Package 'agpower'

May 26, 2025

Title Recurrent Event Analysis Planning Using the Andersen-Gill Model with Robust Standard Errors

Version 0.1.2

Description Package implements functions useful in prospective planning and monitoring of a clinical trial when a recurrent event endpoint is to be assessed by Lin, Wei, Yang, and Ying (2010) model. The equations developed in Ingel and Jahn-Eimermacher (2014) and their consequences are employed.

License Apache License (>= 2)

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Imports stats

Suggests dplyr, testthat (>= 3.0.0), tidyr

Config/testthat/edition 3

NeedsCompilation no

Author Stephen Rush [aut, cre]

Maintainer Stephen Rush <rushacademic@gmail.com>

R topics documented:

`alpNeeded` *Function to compute alpha needed (fixed sample size)*

Description

Function to compute two-sided alpha needed to achieve target power given a rate ratio. Useful for computing probability to achieve hurdles.

Usage

```
alpNeeded(N, bta1, thta, tau, lam0, pow = 0.8, ar = 0.5)
```

Arguments

`N`	Sample size.
`bta1`	log-transform of rate ratio.
`thta`	Variance of frailty parameter.
`tau`	Expected follow-up time.
`lam0`	Event rate for control.
`pow`	Target power.
`ar`	Allocation ratio (Number control / Total)

Details

This function computes the two-sided alpha alp. Function assumes a rate ratio < 1 is favourable to treatment.

Value

The two-sided alpha level.

Examples

```
# alpha needed to achieve multiple powers given rate ratio (and other input).
alpNeeded(N = 1000, bta1 = log(0.8), thta = 2, tau = 1, lam0 = 1.1, pow = c( .7, .8))

# alpha needed for many inputs
if (require("dplyr") & require("tidyr")) {

  assumptions = tibble(RR = 0.8) %>%
    crossing(
      thta = c(2, 3, 4),
      lam0 = 1.1,
      pow = c(0.7, 0.8),
      N = c(500, 1000),
      tau = 1
    ) %>%
    mutate(
     alp = alpNeeded(N = N, bta1 = log(RR), thta = thta, tau = tau, lam0 = lam0, pow = pow)
    )
```

```
  assumptions %>% data.frame()

}
```

---

alpNeeded2 *Function to compute alpha needed (fixed sample size)*

Description

Function to compute two-sided alpha needed to achieve target power given a rate ratio. Useful for computing probability to achieve hurdles.

Usage

```
alpNeeded2(N, bta1, thta, L, pow = 0.8, ar = 0.5)
```

Arguments

N	Sample size.
bta1	log-transform of rate ratio.
thta	Variance of frailty parameter.
L	Number of events.
pow	Target power.
ar	Allocation ratio (Number control / Total).

Details

This function computes the two-sided alpha alp. Function assumes a rate ratio < 1 is favourable to treatment.

Value

The two-sided alpha level.

Examples

```
# alpha needed to achieve multiple powers given rate ratio (and other input).
alpNeeded2(N = 1000, bta1 = log(0.8), thta = 2, L = 1000, pow = c( .7, .8))

# alpha needed for many inputs
if (require("dplyr") & require("tidyr")) {


}
```

assurance *Assurance (expected power) for LWYY*

Description

Function to compute assurance given fixed sample size N and follow-up tau, under input assumptions. Assumes a log-normal prior distribution. Here assurance is taken to mean probability to achieve statistical significance (p-value < alp).

Usage

```
assurance(
  N,
  bta1,
  bta1_sd,
  thta,
  tau,
  lam,
  alp = 0.05,
  ar = 0.5,
  ns = 1000,
  method = c("integration", "montecarlo"),
  frailty.type = c("unblind", "blind"),
  lam.type = c("base", "pool"),
  thtawarning = FALSE
)
```

