outcomes and a time-to-event, Statistics in Medicine, 30: 1366-1380, 2011.

Zhou H and Hanson T, A unified framework for fitting Bayesian semiparametric models to arbitrarily censored survival data, including spatially referenced data, Journal of the American Statistical Association, 113(522), 571-581, 2018.

### See Also

`jm.icriteria`, `jm.plots`, `jm.reg`

### Examples

```
# Example: AIDS data from package 'JM'
library(JM)
data(aids.id)
data(aids)
surv.fit <- coxph(Surv(Time, death) ~ drug + gender + prevOI + AZT,
        data = aids.id, x = TRUE)
lme.fit <- lme(CD4 ~ obstime + obstime:drug + gender + prevOI + AZT,
    random =  ~ obstime | patient, data = aids)
# Weibull PH with form = "riz"
jmfit.wph0 <- jm.reg(surv.fit = surv.fit, lme.fit = lme.fit,
    surv.model = "weibullph", timevar = "obstime", form = "riz")
jm.resid(jmfit.wph0)
```

## `jm.sim` *Simulate from joint models*

### Description

This function simulates longitudinal responses and event times from joint models. Available options for the time-to-event submodel are Weibull AFT, log-logistic AFT, log-normal AFT, exponentiated Weibull AFT, generalized gamma AFT, Weibull PH and generalized log-logistic PH. For the longitudinal process, the linear mixed-effects model is assumed, with *a simple linear model for the random-effects component* (i.e., $b_0 + b_1$ `times`, where $b_0$ and $b_1$ are the random intercept and random slope, respectively).

### Usage

```
jm.sim(
  surv.formula,
  lme.formula,
  surv.model,
  form,
  surv.par,
  lme.par,
  times,
  timevar,
  Data
)
```

## Arguments

`surv.formula` a one-sided formula of the form ~ `z1 + z2 + ... + zp`, where `z1,z2,...,zp` are baseline covariates for the time-to-event process.

`lme.formula` a one-sided formula of the form ~ `times` or ~ `times + x2 + x3 + ... + xq` for the fixed-effects component of the longitudinal model, where `times` is the time variable (time points at which longitudinal measurements are taken) and `x2,x3,...,xq` are time-independent covariates.

`surv.model` the survival model to be used to describe the event process. Available options are given as follow.

- AFT Models: `"weibull"`, `"lnormal"`, `"llogistic"`, `"eweibull"` and `"ggamma"` for Weibull, log-normal, log-logistic, exponentiated Weibull and generalized gamma distributions, respectively.
- PH Models: `"weibullph"` and `"gllph"` for Weibull and generalized log-logistic distributions, respectively.

The parameterization of these distributions is described in `etime.reg`.

`form` a character string to describe the formulation of the joint model. Available options are `form = "rizopoulos"` (can be abbreviated as "riz") for the formulation described by Rizopoulos (2012), and `form = "henderson"` (can be abbreviated as "hen") for the formulation proposed by Henderson et al. (2000). See `jm.reg` for details.

`surv.par` a named list of the form `list(beta = a vector,phi = a scalar,logrho = a scalar, logkappa = a scalar,loggamma = a scalar,status.rho = a scalar)` for the time-to-event model parameters, where `beta` is the vector of regression coefficients for the baseline covariates as specified in `surv.formula` (without an intercept), `phi` is the association parameter, `logrho` = log$\rho$, `logkappa` = log$\kappa$, `loggamma` = log$\gamma$, and `status.rho` is the rate parameter of the exponential distribution from which censored data are generated. See `etime.reg` for parameterizations of the time-to-event distributions in terms of $\rho$, $\kappa$ and $\gamma$.

`lme.par` a named list of the form `list(alpha = a vector,sigma = a scalar,rand_cov = a 2 x 2 covariance matrix)` for the longitudinal model parameters, where `alpha` is the vector of fixed-effects regression coefficients as specified in `lme.formula` (includes an intercept), `sigma` is the standard deviation of measurement errors (i.e., the value of $\sigma$ for $\epsilon$ ~ $N(0, \sigma^2)$), and `rand_cov` is the covariance matrix of the random effects $b_0$ and $b_1$. Note that only a simple linear model $b_0 + b_1$ `times` is allowed for random effects.

