

# Package ‘DTEAssurance’

May 7, 2026

**Type** Package

**Title** Assurance Methods for Clinical Trials with a Delayed Treatment Effect

**Version** 1.1.0

**Description** Provides functions for planning clinical trials subject to a delayed treatment effect using assurance-based methods. Includes two 'shiny' applications for interactive exploration, simulation, and visualisation of trial designs and outcomes. The methodology is described in:  
Salsbury JA, Oakley JE, Julious SA, Hampson LV (2024)  
`` Assurance methods for designing a clinical trial with a delayed treatment effect" <[doi:10.1002/sim.10136](https://doi.org/10.1002/sim.10136)>,  
Salsbury JA, Oakley JE, Julious SA, Hampson LV (2024)  
`` Adaptive clinical trial design with delayed treatment effects using elicited prior distributions" <[doi:10.48550/arXiv.2509.07602](https://doi.org/10.48550/arXiv.2509.07602)>.

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.3.3

**Imports** SHELF, shiny, stats, survival, nleqslv, nph, nphRCT, dplyr, rjags, rpact, magrittr, rlang, future.apply

**Suggests** testthat (>= 3.0.0)

**Config/testthat/edition** 3

**Depends** R (>= 4.1.0)

**Date** 2025-12-02

**URL** <https://jamesalsbury.github.io/DTEAssurance/>

**NeedsCompilation** no

**Author** James Salsbury [aut, cre] (ORCID:  
<<https://orcid.org/0000-0002-2584-3640>>)

**Maintainer** James Salsbury <[jsalsbury1@sheffield.ac.uk](mailto:jsalsbury1@sheffield.ac.uk)>

**Repository** CRAN

**Date/Publication** 2025-12-02 12:20:02 UTC

## Contents

add_recruitment_time . . . . .	2
assurance_adaptive_shiny_app . . . . .	3
assurance_shiny_app . . . . .	4
BPP_func . . . . .	4
calc_dte_assurance . . . . .	6
calc_dte_assurance_adaptive . . . . .	8
calibrate_BPP_threshold . . . . .	11
calibrate_BPP_timing . . . . .	13
cens_data . . . . .	15
INTEREST . . . . .	16
MCMC_sample . . . . .	17
REVEL . . . . .	17
sim_dte . . . . .	18
survival_test . . . . .	19
update_priors . . . . .	20
ZODIAC . . . . .	22
<b>Index</b>	<b>23</b>

---

add\_recruitment\_time *Add recruitment time to a survival dataset*

---

### Description

Simulates recruitment timing for each patient in a survival dataset using either a power model or a piecewise constant (PWC) model. The function appends recruitment times and pseudo survival times (time from recruitment to event or censoring).

### Usage

```
add_recruitment_time(
  data,
  rec_method,
  rec_period = NULL,
  rec_power = NULL,
  rec_rate = NULL,
  rec_duration = NULL
)
```

### Arguments

data	A dataframe containing survival data with columns: time, status, and group
rec_method	Recruitment method: "power" for power model or "PWC" for piecewise constant model
rec_period	Period length for the power model

rec_power	Power parameter for the power model
rec_rate	Comma-separated string of recruitment rates for the PWC model
rec_duration	Comma-separated string of durations corresponding to each rate in the PWC model

**Value**

A dataframe with two additional columns:

**rec\_time** Simulated recruitment time for each patient

**pseudo\_time** Time from recruitment to event or censoring

Class: data.frame

**Examples**

```
set.seed(123)
df <- data.frame(
  time = rexp(20, rate = 0.1),
  status = rbinom(20, 1, 0.8),
  group = rep(c("Control", "Treatment"), each = 10)
)
recruited <- add_recruitment_time(df, rec_method = "power", rec_period = 12, rec_power = 1)
head(recruited)
```

---

assurance\_adaptive\_shiny\_app

*Launch the 'shiny' adaptive assurance app*

---

**Description**

Launches a 'shiny' application to simulate group sequential trials with delayed treatment effects (DTE) using elicited prior distributions. The app allows interactive exploration of trial designs and assurance calculations.

**Usage**

```
assurance_adaptive_shiny_app()
```

**Value**

No return value, called for side effects (invisibly returns NULL). The function launches an interactive 'shiny' application.