Arguments

N	Sample size.
bta1	log-transform of rate ratio.
bta1_sd	assumed standard deviation of log(rate ratio)
thta	Variance of frailty parameter. If frailty.type = "blind", assumes thta derives from pooled model; if frailty.type = "unblind" assumes thta is from correctly specified model. Default "unblind".
tau	Expected follow-up time.
lam	Event rate. If lam.type = "pool", assumes lam is pooled rate; if lam.type = "base", assumes lam is baseline control event rate. Default "base".
alp	Two-sided alpha-level.
ar	Allocation ratio (Number control / Total).
ns	Maximum number of subintervals (if method = "integration") or Number of draws from prior distribution (if method = "montecarlo").
method	Whether to numerically solve ("integration") or estimate by random draws ("montecarlo"). Defaults to numerical.
frailty.type	Indicates whether frailty variance is based on blinded information ("blind") or unblinded ("unblind"). Default "unblind".
lam.type	Indicates whether event rate is based on control rate ("base") or pooled rate ("pool"). Default "base".
thtawarning	If TRUE indicates how many estimates of theta were negative before setting to 0. Default FALSE.

Details

If working with a blinded estimate of frailty variance (i.e. misspecified model), it is recommended to use frailty.type = "blind" and lam.type = "pool". In which case the frailty variance (i.e. model with treatment effect) is derived using thta and the quantile drawn from the prior distribution of the log-rate ratio. If working with an estimate of frailty variance, should use frailty.type = "unblind" instead.

Function assumes a rate ratio < 1 is favourable to treatment.

Value

The assurance given the input assumptions.

Examples

```
assurance(N = 500, bta1 = log(0.8), bta1_sd = 1, thta = 2, tau = 1, lam = 1.1, alp = 0.05,
          ns = 100000)


if (require("dplyr") & require("tidyr")) {

  assumptions = tibble(alp = 0.05) %>%
  crossing(
    N = c(500, 1000),
    RR = c(0.6, 0.7, 0.8),
    bta1_sd = 1,
    thta = c(2, 3, 4),
    tau = c(0.8,0.9, 1.0),
    lam0 = c(3, 3.5)
  ) %>%
   mutate(pow = pow(N = N, bta1 = log(RR), thta = thta, tau = tau, lam = lam0, alp = alp)) %>%
    mutate(
     assurance_in_blind = assurance(N = N, bta1 = log(RR), bta1_sd = bta1_sd, thta = thta,
                                    tau = tau, lam = lam0, alp = alp, ns = 1000,
                                    frailty.type = "blind", lam.type = "pool")
    ) %>%
    mutate(
     assurance_mc_blind = assurance(N = N, bta1 = log(RR), bta1_sd = bta1_sd, thta = thta,
                                    tau = tau, lam = lam0, alp = alp, ns = 1000,
                                    method = "monte",
                                    frailty.type = "blind", lam.type = "pool")
    ) %>%
    mutate(
     assurance_in_unblind = assurance(N = N, bta1 = log(RR), bta1_sd = bta1_sd, thta = thta,
                                      tau = tau, lam = lam0, alp = alp, ns = 1000)
    ) %>%
    mutate(
     assurance_mc_unblind = assurance(N = N, bta1 = log(RR), bta1_sd = bta1_sd, thta = thta,
                                      tau = tau, lam = lam0, alp = alp, ns = 1000)
    )

  assumptions %>% data.frame()

}
```

assurance2 *Assurance (expected power) for LWYY*

Description

Function to compute assurance given fixed sample size N and number of events, under input assumptions. Assumes a log-normal prior distribution. Here assurance is taken to mean probability to achieve statistical significance (p-value < alp).

Usage

```
assurance2(
  N,
  bta1,
  bta1_sd,
  thta,
  L,
  alp = 0.05,
  ar = 0.5,
  ns = 1000,
  method = c("integration", "montecarlo"),
  frailty.type = c("unblind", "blind"),
  thtawarning = FALSE
)
```

Arguments

N	Sample size.
bta1	log-transform of rate ratio.
bta1_sd	assumed standard deviation of log(rate ratio)
thta	Variance of frailty parameter. If frailty.type = "blind", assumes thta derives from pooled model; if frailty.type = "unblind" assumes thta is from correctly specified model. Default "unblind".
L	Number of events
alp	Two-sided alpha-level.
ar	Allocation ratio (Number control / Total).
ns	Maximum number of subintervals (if method = "integration") or Number of draws from prior distribution (if method = "montecarlo").
method	Whether to numerically solve ("integration") or estimate by random draws ("montecarlo"). Defaults to numerical.
frailty.type	Indicates whether frailty variance is based on blinded information ("blind") or unblinded ("unblind"). Default "unblind".
thtawarning	If TRUE indicates how many estimates of theta were negative before setting to 0. Default FALSE.