`times` a numeric vector for the time points at which longitudinal measurements are planned to be taken.

`timevar` the time variable (a character string) of the longitudinal model. For example, if `lme.formula = ~ times + x1`, then `timevar = "times"`.

`Data` a data frame containing the baseline covariates `z`'s for the time-to-event model and the time-independent covariates `x`'s for the longitudinal model. For $n$ subjects, it must must have $n$ rows.

## Details

This function simulates longitudinal responses and event times from joint models. The longitudinal model is of the form $y = x'(s)\alpha + w'(s)b + \epsilon$, where $x'(s)\alpha = \alpha_0 + \alpha_1 s$ or $\alpha_0 + \alpha_1 s + \alpha_2 x_2 + \alpha_3 x_3 + ... + \alpha_q x_q$, $w'(s)b = b_0 + b_1 s$, $\epsilon$ ~ $N(0, \sigma^2)$ and $b$ ~ $N(0, \Sigma_b)$. For an AFT event process, the survivor function is of the form $S_0(\int_0^t \exp[-(z'\beta + \phi(x'(u)\alpha + w'(u)b)]du)$ for `form = "rizopoulos"`, and

$S_0(\int_0^t \exp[-(z'\beta + \phi w'(u)b)]du)$ for `form = "henderson"`. For a PH event process, the survivor function is of the form $\exp[-\Lambda(t)]$, where $\Lambda(t) = \int_0^t \lambda_0(u) \exp[z'\beta + \phi(x'(u)\alpha + w'(u)b)]du$ for `form = "rizopoulos"`, and $\Lambda(t) = \int_0^t \lambda_0(u) \exp[z'\beta + \phi w'(u)b]du$ for `form = "henderson"`. Event times are drawn by solving the equation `survivor function = U` for $t$, where `U` is a Uniform(0, 1) variable. Note that the amount of censoring can be controlled by changing the value of `status.rho`.

## Value

A list with components:

`b`: random effects (a matrix).

`long.data`: simulated longitudinal data.

`surv.data`: simulated event time data.

## Author(s)

Shahedul Khan <khan@math.usask.ca>

## References

Henderson R, Diggle P, and Dobson A, Joint modelling of longitudinal measurements and event time data, Biostatistics, 1: 465-480, 2000.

Rizopoulos D, Joint Models for Longitudinal and Time-to-Event Data: With Applications in R, Chapman and Hall/CRC, 2012.

## See Also

`jm.reg`, `jm.summary`

## Examples

```
n <- 100
surv.formula <- ~ z1 + z2
lme.formula <- ~ times + x1 + x2
cor.mat <- matrix(c(1, -0.2, -0.2, 1), ncol = 2, byrow = TRUE)
sd <- c(0.5, 0.1)
rand_cov <- (sd %*% t(sd)) * cor.mat
lme.par <- list(alpha = c(10, -1, -1, 0.25), sigma = 0.5, rand_cov = rand_cov)
surv.par <- list(beta = c(-0.5, 0.5), phi = -0.25, logrho = log(0.01),
  logkappa = log(2), status.rho = 0.003)
z1 <- rbinom(n, 1, 0.5)
z2 <- rnorm(n)
# Data with all covariate information
Data <- data.frame(z1 = z1, z2 = z2, x1 = z1, x2 = z2)
# Time points at which longitudinal measurements are planned to be taken
times <- c(0, 1, 2, 3, 4)
timevar <- "times"
jmsim.dat <- jm.sim(surv.formula = surv.formula, lme.formula = lme.formula,
  surv.model = "weibullph", form = "hen", surv.par = surv.par,
  lme.par = lme.par, times = times, timevar = timevar, Data = Data)
surv.data <- jmsim.dat$surv.data
long.data <- jmsim.dat$long.data
surv.fit <- coxph(Surv(st, status) ~ z1 + z2, data = surv.data, x = TRUE)
lme.fit <- lme(y ~ times + x1 + x2, random =  ~ times | id, data = long.data)
```

```
swph <- jm.reg(surv.fit = surv.fit, lme.fit = lme.fit,
    surv.model = "weibullph", timevar = "times", form = "hen")
jm.summary(swph)
```

## jm.summary *Summary of a jm.reg fit*

### Description

Produces a summary of the Bayesian joint model fit. It uses a `jm.reg` fit as its argument.