**Examples**

```
# Launch the interactive Shiny app
assurance_adaptive_shiny_app()
```

assurance\_shiny\_app    *Launch the 'shiny' Assurance app*

---

### Description

Launches a 'shiny' application to calculate assurance for clinical trials where delayed treatment effects (DTE) may be present. The app allows elicitation of prior distributions and calculates assurance metrics.

### Usage

```
assurance_shiny_app()
```

### Value

No return value, called for side effects (invisibly returns NULL). The function launches an interactive 'shiny' application.

### Examples

```
# Launch the interactive Shiny app
assurance_shiny_app()
```

---

BPP\_func                    *Calculate Bayesian Predictive Probability given interim data and posterior samples*

---

### Description

Calculate Bayesian Predictive Probability given interim data and posterior samples

### Usage

```
BPP_func(
  data,
  posterior_df,
  control_distribution = "Exponential",
  n_c_planned,
  n_t_planned,
  rec_time_planned,
  df_cens_time,
  censoring_model,
  analysis_model,
  n_sims = 500
)
```

**Arguments**

<code>data</code>	A data frame containing interim survival data, censored at <code>df_cens_time</code> , with columns: <ul style="list-style-type: none"> <li>• <code>time</code> Final observed/event time at the interim (on the analysis time scale).</li> <li>• <code>group</code> Treatment group indicator (e.g. "Control", "Treatment").</li> <li>• <code>rec_time</code> Recruitment (calendar) time.</li> <li>• <code>pseudo_time</code> <code>time + rec_time</code> (calendar time at event/censoring).</li> <li>• <code>status</code> Event indicator at the interim (1 = event, 0 = censored).</li> <li>• <code>survival_time</code> Observed follow-up time from randomisation to event/censoring at the interim.</li> </ul>
<code>posterior_df</code>	A data frame of posterior samples with columns: <code>lambda_c</code> , <code>delay_time</code> and <code>HR</code> , corresponding to the control hazard, the delay (change point) time and the post-delay hazard ratio, respectively.
<code>control_distribution</code>	Distributional form assumed for the control arm: either "Exponential" (default) or "Weibull".
<code>n_c_planned</code>	Planned maximum number of patients in the control group.
<code>n_t_planned</code>	Planned maximum number of patients in the treatment group.
<code>rec_time_planned</code>	Planned maximum recruitment calendar time for the full trial.
<code>df_cens_time</code>	Calendar time at which <code>df</code> has been censored (interim analysis time).
<code>censoring_model</code>	A named list specifying the censoring mechanism for the future data: <ul style="list-style-type: none"> <li>• <code>method</code>: one of "Time", "Events", or "IF".</li> <li>• <code>time</code>, <code>events</code>, <code>IF</code>: parameters for the corresponding method.</li> </ul>
<code>analysis_model</code>	A named list specifying the final analysis and decision rule: <ul style="list-style-type: none"> <li>• <code>method</code>: e.g. "LRT", "WLRT", or "MW".</li> <li>• <code>alpha</code>: one-sided type I error level.</li> <li>• <code>alternative_hypothesis</code>: direction of the alternative (e.g. "one.sided").</li> <li>• <code>rho</code>, <code>gamma</code>, <code>t_star</code>, <code>s_star</code>: additional parameters for WLRT or MW (if applicable).</li> </ul>
<code>n_sims</code>	Number of predictive simulations to run (default is 1000).

**Value**

A single numeric value giving the Bayesian predictive probability of success at the final analysis under the specified design, censoring model and analysis model.

**Examples**

```
set.seed(123)
n <- 30
cens_time <- 15
```

```

time <- runif(n, 0, 12)
rec_time <- runif(n, 0, 12)

df <- data.frame(
  time = time,
  group = c(rep("Control", n/2), rep("Treatment", n/2)),
  rec_time = rec_time
)

df$pseudo_time <- df$time + df$rec_time
df$status <- df$pseudo_time < cens_time
df$survival_time <- ifelse(df$status == TRUE, df$time, cens_time - df$rec_time)

posterior_df <- data.frame(HR = rnorm(20, mean = 0.75, sd = 0.05),
  delay_time = rep(0, 20),
  lambda_c = rnorm(20, log(2)/9, sd = 0.01))

censoring_model = list(method = "Time", time = 25)
analysis_model = list(method = "LRT",
  alpha = 0.025,
  alternative_hypothesis = "one.sided")

BPP_outcome <- BPP_func(df,
  posterior_df,
  control_distribution = "Exponential",
  n_c_planned = n/2,
  n_t_planned = n/2,
  rec_time_planned = 12, df_cens_time = 15,
  censoring_model = censoring_model,
  analysis_model = analysis_model,
  n_sims = 10)