Details

If working with a blinded estimate of frailty variance (i.e. misspecified model), it is recommended to use frailty.type = "blind". In which case the frailty variance (i.e. model with treatment effect) is derived using thta and the quantile drawn from the prior distribution of the log-rate ratio. If working with an estimate of frailty variance, should use frailty.type = "unblind" instead.

Function assumes a rate ratio < 1 is favourable to treatment.

Value

The assurance given the input assumptions.

Examples

```
assurance2(N = 500, bta1 = log(0.8), bta1_sd = 1, thta = 2, L = 1000, alp = 0.05,
          ns = 100000)


if (require("dplyr") & require("tidyr")) {

  assumptions = tibble(alp = 0.05) %>%
  crossing(
    N = c(500, 1000),
    RR = c(0.6, 0.7, 0.8),
    bta1_sd = 1,
    thta = c(2, 3, 4),
    L = c(500, 1000, 1500)
  ) %>%
    mutate(pow = pow2(N = N, bta1 = log(RR), thta = thta, L = L, alp = alp)) %>%
    mutate(
     assurance_in_blind = assurance2(N = N, bta1 = log(RR), bta1_sd = bta1_sd, thta = thta,
                                     L = L, alp = alp, ns = 1000,
                                     frailty.type = "blind")
    ) %>%
    mutate(
     assurance_mc_blind = assurance2(N = N, bta1 = log(RR), bta1_sd = bta1_sd, thta = thta,
                                     L = L, alp = alp, ns = 1000,
                                     method = "monte",
                                     frailty.type = "blind")
    ) %>%
    mutate(
    assurance_in_unblind = assurance2(N = N, bta1 = log(RR), bta1_sd = bta1_sd, thta = thta,
                                       L = L, alp = alp, ns = 1000)
    ) %>%
    mutate(
    assurance_mc_unblind = assurance2(N = N, bta1 = log(RR), bta1_sd = bta1_sd, thta = thta,
                                       L = L, alp = alp, ns = 1000)
    )

  assumptions %>% data.frame()

}
```

`btaNeeded` *Function to compute log rate ratio needed (fixed sample size)*

Description

Function to compute log rate ratio needed to achieve target power at one-sided Type I control level alp/2. Useful to compute critical value (set pow = 0.5).

Usage

```
btaNeeded(
  N,
  thta,
  tau,
  lam,
  alp = 0.05,
  pow = 0.8,
  ar = 0.5,
  frailty.type = c("unblind", "blind"),
  lam.type = c("base", "pool"),
  interval = c(log(0.5), log(1))
)
```

Arguments

`N`	Sample size.
`thta`	Variance of frailty parameter. If frailty.type = "blind", assumes thta derives from pooled model; if frailty.type = "unblind" assumes thta is from correctly specified model. Default "unblind".
`tau`	Expected follow-up time.
`lam`	Event rate. If lam.type = "pool", assumes lam is pooled rate; if lam.type = "base", assumes lam is baseline control event rate. Default "base".
`alp`	Two-sided alpha-level.
`pow`	Target power.
`ar`	Allocation ratio (Number control / Total)
`frailty.type`	Indicates whether frailty variance is based on blinded information ("blind") or unblinded ("unblind"). Default "unblind".
`lam.type`	Indicates whether event rate is based on control rate ("base") or pooled rate ("pool"). Default "base".
`interval`	Initial search interval for bta1.

Details

This function computes the log rate ratio bta1 as the root of the equation pow(bta1, N, thta_, tau, lam0_, alp, ar) - pow = 0, where thta_ and lam0_ also depend on bta1 if using estimates from blinded analyses. If frailty.type = "blind": thta_ = thtap2thta(thta, bta1); otherwise if frailty = "unblind": thta_ = thta. If lam.type = "pool" then lam0_ = lam * 2 / (1 + exp(bta1)); otherwise if lam.type = "base": lam0_ = lam. Function assumes a rate ratio < 1 is favourable to treatment.