### Usage

```
jm.summary(jmfit)
```

### Arguments

| | |
|---|---|
| `jmfit` | a `jm.reg` fit with `method` = `"MCMC"`. |

### Details

This function uses the posterior draws and the posterior summary from class "stanfit".

### Value

A summary of the survival data, longitudinal data, the joint model, number of MCMC chains, number of iterations per chain after warmup, and computaional time. It also gives

`Posterior summary (survival sub-model)`: posterior summary for the survival sub-model.

`Posterior summary (exp(coef) of the survival sub-model)`: exp(coef) for the survival submodel (for PH models only).

`Posterior summary (longitudinal sub-model)`: posterior summary for the longitudinal submodel.

### Author(s)

Shahedul Khan <khan@math.usask.ca>

### See Also

`jm.icriteria`, `jm.plots`, `jm.reg`, `jm.resid`

### Examples

```
# Example: pbc data
lme.fit <- lme(log(bilirubin) ~ drug + ns(futime, 2),
  data = pbc.long, random = ~ ns(futime, 2) | id)
surv.fit <- coxph(Surv(st, status2) ~ drug * age, data = pbc.surv, x = TRUE)
jmfit.ll2 <- jm.reg(surv.fit = surv.fit, lme.fit = lme.fit,
    surv.model = "llogistic", timevar = "futime", form = "riz")
jm.summary(jmfit.ll2)
jmfit.wph2 <- jm.reg(surv.fit = surv.fit, lme.fit = lme.fit,
    surv.model = "weibullph", timevar = "futime", form = "riz")
jm.summary(jmfit.wph2)
```

## jm.surv *Prediction of survival probabilities from a joint model fit*

### Description

Dynamic predictions of survival probabilites as described by Rizopoulos (2016, 2020).

### Usage

```
jm.surv(
  jmfit,
  newdata,
  n.sample = NULL,
  st = NULL,
  plot = TRUE,
  posterior.mean = TRUE,
  include.y = TRUE,
  xlim = NULL,
  ylim = NULL,
  xlab = NULL,
  ylab.left = NULL,
  ylab.right = NULL,
  main = NULL,
  adapt_delta = 0.9,
  max_treedepth = 15,
  warmup = 200,
  seed = sample.int(.Machine$integer.max, 1)
)
```

### Arguments

| | |
|---|---|
| `jmfit` | a `jm.reg` fit with `method = "MCMC"`. |
| `newdata` | a data frame in which to look for variables with which to predict. The data frame must have columns that correspond by name to the original variables used to fit the joint model. |
| `n.sample` | an integer denoting how many MCMC samples to use. Default is `min(200,n.iter)`, where n.iter is the number of MCMC iterations (for all chains combined). |
| `st` | a numeric vector of times at which survival probabilities P($T > t \mid T > s$, history of the longitudinal response, covariate information) are to be computed (see Details). If `NULL`, $s$ is taken to be the last time the subject provided a longitudinal measurement, and `t = seq(s,max.st,length = 15)`, where `max.st` is `floor(max(event times))`. A warning will be given If $s >$ the last time the individual provided a longitudinal measurement. |
| `plot` | if `TRUE`, a plot of the survival curve is produced. |
| `posterior.mean` | if `TRUE`, posterior means of the survival probabilities are used to produce the survival curve, otherwise posterior medians are used. |
| `include.y` | if `TRUE`, the fitted longitudinal profile is included in the survival curve plot. |
| `xlim` | x limits (a vector of length 2 of the form `c(minimum,maximum)`) of the plot (optional). |

`ylim` limits for the left y axis (a vector of length 2 of the form `c(minimum,maximum)`) of the plot (optional). If `include.y = TRUE`, the left y axis is for the the fitted longitudinal profile, otherwise it is for predictions of survival probabilites.

`xlab` a label (character expression) for the x axis (optional).

`ylab.left` a label (character expression) for the left y axis (optional). If `include.y = TRUE`, the left y axis is for the the fitted longitudinal profile, otherwise it is for predictions of survival probabilites.