```

---

calc\_dte\_assurance      *Calculate Assurance for a Trial with a Delayed Treatment Effect*

---

### Description

Simulates operating characteristics for a clinical trial under prior uncertainty about a delayed treatment effect. The function integrates beliefs about control survival, treatment delay, post-delay hazard ratio, recruitment, censoring, and analysis method to estimate assurance and other trial metrics.

### Usage

```

calc_dte_assurance(
  n_c,

```

```

n_t,
control_model,
effect_model,
censoring_model,
recruitment_model,
analysis_model,
n_sims = 1000
)

```

## Arguments

n_c	Vector of control group sample sizes
n_t	Vector of treatment group sample sizes
control_model	A named list specifying the control arm survival distribution: <ul style="list-style-type: none"> <li>• dist: Distribution type ("Exponential" or "Weibull")</li> <li>• parameter_mode: Either "Fixed" or "Distribution"</li> <li>• fixed_type: If "Fixed", specify as "Parameters" or "Landmark"</li> <li>• lambda, gamma: Scale and shape parameters</li> <li>• t1, t2: Landmark times</li> <li>• surv_t1, surv_t2: Survival probabilities at landmarks</li> <li>• t1_Beta_a, t1_Beta_b, diff_Beta_a, diff_Beta_b: Beta prior parameters</li> </ul>
effect_model	A named list specifying beliefs about the treatment effect: <ul style="list-style-type: none"> <li>• delay_SHELF, HR_SHELF: SHELF objects encoding beliefs</li> <li>• delay_dist, HR_dist: Distribution types ("hist" by default)</li> <li>• P_S: Probability that survival curves separate</li> <li>• P_DTE: Probability of delayed separation, conditional on separation</li> </ul>
censoring_model	A named list specifying the censoring mechanism: <ul style="list-style-type: none"> <li>• method: "Time", "Events", or "IF"</li> <li>• time, events, IF: Parameters for each method</li> </ul>
recruitment_model	A named list specifying the recruitment process: <ul style="list-style-type: none"> <li>• method: "power" or "PWC"</li> <li>• period, power: Parameters for power model</li> <li>• rate, duration: Comma-separated strings for PWC model</li> </ul>
analysis_model	A named list specifying the statistical test and decision rule: <ul style="list-style-type: none"> <li>• method: "LRT", "WLRT", or "MW"</li> <li>• alpha, alternative_hypothesis: Type I error and hypothesis direction</li> <li>• rho, gamma, t_star, s_star: Parameters for WLRT or MW</li> <li>• success_threshold_HR: Optional threshold for declaring success</li> </ul>
n_sims	Number of simulations to run (default = 1000)

**Value**

A named list containing:

**assurance** Estimated assurance (probability of success under prior uncertainty)

**CI** 95% confidence interval for assurance

**duration** Mean trial duration across simulations

**sample\_size** Mean sample size across simulations

**diagnostics** Additional diagnostics if success\_threshold\_HR is specified

Class: list

**Examples**

```
set.seed(123)
control_model <- list(dist = "Exponential", parameter_mode = "Fixed",
  fixed_type = "Parameters", lambda = 0.1)
effect_model <- list(delay_SHELF = SHELF::fitdist(c(3, 4, 5),
  probs = c(0.25, 0.5, 0.75), lower = 0, upper = 10),
  delay_dist = "gamma",
  HR_SHELF = SHELF::fitdist(c(0.55, 0.6, 0.7), probs = c(0.25, 0.5, 0.75), lower = 0, upper = 1.5),
  HR_dist = "gamma",
  P_S = 1, P_DTE = 0)
censoring_model <- list(method = "Time", time = 12)
recruitment_model <- list(method = "power", period = 12, power = 1)
analysis_model <- list(method = "LRT", alpha = 0.025, alternative_hypothesis = "two.sided")
result <- calc_dte_assurance(n_c = 300, n_t = 300,
  control_model = control_model,
  effect_model = effect_model,
  censoring_model = censoring_model,
  recruitment_model = recruitment_model,
  analysis_model = analysis_model,
  n_sims = 10)

str(result)
```

---

calc\_dte\_assurance\_adaptive

*Calculates operating characteristics for a Group Sequential Trial with  
a Delayed Treatment Effect*

---

**Description**

Simulates assurance and operating characteristics for a group sequential trial under prior uncertainty about a delayed treatment effect. The function integrates beliefs about control survival, treatment delay, post-delay hazard ratio, recruitment, and group sequential design (GSD) parameters.