Value

The log rate ratio.

Examples

```
# Based on unblinded estimates
btaNeeded(N = 1000, thta = 2, tau = 1, lam = 1.1, alp = c(0.01, 0.05), pow = c(.5))
exp(btaNeeded(N = 1000, thta = 2, tau = 1, lam = 1.1, alp = c(0.01, 0.05), pow = c(.5)))

# Based on blinded estimates
btaNeeded(N = 1000, thta = 2, tau = 1, lam = 0.7, alp = c(0.01, 0.05), pow = c( .5),
  frailty.type = "bl", lam.type = "po")
exp(btaNeeded(N = 1000, thta = 2, tau = 1, lam = 0.7, alp = c(0.01, 0.05), pow = c( .5),
               frailty.type = "bl", lam.type = "po"))

# Based on blinded estimates
if (require("dplyr") & require("tidyr")) {

  assumptions = tibble(alp = 0.05) %>%
    crossing(
      thta = c(2, 3, 4),
      lam = 1.1,
      pow = c(0.5, 0.8),
      N = c(500, 1000),
      tau = 1
    ) %>%
    mutate(
      bta1 = btaNeeded(N = N, thta = thta, tau = tau, lam = lam, alp = alp, pow = pow),
      RR = exp(bta1)
    )
  assumptions %>% data.frame()

}
```

`btaNeeded2` *Function to compute log rate ratio needed (fixed sample size)*

Description

Function to compute log rate ratio needed to achieve target power at one-sided Type I control level alp/2. Useful to compute critical value (set pow = 0.5).

Usage

```
btaNeeded2(
  N,
  thta,
  L,
  alp = 0.05,
  pow = 0.8,
  ar = 0.5,
  frailty.type = c("unblind", "blind"),
```

```
  interval = c(log(0.5), log(1))
)
```

Arguments

N	Sample size.
thta	Variance of frailty parameter. If frailty.type = "blind", assumes thta derives from pooled model; if frailty.type = "unblind" assumes thta is from correctly specified model. Default "unblind".
L	Number of events
alp	Two-sided alpha-level.
pow	Target power.
ar	Allocation ratio (Number control / Total)
frailty.type	Indicates whether frailty variance is based on blinded information ("blind") or unblinded ("unblind"). Default "unblind".
interval	Initial search interval for bta1.

Details

This function computes the log rate ratio bta1 as the root of the equation pow2(bta1, N, thta_, L, alp, ar) - pow = 0, where thta_ depends on bta1 if using estimates from blinded analyses. If frailty.type = "blind": thta_ = thtap2thta(thta, bta1); otherwise if frailty = "unblind": thta_ = thta. Function assumes a rate ratio < 1 is favourable to treatment.

Value

The log rate ratio.

Examples

```
# Based on unblinded estimates

# Based on blinded estimates
btaNeeded2(N = 1000, thta = 2, L = 1000, alp = c(0.01, 0.05), pow = c(.5), frailty.type = "bl")
exp(btaNeeded2(N = 1000, thta = 2, L = 1000, alp = c(0.01, 0.05), pow = c(.5), frailty.type = "bl"))
```

---

btaNeeded3 *Function to compute log rate ratio needed*

Description

Function to compute log rate ratio needed to achieve target power at one-sided Type I control level alp/2. Useful to compute critical value (set pow = 0.5).

Usage

```
btaNeeded3(
  thta,
  L,
  tau,
  lam,
  alp = 0.05,
  pow = 0.8,
  ar = 0.5,
  frailty.type = c("unblind", "blind"),
  lam.type = c("base", "pool"),
  interval = c(log(0.5), log(1))
)
```

Arguments

thta	Variance of frailty parameter. If frailty.type = "blind", assumes thta derives from pooled model; if frailty.type = "unblind" assumes thta is from correctly specified model. Default "unblind".
L	Number of events.
tau	Expected follow-up time.
lam	Event rate. If lam.type = "pool", assumes lam is pooled rate; if lam.type = "base", assumes lam is baseline control event rate. Default "base".
alp	Two-sided alpha-level.
pow	Target power.
ar	Allocation ratio (Number control / Total)
frailty.type	Indicates whether frailty variance is based on blinded information ("blind") or unblinded ("unblind"). Default "unblind".
lam.type	Indicates whether event rate is based on control rate ("base") or pooled rate ("pool"). Default "base".
interval	Initial search interval for bta1.