`ylab.right` a label (character expression) for the right y axis (optional). If `include.y = TRUE`, the right y axis is for predictions of survival probabilites, otherwise it is ignored.

`main` main title (character expression) of the plot (optional).

`adapt_delta` the target average proposal acceptance probability during Stan's adaptation period (see Details).

`max_treedepth` a positive integer specifying the maximum treedepth (see the Stan manual).

`warmup` a positive integer specifying the number of burnin iterations (see Details).

`seed` the seed for random number generation.

## Details

The algorithm proposed by Rizopoulos (2020) is used for dynamic predictions. The history of the longitudinal response up to time $s$ along with covariate information are taken into consideration for predictions. The algorithm is described as follows (see also Rizopoulos (2016, 2020)).

1. Randomly take a realization from the posterior simulations of the joint model parameters. Denote it by $\theta^{(1)}$.
2. Given $\theta^{(1)}$, the history of the longitudinal response, covariate information, and survival up to time $s$, draw a realization of the random-effects vector $b$ from its posterior distribution. For this step, the MCMC algorithm (implemented in Stan) is used with $n = 1$, `event time` = $s$ and `status = 0`. The number of burnin iterations, adapt_delta and max_treedepth in Stan can be controlled using the arguments `warmup` (default is 200), `adapt_delta` (default is 0.90) and `max_treedepth` (default is 15). Note that Rizopoulos (2020) used the Metropolis-Hastings algorithm with independent proposals (a properly centered and scaled multivariate t distribution) to estimate $b$.
3. Using $\theta^{(1)}$ and $b$, compute P($T > t|T > s$) = P($T > t$)/P($T > s$).
4. Repeat steps 1-3 `n.sample` times.

Posterior summeries of the conditional probabilities are derived from the MCMC sample drawn using the above algorithm.

## Value

Posterior summaries of the predictions of survival probabilites are returned. A plot of the survival curve is produced if `plot = TRUE`, with the predictions displayed by the solid line and the pointwise Bayesian intervals displayed by the dashed lines.

## Author(s)

Shahedul Khan <khan@math.usask.ca>

## References

Rizopoulos D, JMbayes: Joint Modeling of Longitudinal and Time-to-Event Data under a Bayesian Approach, R package version 0.8-85, 2020, https://cran.r-project.org/web/packages/JMbayes.

Rizopoulos D, The R package JMbayes for fitting joint models for longitudinal and time-to-event data Using MCMC, Journal of Statistical Software, 72(7): 1-45, 2016.

## See Also

jm.lppredict, jm.lspredict, jm.reg

## Examples

```
# Example: AIDS data from package 'JM'
library(JM)
data(aids.id)
data(aids)
surv.fit <- coxph(Surv(Time, death) ~ drug + gender + prevOI + AZT,
        data = aids.id, x = TRUE)
lme.fit <- lme(CD4 ~ obstime + obstime:drug + gender + prevOI + AZT,
    random =  ~ obstime | patient, data = aids)
jmfit.ew0 <- jm.reg(surv.fit = surv.fit, lme.fit = lme.fit,
    surv.model = "eweibull",  timevar = "obstime", form = "riz")
newdata <- data.frame(aids[aids$patient == 409, ])
jm.surv(jmfit.ew0, newdata = newdata, posterior.mean = FALSE)
```

---

# LR.test *Likelihood ratio test*

---

## Description

Likelihood ratio test for Weibull AFT as a submodel of the exponentiated Weibull AFT and generalized gamma AFT models.

## Usage

```
LR.test(fit)
```

## Arguments

| | |
|---|---|
| fit | a list of two etime.reg fits: one must be the Weibull AFT fit and the other one is the exponentiated Weibull AFT fit or the generalized gamma AFT fit. |

## Details

The log-likelihood values from the two fits are used for the likelihood ratio test.

## Value

chi.sq: chi-square test statistic.

p value: p value.

### Author(s)

Shahedul Khan <khan@math.usask.ca>

### See Also

etime.reg, etime.resid.