**Usage**

```
calc_dte_assurance_adaptive(
  n_c,
  n_t,
  control_model,
  effect_model,
  recruitment_model,
  GSD_model,
  analysis_model = NULL,
  n_sims = 1000
)
```

**Arguments**

n_c	Control group sample size
n_t	Treatment group sample size
control_model	A named list specifying the control arm survival distribution: <ul style="list-style-type: none"> <li>• dist: Distribution type ("Exponential" or "Weibull")</li> <li>• parameter_mode: Either "Fixed" or "Distribution"</li> <li>• fixed_type: If "Fixed", specify as "Parameters" or "Landmark"</li> <li>• lambda, gamma: Scale and shape parameters</li> <li>• t1, t2: Landmark times</li> <li>• surv_t1, surv_t2: Survival probabilities at landmarks</li> <li>• t1_Beta_a, t1_Beta_b, diff_Beta_a, diff_Beta_b: Beta prior parameters</li> </ul>
effect_model	A named list specifying beliefs about the treatment effect: <ul style="list-style-type: none"> <li>• delay_SHELF, HR_SHELF: SHELF objects encoding beliefs</li> <li>• delay_dist, HR_dist: Distribution types ("hist" by default)</li> <li>• P_S: Probability that survival curves separate</li> <li>• P_DTE: Probability of delayed separation, conditional on separation</li> </ul>
recruitment_model	A named list specifying the recruitment process: <ul style="list-style-type: none"> <li>• method: "power" or "PWC"</li> <li>• period, power: Parameters for power model</li> <li>• rate, duration: Comma-separated strings for PWC model</li> </ul>
GSD_model	A named list specifying the group sequential design: <ul style="list-style-type: none"> <li>• events: Total number of events</li> <li>• alpha_spending: Cumulative alpha spending vector</li> <li>• alpha_IF: Information Fraction at which we look for efficacy</li> <li>• futility_type: beta (for beta-spending), BPP (for Bayesian Predictive Probability) or none</li> <li>• futility_IF: Information Fraction at which we look for futility</li> <li>• beta_spending: Cumulative beta spending vector</li> </ul>

- BPP\_threshold: BPP value at which we will stop for futility
- analysis\_model A named list specifying the final analysis and decision rule:
- method: e.g. "LRT", "WLRT", or "MW".
  - alpha: one-sided type I error level.
  - alternative\_hypothesis: direction of the alternative (e.g. "one.sided").
  - rho, gamma, t\_star, s\_star: additional parameters for WLRT or MW (if applicable).
- n\_sims Number of simulations to run (default = 1000)

### Value

A data frame with one row per simulated trial and the following columns:

**Trial** Simulation index

**IF** Information fraction label used at the decision point

**Decision** Interim decision outcome (e.g., "Continue", "Stop for efficacy", "Stop for futility")

**StopTime** Time at which the trial stopped or completed

**SampleSize** Total sample size at the time of decision

**Final\_Decision** Final classification of trial success based on the test statistic and threshold

Class: data.frame

### Examples

```
set.seed(123)
control_model <- list(dist = "Exponential", parameter_mode = "Fixed",
  fixed_type = "Parameters", lambda = 0.1)
effect_model <- list(P_S = 1, P_DTE = 0,
  HR_SHELF = SHELF::fitdist(c(0.6, 0.65, 0.7), probs = c(0.25, 0.5, 0.75), lower = 0, upper = 2),
  HR_dist = "gamma",
  delay_SHELF = SHELF::fitdist(c(3, 4, 5), probs = c(0.25, 0.5, 0.75), lower = 0, upper = 10),
  delay_dist = "gamma"
)
recruitment_model <- list(method = "power", period = 12, power = 1)
GSD_model <- list(events = 300, alpha_spending = c(0.0125, 0.025),
  alpha_IF = c(0.75, 1), futility_type = "none")
result <- calc_dte_assurance_adaptive(n_c = 300, n_t = 300,
  control_model = control_model,
  effect_model = effect_model,
  recruitment_model = recruitment_model,
  GSD_model = GSD_model,
  n_sims = 10)

str(result)
```