Details

This function computes the log rate ratio bta1 as the root of the equation eventsNeeded(bta1, thta_, tau, lam0_, alp, pow, ar) - L = 0, where thta_ and lam0_ also depend on bta1 if using estimates from blinded analyses. If frailty.type = "blind": thta_ = thtap2thta(thta, bta1); otherwise if frailty = "unblind": thta_ = thta. If lam.type = "pool" then lam0_ = lam * 2 / (1 + exp(bta1)); otherwise if lam.type = "base": lam0_ = lam. Function assumes a rate ratio < 1 is favourable to treatment.

Value

The log rate ratio.

Examples

```
# Based on unblinded estimates
btaNeeded3(thta = 2, L = 1000, tau = 1, lam = 1.1, alp = c(0.01, 0.05), pow = c(.5))
exp(btaNeeded3(thta = 2, L = 1000, tau = 1, lam = 1.1, alp = c(0.01, 0.05), pow = c(.5)))
```

```
# Based on blinded estimates
btaNeeded3(thta = 2, L = 1000, tau = 1, lam = 0.7, alp = c(0.01, 0.05), pow = c( .5),
  frailty.type = "bl", lam.type = "po")
exp(btaNeeded3(thta = 2, L = 1000, tau = 1, lam = 0.7, alp = c(0.01, 0.05), pow = c( .5),
               frailty.type = "bl", lam.type = "po"))

# Based on blinded estimates
if (require("dplyr") & require("tidyr")) {

  assumptions = tibble(alp = 0.05) %>%
    crossing(
      thta = c(2, 3, 4),
      lam = 1.1,
      pow = c(0.5, 0.8),
      L = c(500, 1000),
      tau = 1
    ) %>%
    mutate(
      bta1 = btaNeeded3(thta = thta, L = L, tau = tau, lam = lam, alp = alp, pow = pow),
      RR = exp(bta1)
    )
  assumptions %>% data.frame()

}
```

---

erNeeded *Baseline event rate needed (fixed sample size)*

Description

Function to compute baseline (control) event rate needed to achieve given power at one-sided Type I control level alp/2.

Usage

```
erNeeded(
  N,
  bta1,
  thta,
  tau,
  alp = 0.05,
  pow = 0.8,
  ar = 0.5,
  lam0warning = FALSE
)
```

Arguments

N	Sample size.
bta1	log-transform of rate ratio.
thta	Variance of frailty parameter.

tau	Expected follow-up time.
alp	Two-sided alpha-level.
pow	Target power.
ar	Allocation ratio (Number control / Total).
lam0warning	If TRUE indicates how many estimates of lam0 were negative before setting to Inf. Default FALSE.

Details

Assumes rate ratio < 1 is favourable to treatment. A negative estimated event rate indicates no event rate is sufficient under the input assumptions.

Value

The baseline event rate needed to achieve target power at one-sided Type I control level alpha/2, given the input assumptions.

Examples

```
erNeeded(N = 500, bta1 = log(0.6), thta = 2, tau = 1, alp = 0.05, pow = 0.8)
erNeeded(N = 500, bta1 = log(0.6), thta = 3, tau = 1, alp = 0.05, pow = 0.8)


if (require("dplyr") & require("tidyr")) {

  assumptions = tibble(alp = 0.05) %>%
  crossing(
    tau = c(0.8,0.9, 1.0),
    RR = c(0.6, 0.7, 0.8),
    thta = c(2, 3, 4),
    pow = 0.8,
    N = c(500, 1000)
  ) %>%
    mutate(er = erNeeded(N = N, bta1 = log(RR), thta = thta, tau = tau, alp, pow))

  assumptions %>% data.frame()

}
```

---

eventsNeeded *Number of events needed*

Description

Function to compute number of events (L) needed to achieve power (pow) at one-sided Type I control level alpha/2 (alp/2).