### Examples

```
library(frailtypack)
data(readmission)
fit.gg <- etime.reg(c(t.start, t.stop, event) ~ sex + chemo + charlson,
    surv.model = "ggamma", Data = readmission)
fit.ew <- etime.reg(c(t.start, t.stop, event) ~ sex + chemo + charlson,
    surv.model = "eweibull", Data = readmission)
fit.w <- etime.reg(c(t.start, t.stop, event) ~ sex + chemo + charlson,
    surv.model = "weibull", Data = readmission)
LR.test(list(fit.gg, fit.w))
LR.test(list(fit.ew, fit.w))
```

## pbc.long *Mayo Clinic Primary Biliary Cirrhosis Data*

### Description

The data is from the Mayo Clinic trial in primary biliary cirrhosis (PBC) of the liver conducted between 1974 and 1984. A total of 424 PBC patients, referred to Mayo Clinic during that ten-year interval, met eligibility criteria for the randomized placebo controlled trial of the drug D-penicillamine. The first 312 cases in the data set participated in the randomized trial and contain largely complete data. This longitudinal data set contains multiple laboratory results, but only on the first 312 patients.

### Usage

```
data(pbc.long)
```

### Format

A data frame with 1945 rows.

- `id`: subject identifier
- `st`: number of years between registration and the earlier of death, transplantion, or study analysis time
- `status`: 0 = alive, 1 = transplanted, 2 = dead
- `status2`: 1 = dead, 0 = alive or transplanted
- `drug`: 1= D-penicillamine, 0=placebo
- `age`: age in years, at registration
- `sex`: 0 = male, 1 = female
- `futime`: number of years between enrollment and this visit date, remaining values on the line of data refer to this visit

- `ascites`: presence of ascites: 0 = no, 1 = yes
- `hepatomegaly`: presence of hepatomegaly: 0 = no, 1 = yes
- `spiders`: presence of spiders: 0 = no, 1 = yes
- `edema`: presence of edema: 0 = no edema and no diuretic therapy for edema; 0.5 = edema present without diuretics, or edema resolved by diuretics; 1 = edema despite diuretic therapy
- `bilirubin`: serum bilirubin in mg/dl
- `cholesterol`: serum cholesterol in mg/dl
- `albumin`: albumin in gm/dl
- `alkaline`: alkaline phosphatase in U/liter
- `sgot`: SGOT in U/ml (serum glutamic-oxaloacetic transaminase, the enzyme name has subsequently changed to "ALT" in the medical literature)
- `platelets`: platelets per cubic ml/1000
- `proTime`: prothrombin time in seconds
- `histStage`: histologic stage of disease

## Source

`https://www.mayo.edu/research/documents/pbcseqhtml/doc-10027141`

## References

Therneau T and Grambsch P, Modeling survival data: extending the Cox Model, Springer-Verlag, 2000.

## Examples

```
summary(pbc.long)
```

---

# `pbc.surv` *Mayo Clinic Primary Biliary Cirrhosis Data*

---

## Description

The data is from the Mayo Clinic trial in primary biliary cirrhosis (PBC) of the liver conducted between 1974 and 1984. A total of 424 PBC patients, referred to Mayo Clinic during that ten-year interval, met eligibility criteria for the randomized placebo controlled trial of the drug D-penicillamine. The first 312 cases in the data set participated in the randomized trial and contain largely complete data. This data frame contains the first measurement for each patient from the data set `pbc.long`.

## Usage

```
data(pbc.surv)
```

### Format

A data frame with 312 rows.

- id: subject identifier
- st: number of years between registration and the earlier of death, transplantion, or study analysis time
- status: 0 = alive, 1 = transplanted, 2 = dead
- status2: 1 = dead, 0 = alive or transplanted
- drug: 1= D-penicillamine, 0=placebo
- age: age in years, at registration
- sex: 0 = male, 1 = female
- futime: number of years between enrollment and this visit date, remaining values on the line of data refer to this visit
- ascites: presence of ascites: 0 = no, 1 = yes
- hepatomegaly: presence of hepatomegaly: 0 = no, 1 = yes
- spiders: presence of spiders: 0 = no, 1 = yes
- edema: presence of edema: 0 = no edema and no diuretic therapy for edema; 0.5 = edema present without diuretics, or edema resolved by diuretics; 1 = edema despite diuretic therapy
- bilirubin: serum bilirubin in mg/dl
- cholesterol: serum cholesterol in mg/dl
- albumin: albumin in gm/dl
- alkaline: alkaline phosphatase in U/liter
- sgot: SGOT in U/ml (serum glutamic-oxaloacetic transaminase, the enzyme name has subsequently changed to "ALT" in the medical literature)
- platelets: platelets per cubic ml/1000
- proTime: prothrombin time in seconds
- histStage: histologic stage of disease

### Source

https://www.mayo.edu/research/documents/pbcseqhtml/doc-10027141

### References

Therneau T and Grambsch P, Modeling survival data: extending the Cox Model, Springer-Verlag, 2000.