---

 calibrate\_BPP\_threshold

*Function to calculate the 'optimal' BPP threshold value*


---

## Description

Function to calculate the 'optimal' BPP threshold value

## Usage

```
calibrate_BPP_threshold(
  n_c,
  n_t,
  control_model,
  effect_model,
  recruitment_model,
  IA_model,
  analysis_model,
  data_generating_model,
  n_sims = 100
)
```

## Arguments

n_c	Number of control patients
n_t	Number of treatment patients
control_model	A named list specifying the control arm survival distribution: <ul style="list-style-type: none"> <li>• dist: Distribution type ("Exponential" or "Weibull")</li> <li>• parameter_mode: Either "Fixed" or "Distribution"</li> <li>• fixed_type: If "Fixed", specify as "Parameters" or "Landmark"</li> <li>• lambda, gamma: Scale and shape parameters</li> <li>• t1, t2: Landmark times</li> <li>• surv_t1, surv_t2: Survival probabilities at landmarks</li> <li>• t1_Beta_a, t1_Beta_b, diff_Beta_a, diff_Beta_b: Beta prior parameters</li> </ul>
effect_model	A named list specifying beliefs about the treatment effect: <ul style="list-style-type: none"> <li>• delay_SHELF, HR_SHELF: SHELF objects encoding beliefs</li> <li>• delay_dist, HR_dist: Distribution types ("hist" by default)</li> <li>• P_S: Probability that survival curves separate</li> <li>• P_DTE: Probability of delayed separation, conditional on separation</li> </ul>
recruitment_model	A named list specifying the recruitment process: <ul style="list-style-type: none"> <li>• method: "power" or "PWC"</li> </ul>



```

data_generating_model = list(lambda_c = log(2)/12,
                             delay_time = 3,
                             post_delay_HR = 0.75)

threshold <- calibrate_BPP_threshold(n_c = 15, n_t = 15,
                                   control_model = control_model,
                                   effect_model = effect_model,
                                   recruitment_model = recruitment_model,
                                   IA_model = IA_model,
                                   analysis_model = analysis_model,
                                   data_generating_model = data_generating_model,
                                   n_sims = 2)

```

---

calibrate\_BPP\_timing    *Function to calculate the 'optimal' information fraction to calculate BPP*

---

### Description

Function to calculate the 'optimal' information fraction to calculate BPP

### Usage

```

calibrate_BPP_timing(
  n_c,
  n_t,
  control_model,
  effect_model,
  recruitment_model,
  IA_model,
  analysis_model,
  n_sims = 50
)

```

### Arguments

n_c	Number of control patients
n_t	Number of treatment patients
control_model	A named list specifying the control arm survival distribution: <ul style="list-style-type: none"> <li>• dist: Distribution type ("Exponential" or "Weibull")</li> <li>• parameter_mode: Either "Fixed" or "Distribution"</li> <li>• fixed_type: If "Fixed", specify as "Parameters" or "Landmark"</li> <li>• lambda, gamma: Scale and shape parameters</li> <li>• t1, t2: Landmark times</li> </ul>