Usage

```
eventsNeeded(bta1, thta, tau, lam0, alp = 0.05, pow = 0.8, ar = 0.5)
```

Arguments

bta1	log-transform of rate ratio.
thta	Variance of frailty parameter.
tau	Expected follow-up time.
lam0	Baseline rate for control.
alp	Two-sided alpha-level.
pow	Target power.
ar	Allocation ratio (Number control / Total)

Details

Assumes rate ratio < 1 is favourable to treatment.

Value

The number of events (L) needed.

Examples

```
eventsNeeded(bta1 = log(0.8), thta = 1, tau = 0.8, lam0 = 3.5, alp = 0.05, pow = 0.8)

if (require("dplyr") & require("tidyr")) {

  assumptions = tibble(alp = 0.05) %>%
    crossing(
      tau = c(0.8,0.9, 1.0),
      RR = c(0.6, 0.7, 0.8),
      lam0 = c(3, 3.5),
      thta = c(2, 3, 4),
      pow = 0.8
    ) %>%
    mutate(
      L = eventsNeeded(bta1 = log(RR), thta = thta, tau = tau,
                       lam0 = lam0, alp = alp, pow = pow)
    )

  assumptions %>% data.frame()

}
```

---

eventsNeeded2 *Number events needed (fixed sample size)*

Description

Function to compute number events (L) needed to achieve given power with fixed sample size N.

Usage

```
eventsNeeded2(N, bta1, thta, alp = 0.05, pow = 0.8, ar = 0.5, Lmessage = FALSE)
```

Arguments

N	Sample size.
bta1	log-transform of rate ratio.
thta	Variance of frailty parameter.
alp	Two-sided alpha-level.
pow	Target power.
ar	Allocation ratio (Number control / Total).
Lmessage	If TRUE indicates how many estimates of L were negative before setting to Inf. Default FALSE.

Details

Assumes rate ratio < 1 is favourable to treatment. A negative estimated number of events indicates no number of events is sufficient under the input assumptions.

Value

The number of events (L) needed to achieve target power at one-sided Type I control level alpha/2, given the input assumptions.

Examples

```
eventsNeeded2(N = 500, bta1 = log(0.8), thta = 2, alp = 0.05, pow = 0.8)
eventsNeeded2(N = 500, bta1 = log(0.8), thta = 3, alp = 0.05, pow = 0.8)

if (require("dplyr") & require("tidyr")) {

  assumptions = tibble(alp = 0.05) %>%
  crossing(
    RR = c(0.6, 0.7, 0.8),
    thta = c(2, 3, 4),
    pow = 0.8,
    N = c(500, 1000)
  ) %>%
    mutate(L = eventsNeeded2(N = N, bta1 = log(RR), thta = thta, alp, pow))

  assumptions %>% data.frame()

}
```

---

nNeeded *Function to compute sample size needed to achieve target power*

Description

Computes sample size needed to achieve target power at one-sided Type I control level alp/2.

Usage

```
nNeeded(bta1, thta, tau, lam0, alp = 0.05, pow = 0.8, ar = 0.5)
```

Arguments

bta1	log-transform of rate ratio.
thta	Variance of frailty parameter.
tau	Expected follow-up time.
lam0	Baseline rate for control.
alp	Two-sided alpha-level.
pow	Target power.
ar	Allocation ratio (Number control / Total)

Value

The sample size required given the input assumptions to achieve target power.

Examples

```
nNeeded(bta1 = log(0.8), thta = 1, tau = 0.8, lam0 = 3.5, alp = 0.05, pow = 0.8)

if (require("dplyr") & require("tidyr")) {

  assumptions = tibble(alp = 0.05) %>%
  crossing(
    tau = c(0.8,0.9, 1.0),
    RR = c(0.6, 0.7, 0.8),
    lam0 = c(3, 3.5),
    thta = c(2, 3, 4),
    pow = 0.8
  ) %>%
   mutate(N = nNeeded(bta1 = log(RR), thta = thta, tau = tau, lam0 = lam0, alp = alp, pow = pow))

  assumptions %>% data.frame()

}
```

---

nNeeded2 *Function to compute sample size needed to achieve target power*

Description

Computes sample size needed to achieve target power at one-sided Type I control level alp/2. A negative estimated sample size indicates no sample size is sufficient under the input assumptions.