### Examples

```
summary(pbc.surv)
```

## print.JMR *Prints JMR objects*

### Description

Prints JMR objects

### Usage

```
## S3 method for class 'JMR'
print(x, digits = max(options()$digits - 4, 3))
```

### Arguments

| | |
|---|---|
| x | A JMR object |
| digits | minimal number of significant digits |

### Value

No value is returned.

### Author(s)

Shahedul Khan <khan@math.usask.ca>

## sim.rec *Simulate recurrent event data*

### Description

Simulate recurrent event data for fixed covariates

### Usage

```
sim.rec(surv.formula, surv.par, surv.model, Data, origin, end.time)
```

### Arguments

| | |
|---|---|
| surv.formula | a one-sided formula of the form ~ z1 + z2 + ... + zp, where z1,z2,...,zp are baseline covariates for the time-to-event process. |
| surv.par | a list of the form list(beta = a vector,logrho = a scalar,logkappa = a scalar) for the Weibull AFT, log-logistic AFT, log-normal AFT and Weibull PH models, and list(beta = a vector,logrho = a scalar,logkappa = a scalar,loggamma = a scalar) for the exponentiated Weibull AFT, generalized gamma AFT and generalized log-logistic PH models, where beta is the vector of regression coefficients for the baseline covariates as specified in surv.formula (without an intercept), and logrho, logkappa and loggamma are the log of the parameters $\rho$, $\kappa$ and $\gamma$ (see etime.reg for a description of the distribution parameterizations). Note that surv.par must be a named list as shown above. |

`surv.model` the model to be used to describe the recurrent event process. Available options are `"weibull"`, `"llogistic"`, `"lnormal"`, `"eweibull"` and `"ggamma"` for the AFT model, and `"weibullph"` and `"gllph"` for the PH model.

`Data` a data frame containing the covariates `z1,z2,...,zp` for the time-to-event model. For $n$ subjects, it must must have $n$ rows.

`origin` the time origin (an $n$ x 1 vector for $n$ subjects; typically, a vector of zeros).

`end.time` the end of follow-up time (an $n$ x 1 vector for $n$ subjects).

## Details

Simulate recurrent event data. See Section 2.4 of Cook and Lawless.

## Value

a data frame containing `id`, `start`, `stop`, `status` and covariates.

## Author(s)

Shahedul Khan <khan@math.usask.ca>

## References

Cook RJ and Lawless J, The statistical analysis of recurrent events, Springer, 2007.

## See Also

`etime.reg`

## Examples

```
n <- 100 # number of subjects
Data <- data.frame(z1 = rnorm(n), z2 = rbinom(n, 1, 0.5)) # data frame
origin <- rep(0, n) # time origin for n subjects
end.time <- runif(n, 10, 15) # end of follow-up time
# Simulate data from the generalized gamma AFT model
surv.par <- list(beta = c(0.5, -0.5), logrho = log(0.025),
    logkappa = log(2), loggamma = log(0.4))
Data.gg <- sim.rec( ~ z1 + z2, surv.par, "ggamma", Data, origin, end.time)
# Fit the generalized gamma AFT model
fit.gg <- etime.reg(c(start, stop, status) ~ z1 + z2,
            surv.model = "ggamma", Data = Data.gg)
fit.gg
# Simulate data from the Weibull PH model
surv.par <- list(beta = c(0.5, -0.5), logrho = log(0.1), logkappa = log(2))
Data.wph <- sim.rec( ~ z1 + z2, surv.par, "weibullph", Data, origin, end.time)
# Fit the Weibull PH model
fit.wph <- etime.reg(c(start, stop, status) ~ z1 + z2,
            surv.model = "weibullph", Data = Data.wph)
fit.wph
```

# Index