	<ul style="list-style-type: none"> <li>• <code>surv_t1</code>, <code>surv_t2</code>: Survival probabilities at landmarks</li> <li>• <code>t1_Beta_a</code>, <code>t1_Beta_b</code>, <code>diff_Beta_a</code>, <code>diff_Beta_b</code>: Beta prior parameters</li> </ul>
<code>effect_model</code>	<p>A named list specifying beliefs about the treatment effect:</p> <ul style="list-style-type: none"> <li>• <code>delay_SHELF</code>, <code>HR_SHELF</code>: SHELF objects encoding beliefs</li> <li>• <code>delay_dist</code>, <code>HR_dist</code>: Distribution types ("hist" by default)</li> <li>• <code>P_S</code>: Probability that survival curves separate</li> <li>• <code>P_DTE</code>: Probability of delayed separation, conditional on separation</li> </ul>
<code>recruitment_model</code>	<p>A named list specifying the recruitment process:</p> <ul style="list-style-type: none"> <li>• <code>method</code>: "power" or "PWC"</li> <li>• <code>period</code>, <code>power</code>: Parameters for power model</li> <li>• <code>rate</code>, <code>duration</code>: Comma-separated strings for PWC model</li> </ul>
<code>IA_model</code>	<p>A named list specifying the censoring mechanism for the future data:</p> <ul style="list-style-type: none"> <li>• <code>events</code>: Number of events which is 100% information fraction</li> <li>• <code>IF</code>: The information fraction at which to censor and calculate BPP</li> </ul>
<code>analysis_model</code>	<p>A named list specifying the final analysis and decision rule:</p> <ul style="list-style-type: none"> <li>• <code>method</code>: e.g. "LRT", "WLRT", or "MW".</li> <li>• <code>alpha</code>: one-sided type I error level.</li> <li>• <code>alternative_hypothesis</code>: direction of the alternative (e.g. "one.sided").</li> <li>• <code>rho</code>, <code>gamma</code>, <code>t_star</code>, <code>s_star</code>: additional parameters for WLRT or MW (if applicable).</li> </ul>
<code>n_sims</code>	Number of data sets to simulate (default is 100).

### Value

A vector of length `n_sims` corresponding to the value of BPP for each simulated trial

### Examples

```
#' set.seed(123)
control_model = list(dist = "Exponential",
                    parameter_mode = "Distribution",
                    t1 = 12,
                    t1_Beta_a = 20,
                    t1_Beta_b = 32)

effect_model = list(delay_SHELF = SHELF::fitdist(c(5.5, 6, 6.5),
                    probs = c(0.25, 0.5, 0.75), lower = 0, upper = 12),
                    delay_dist = "gamma",
                    HR_SHELF = SHELF::fitdist(c(0.5, 0.6, 0.7),
                    probs = c(0.25, 0.5, 0.75), lower = 0, upper = 1),
                    HR_dist = "gamma",
                    P_S = 1,
                    P_DTE = 0)
```

```

recruitment_model <- list(method = "power", period = 12, power = 1)

IA_model = list(events = 20, IF = 0.5)

analysis_model = list(method = "LRT",
                      alpha = 0.025,
                      alternative_hypothesis = "one.sided")

timing <- calibrate_BPP_timing(n_c = 15, n_t = 15,
                             control_model = control_model,
                             effect_model = effect_model,
                             recruitment_model = recruitment_model,
                             IA_model = IA_model,
                             analysis_model = analysis_model,
                             n_sims = 2)

```

---

cens\_data

*Censor a survival dataset*


---

## Description

Applies administrative censoring to a survival dataset using one of three methods: fixed time, fixed number of events, or fixed information fraction. The input data must contain columns for pseudo survival time, recruitment time, and observed time.

## Usage

```

cens_data(
  data,
  cens_method = "Time",
  cens_time = NULL,
  cens_IF = NULL,
  cens_events = NULL
)

```

## Arguments

data	A dataframe containing uncensored survival data with columns: pseudo_time, rec_time, and time
cens_method	Censoring method: "Time" (default), "Events", or "IF"
cens_time	Time point for censoring (required if cens_method = "Time")
cens_IF	Information fraction for censoring (required if cens_method = "IF")
cens_events	Number of events for censoring (required if cens_method = "Events")

**Value**

A list containing:

**data** Censored dataframe with updated status and filtered rows

**cens\_events** Number of events used for censoring (if applicable)

**cens\_time** Time point used for censoring

**sample\_size** Number of patients remaining after censoring

**Examples**

```
set.seed(123)
df <- data.frame(
  pseudo_time = rexp(20, rate = 0.1),
  rec_time = runif(20, 0, 12),
  time = rexp(20, rate = 0.1)
)
censored <- cens_data(df, cens_method = "Time", cens_time = 10)
str(censored)
```

---

INTEREST

*INTEREST data set*

---

**Description**

A reconstructed survival data set for the INTEREST clinical trial

**Usage**

INTEREST

**Format**

A data frame with 710 rows and 2 variables:

**Survival time** Survival Time (in months)

**Status** Event indicator (0=Alive, 1=Dead)

**Source**

Reconstructed survival data set from the following publication: <https://www.sciencedirect.com/science/article/pii/S01406736>

---

MCMC\_sample

*MCMC\_sample*


---

**Description**

An MCMC sample for the example given in Salsbury et al (2024)