Usage

```
nNeeded2(bta1, thta, L, alp = 0.05, pow = 0.8, ar = 0.5)
```

Arguments

bta1	log-transform of rate ratio.
thta	Variance of frailty parameter.
L	Number of events
alp	Two-sided alpha-level.
pow	Target power.
ar	Allocation ratio (Number control / Total)

Value

The sample size required given the input assumptions to achieve target power.

Examples

```
nNeeded2(bta1 = log(0.8), thta = 1, L = 1000, alp = 0.05, pow = 0.8)

if (require("dplyr") & require("tidyr")) {

  assumptions = tibble(alp = 0.05) %>%
  crossing(
    L = c(500, 1000, 1500),
    RR = c(0.6, 0.7, 0.8),
    thta = c(2, 3, 4),
    pow = 0.8
  ) %>%
    mutate(N = nNeeded2(bta1 = log(RR), thta = thta, L = L, alp = alp, pow = pow))

  assumptions %>% data.frame()

}
```

---

pow *Power for LWYY (fixed sample size)*

Description

Function to compute power at one-sided Type I control level alp/2.

Usage

```
pow(N, bta1, thta, tau, lam0, alp = 0.05, ar = 0.5)
```

Arguments

N	Sample size.
bta1	log-transform of rate ratio.
thta	Variance of frailty parameter.
tau	Expected follow-up time.
lam0	Baseline rate for control.
alp	Two-sided alpha-level.
ar	Allocation ratio (Number control / Total)

Details

Note: Approximation breaks down in no event scenario. For example, pow(N=1000, bta1 = log(1), thta = 1, lam0 = 0, tau = 1, alp = 0.05) returns a power of 0.025

Value

The power given the input assumptions.

Examples

```
pow(N = 500, bta1 = log(0.8), thta = 2, tau = 1, lam0 = 1.1, alp = 0.05)
pow(N = 500, bta1 = log(0.8), thta = 3, tau = 1, lam0 = 1.1, alp = 0.05)

if (require("dplyr") & require("tidyr")) {

  assumptions = tibble(alp = 0.05) %>%
  crossing(
    tau = c(0.8,0.9, 1.0),
    RR = c(0.6, 0.7, 0.8),
    lam0 = c(3, 3.5),
    thta = c(2, 3, 4),
    N = c(500, 1000)
  ) %>%
   mutate(pow = pow(N = N, bta1 = log(RR), thta = thta, tau = tau, lam0 = lam0, alp = alp))

  assumptions %>% data.frame()

}
```

---

pow2 *Power for LWYY (fixed sample size)*

Description

Function to compute power at one-sided Type I control level alp/2.

Usage

```
pow2(N, bta1, thta, L, alp = 0.05, ar = 0.5)
```

Arguments

N	Sample size.
bta1	log-transform of rate ratio.
thta	Variance of frailty parameter.
L	Number of events.
alp	Two-sided alpha-level.
ar	Allocation ratio (Number control / Total)

Value

The power given the input assumptions.

Examples

```
pow2(N = 500, bta1 = log(0.8), thta = 2, L = 1000, alp = 0.05)
pow2(N = 500, bta1 = log(0.8), thta = 3, L = 1000, alp = 0.05)

if (require("dplyr") & require("tidyr")) {

  assumptions = tibble(alp = 0.05) %>%
  crossing(
    thta = c(1),
    RR = c(0.6, 0.7, 0.8),
    L = c(1000, 1500, 2000),
    N = c(500, 1000)
  ) %>%
    mutate(pow = pow2(N = N, bta1 = log(RR), thta = thta, L = L, alp = alp))

  assumptions %>% data.frame()

}
```

`power.lwyy.test` *Power calculations for one-sided two-sample test of LWYY model parameter.*

Description

Function to compute power at one-sided Type I control level `alp/2` or determine parameters to target given power to test the hypotheses `H0: RR = 1` vs `HA: RR < 1` for the LWYY (Andersen-Gill with robust standard errors) model.