**Usage**

MCMC\_sample

**Format**

A data frame with 100000 rows and 1 variables:

**x** Sample from the MAP prior

**Source**

A MCMC sample for the control group for the example given in <https://onlinelibrary.wiley.com/doi/full/10.1002/sim.10136>. Three historical data sets are used to generate a Meta-Analytic-Predictive Prior distribution

---

REVEL

*REVEL data set*


---

**Description**

A reconstructed survival data set for the REVEL clinical trial

**Usage**

REVEL

**Format**

A data frame with 625 rows and 2 variables:

**Survival time** Survival Time (in months)

**Status** Event indicator (0=Alive, 1=Dead)

**Source**

Reconstructed survival data set from the following publication: [https://www.thelancet.com/journals/lancet/article/PIIS0140-6736\(14\)60845-X/fulltext](https://www.thelancet.com/journals/lancet/article/PIIS0140-6736(14)60845-X/fulltext)

---

sim_dte	<i>Simulates survival times for a delayed treatment effect (DTE) scenario, where the treatment group experiences a delayed onset of benefit. Control and treatment groups are generated under exponential or Weibull distributions.</i>
---------	---

---

### Description

Simulates survival times for a delayed treatment effect (DTE) scenario, where the treatment group experiences a delayed onset of benefit. Control and treatment groups are generated under exponential or Weibull distributions.

### Usage

```
sim_dte(
  n_c,
  n_t,
  lambda_c,
  delay_time,
  post_delay_HR,
  dist = "Exponential",
  gamma_c = NULL
)
```

### Arguments

n_c	The number of patients in the control group
n_t	The number of patients in the treatment group
lambda_c	The baseline hazard rate for the control group
delay_time	The length of delay before treatment effect begins
post_delay_HR	The hazard ratio after the delay period
dist	The distribution for the control group; must be one of "Exponential" (default) or "Weibull"
gamma_c	The shape parameter for the Weibull distribution (only used if dist = "Weibull")

### Value

A data frame with two columns:

time	Simulated survival times
group	Group assignment: "Control" or "Treatment"

Class: data.frame

**Examples**

```
set.seed(123)
sim_data <- sim_dte(n_c = 10, n_t = 10, lambda_c = 0.1,
                  delay_time = 6, post_delay_HR = 0.6)
head(sim_data)
```

---

survival_test	<i>Calculate statistical significance on a survival dataset</i>
---------------	---

---

**Description**

Performs a survival analysis using either the standard log-rank test (LRT) or a weighted log-rank test (WLRT). The function estimates the hazard ratio and determines whether the result is statistically significant based on the specified alpha level and alternative hypothesis.

**Usage**

```
survival_test(
  data,
  analysis_method = "LRT",
  alternative = "one.sided",
  alpha = 0.05,
  rho = 0,
  gamma = 0,
  t_star = NULL,
  s_star = NULL
)
```

**Arguments**

data	A dataframe containing survival data. Must include columns for survival time, event status, and treatment group.
analysis_method	Method of analysis: "LRT" (default) for standard log-rank test, or "WLRT" for weighted log-rank test.
alternative	String specifying the alternative hypothesis. Must be one of "one.sided" or "two.sided" (default).
alpha	Type I error threshold for significance testing.
rho	Rho parameter for the Fleming-Harrington weighted log-rank test.
gamma	Gamma parameter for the Fleming-Harrington weighted log-rank test.
t_star	Parameter $t^*$ used in modestly weighted tests.
s_star	Parameter $s^*$ used in modestly weighted tests.

**Value**

A list containing:

**Signif** Logical indicator of statistical significance based on the chosen test and alpha level.

**observed\_HR** Estimated hazard ratio from a Cox proportional hazards model.

**Examples**

```
set.seed(123)
df <- data.frame(
  survival_time = rexp(40, rate = 0.1),
  status = rbinom(40, 1, 0.8),
  group = rep(c("Control", "Treatment"), each = 20)
)
result <- survival_test(df, analysis_method = "LRT", alpha = 0.05)
str(result)
```

---

 update\_priors

---

*Update prior distributions using interim survival data*


---

**Description**

This function updates elicited priors (defined through SHELF objects and parametric prior distributions) using interim survival data under a delayed-effect, piecewise-exponential model for the treatment arm and an exponential or Weibull model for the control arm.

**Usage**

```
update_priors(data, control_model, effect_model, n_samples = 1000)
```

**Arguments**

data	A data frame containing interim survival data with columns: <ul style="list-style-type: none"> <li>• survival_time Observed time from randomisation to event/censoring.</li> <li>• status Event indicator (1 = event, 0 = censored).</li> <li>• group Group identifier (e.g., "Control", "Treatment").</li> </ul>
control_model	A named list specifying the control arm survival distribution: <ul style="list-style-type: none"> <li>• dist: Distribution type ("Exponential" or "Weibull")</li> <li>• parameter_mode: Either "Fixed" or "Distribution"</li> <li>• fixed_type: If "Fixed", specify as "Parameters" or "Landmark"</li> <li>• lambda, gamma: Scale and shape parameters</li> <li>• t1, t2: Landmark times</li> <li>• surv_t1, surv_t2: Survival probabilities at landmarks</li> <li>• t1_Beta_a, t1_Beta_b, diff_Beta_a, diff_Beta_b: Beta prior parameters</li> </ul>

effect_model	A named list specifying beliefs about the treatment effect: <ul style="list-style-type: none"> <li>• delay_SHELF, HR_SHELF: SHELF objects encoding beliefs</li> <li>• delay_dist, HR_dist: Distribution types ("hist" by default)</li> <li>• P_S: Probability that survival curves separate</li> <li>• P_DTE: Probability of delayed separation, conditional on separation</li> </ul>
n_samples	Number of posterior samples to generate (default: 1000).

### Value

A data frame containing Monte Carlo samples from the updated (posterior) distribution of the model parameters. Columns normally include:

- lambda\_c Posterior samples for the control hazard parameter.
- delay\_time Posterior samples for the delay/changepoint time  $T$ .
- HR Posterior samples for the post-delay hazard ratio.
- gamma\_c (only if control\_distribution = "Weibull") Posterior samples for the Weibull shape parameter.

Priors for lambda\_c, T, and HR are constructed from elicited distributions using the SHELF framework, then updated through sampling-based posterior inference.

### Examples

```
set.seed(123)
interim_data = data.frame(survival_time = runif(10, min = 0, max = 10),
  status = rbinom(10, size = 1, prob = 0.5),
  group = c(rep("Control", 5), rep("Treatment", 5)))
control_model = list(dist = "Exponential",
  parameter_mode = "Distribution",
  t1 = 12,
  t1_Beta_a = 20,
  t1_Beta_b = 32)

effect_model = list(delay_SHELF = SHELF::fitdist(c(5.5, 6, 6.5),
  probs = c(0.25, 0.5, 0.75), lower = 0, upper = 12),
  delay_dist = "gamma",
  HR_SHELF = SHELF::fitdist(c(0.5, 0.6, 0.7),
  probs = c(0.25, 0.5, 0.75), lower = 0, upper = 1),
  HR_dist = "gamma",
  P_S = 1,
  P_DTE = 0)

posterior_df <- update_priors(
  data = interim_data,
  control_model = control_model,
  effect_model = effect_model,
  n_samples = 10)
```

---

ZODIAC

*ZODIAC data set*

---

**Description**

A reconstructed survival data set for the ZODIAC clinical trial

**Usage**

ZODIAC

**Format**

A data frame with 697 rows and 2 variables:

**Survival time** Survival Time (in months)

**Status** Event indicator (0=Alive, 1=Dead)

**Source**

Reconstructed survival data set from the following publication: <https://www.sciencedirect.com/science/article/abs/pii/S14702>

# Index

## \* datasets

INTEREST, [16](#)

MCMC\_sample, [17](#)

REVEL, [17](#)

ZODIAC, [22](#)

add\_recruitment\_time, [2](#)

assurance\_adaptive\_shiny\_app, [3](#)

assurance\_shiny\_app, [4](#)

BPP\_func, [4](#)

calc\_dte\_assurance, [6](#)

calc\_dte\_assurance\_adaptive, [8](#)

calibrate\_BPP\_threshold, [11](#)

calibrate\_BPP\_timing, [13](#)

cens\_data, [15](#)

INTEREST, [16](#)

MCMC\_sample, [17](#)

REVEL, [17](#)

sim\_dte, [18](#)

survival\_test, [19](#)

update\_priors, [20](#)

ZODIAC, [22](#)