Usage

```
power.lwyy.test(
  N = NULL,
  RR = NULL,
  bta1 = NULL,
  thta,
  L = NULL,
  tau = NULL,
  lam = NULL,
  alp = 0.05,
  pow = NULL,
  ar = 0.5,
  frailty.type = c("unblind", "blind"),
  lam.type = c("base", "pool")
)
```

Arguments

N	Sample size.
RR, bta1	Rate ratio, log-transform of rate ratio. Provide at most one of RR and bta1.
thta	Variance of frailty parameter.
L	Number of events.
tau	Expected follow-up time.
lam	Event rate. If lam.type = "base", it is the event rate for control. If lam.type = "pool", it is the pooled rate.
alp	Two-sided alpha-level.
pow	Target power.
ar	Allocation ratio (Number control / Total)
frailty.type	Indicates whether frailty variance is based on blinded information ("blind") or unblinded ("unblind"). Default "unblind".
lam.type	Indicates whether event rate is based on control rate ("base") or pooled rate ("pool"). Ignored if lam = NULL. Default "base".

Details

An object of class "power.lwyytest" has the following components:

- method: Description of the test.
- N: Total sample size. N = Nc + Nt, where Nc = ar * N is the number on control and Nt = (1-ar) * N is the number on treatment.
- RR, RRCV: Rate ratio powered for and critical value for rate ratio.
- bta1, bta1CV: The log of RR, RRCV.
- thta: The variance of the frailty parameter.
- thtap: If frailty.type = "blind", the input variance from a blinded source.
- L: Number of events. Note that all else being equal, an increase (decrease) in N requires a decrease (increase) in L.
- tau, lam0: The mean follow-up time and control event rate. Note if tau = 1, then lam0 may be interpreted as mean cumulative intensity on control.
- lamp: If lam.type = "pool" and argument lam is not null, the input rate from a blinded source.
- alp: alp/2 is the one-sided Type I control level of the test H0: RR = 1 vs HA: RR < 1.
- pow: Power of the test.
- ar: Allocation to control ratio Nc/N.
- note: Set of key notes about the assessment.

Value

An object of class "power.lwyytest", a list of arguments, including those computed, with method and note elements.

Examples

```
x = power.lwyy.test(N = 1000, RR = 0.8, thta = 1, L = 1000, tau = 0.9, alp = 0.05, ar = 0.5)
print(x)

x = power.lwyy.test(N = 1000, RR = 0.8, thta = 1, tau = 0.9, lam = 1.23, alp = 0.05, ar = 0.5)
print(x)

x = power.lwyy.test(N = 1000, RR = 0.8, thta = 1, tau = NULL, lam = 1.23, alp = 0.05, ar = 0.5)
print(x, digits = 3)
```

`thtap2thta` *Obtain theta from pooled theta (model without treatment).*

Description

Function transforms estimate of blinded pooled theta (from misspecified model without adjusting for treatment) to estimate of theta for model adjusting for treatment.

Usage

```
thtap2thta(bta1, thtap, ar = 0.5, thtawarning = FALSE)
```

Arguments

`bta1`	log-transform of assumed rate ratio.
`thtap`	Estimate of the variance of the frailty parameter under misspecification (no adjustment for treatment).
`ar`	Allocation ratio (Number in control / Total).
`thtawarning`	If TRUE indicates how many estimates of theta were negative before setting to 0. Default FALSE.

Details

This function assumes a recurrent event distribution with exponential baseline function and frailty parameter distribution with mean 1 and variance θ, and derives from expectations of the variance under misspecification.

Specifically, the following relationship between the frailty variance for the misspecified model (θ_p) and correctly specified model (θ) is used

$$\theta_p = Var(Z \cdot \exp\beta) = c^2 - 2pc - 2(1-p)\exp\beta c + (p + (1-p)\exp 2\beta)(\theta + 1)$$

, where $c = E(Z \exp X\beta)$ and $p = ar$.

Value

An estimate of the variance of the frailty parameter when the model includes treatment.

Examples

```
thtap2thta(2, log(c(0.8, 0.7, 0.6)))
```

Index