Package 'CoTiMA'

January 20, 2025

Type Package

Title Continuous Time Meta-Analysis ('CoTiMA')

Version 0.8.0

Date 2024-04-24

Description The 'CoTiMA' package performs meta-
analyses of correlation matrices of repeatedly measured variables taken from
studies that used different time intervals. Different time intervals between measurement occasions impose problems for
meta-analyses because the effects (e.g. cross-
lagged effects) cannot be simply aggregated, for example, by means of common
fixed or random effects analysis. However, continuous time math, which is applied in 'CoTiMA', can be used to extrapolate or
intrapolate the results from all studies to any desired time lag. By this, effects obtained in studies that used different
time intervals can be meta-analyzed. 'CoTiMA' fits models to empirical data using the structural equation model (SEM) package
'ctsem', the effects specified in a SEM are related to parameters that are not directly included in the model (i.e.,
continuous time parameters; together, they represent the continuous time structural equation model, CTSEM). Statistical
model comparisons and significance tests are then performed on the continuous time parameter estimates. 'CoTiMA' also allows
analysis of publication bias (Egger's test, PET-
PEESE estimates, zcurve analysis etc.) and analysis of statistical power
(post hoc power, required sample sizes).
See Dormann, C., Guthier, C., & Cortina, J. M. (2019) <doi:10.1177/1094428119847277>.
and Guthier, C., Dormann, C., & Voelkle, M. C. (2020) <doi:10.1037/bul0000304>.

License GPL-3

URL https://github.com/CoTiMA/CoTiMA

Encoding UTF-8

LazyData true

Depends OpenMx (>= 2.18.1), ctsem (>= 3.8.1), lavaan (>= 0.6), foreach
(>= 1.5.1), R (>= 3.5.0)

Imports MBESS (>= 4.6.0), crayon (>= 1.3.4), psych (>= 1.9.12), doParallel (>= 1.0.15), rootSolve (>= 1.8.2), abind (>= 1.4-5), RPushbullet (>= 0.3.3), openxlsx (>= 4.2.2), zcurve (>= 1.0.7), scholar (>= 0.2.0), stringi (>= 1.0.7), MASS, methods

Suggests R.rsp

VignetteBuilder R.rsp

RoxygenNote 7.3.1

NeedsCompilation no

Author Christian Dormann [aut, cph], Markus Homberg [aut, com, cre], Olga Diener [ctb], Christina Guthier [ctb], Manuel Voelkle [ctb]

Maintainer Markus Homberg <cotima@uni-mainz.de>

Repository CRAN

Date/Publication 2024-04-25 12:40:02 UTC

Contents

A128 *A128 example matrix*

Description

A128 example matrix

Usage

```
A128
```

Format

An object of class `matrix` (inherits from `array`) with 2 rows and 2 columns.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

`A313` *A313 example matrix*

Description

A313 example matrix

Usage

```
A313
```

Format

An object of class `matrix` (inherits from `array`) with 2 rows and 2 columns.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

`ageM1` *ageM1 example vector*

Description

ageM1 example vector

Usage

```
ageM1
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

ageM128 *ageM128 example vector*

Description

ageM128 example vector

Usage

```
ageM128
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

ageM18 *ageM18 example vector*

Description

ageM18 example vector

Usage

```
ageM18
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

ageM201 *ageM201 example vector*

Description

ageM201 example vector

Usage

```
ageM201
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

ageM313 *ageM313 example vector*

Description

ageM313 example vector

Usage

```
ageM313
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

ageM32 *ageM32 example vector*

Description

ageM32 example vector

Usage

```
ageM32
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

ageM4 *ageM4 example vector*

Description

ageM4 example vector

Usage

```
ageM4
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

ageSD1 *ageSD1 example vector*

Description

ageSD1 example vector

Usage

```
ageSD1
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

ageSD128 *ageSD128 example vector*

Description

ageSD128 example vector

Usage

```
ageSD128
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

ageSD18 *ageSD18 example vector*

Description

ageSD18 example vector

Usage

```
ageSD18
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

ageSD201 *ageSD201 example vector*

Description

ageSD201 example vector

Usage

```
ageSD201
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

ageSD313 *ageSD313 example vector*

Description

ageSD313 example vector

Usage

```
ageSD313
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

ageSD32 *ageSD32 example vector*

Description

ageSD32 example vector

Usage

```
ageSD32
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

ageSD4 *ageSD4 example vector*

Description

ageSD4 example vector

Usage

```
ageSD4
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

alphas128 *alphas128 example vector*

Description

alphas128 example vector

Usage

```
alphas128
```

Format

An object of class `numeric` of length 9.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

alphas313 *alphas313 example vector*

Description

alphas313 example vector

Usage

```
alphas313
```

Format

An object of class `numeric` of length 6.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

burnout1 *burnout1 example vector*

Description

burnout1 example vector

Usage

```
burnout1
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

burnout128 *burnout128 example vector*

Description

burnout128 example vector

Usage

```
burnout128
```

Format

An object of class `character` of length 2.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

burnout18 *burnout18 example vector*

Description

burnout18 example vector

Usage

```
burnout18
```

Format

An object of class `character` of length 2.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

burnout201 *burnout201 example vector*

Description

burnout201 example vector

Usage

```
burnout201
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

burnout313 *burnout313 example vector*

Description

burnout313 example vector

Usage

```
burnout313
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

burnout32 *burnout32 example vector*

Description

burnout32 example vector

Usage

```
burnout32
```

Format

An object of class `character` of length 2.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

burnout4 *burnout4 example vector*

Description

burnout4 example vector

Usage

```
burnout4
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

combineVariables128 *combineVariables128 example vector*

Description

combineVariables128 example vector

Usage

```
combineVariables128
```

Format

An object of class `list` of length 3.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

combineVariablesNames128 *combineVariablesNames128 example vector*

Description

combineVariablesNames128 example vector

Usage

```
combineVariablesNames128
```

Format

An object of class `character` of length 3.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

CoTiMABiG_D_BO *ctmaBiG-object reproducing results of Guthier et al. (2020)*

Description

ctmaBiG-object reproducing results of Guthier et al. (2020)

Usage

```
CoTiMABiG_D_BO
```

Format

An object of class `CoTiMAFit` of length 10.

Author(s)

C. Guthier, C. Dormann & J. Cortina <CoTiMA@uni-mainz.org>

CoTiMAFullFit_3 *ctmaFit-object with a 'full' CoTiMA of 3 studies*

Description

ctmaFit-object with a 'full' CoTiMA of 3 studies

Usage

```
CoTiMAFullFit_3
```

Format

An object of class `CoTiMAFit` of length 13.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

CoTiMAFullFit_6 *ctmaFit-object with a 'full' CoTiMA of 6 studies*

Description

ctmaFit-object with a 'full' CoTiMA of 6 studies

Usage

```
CoTiMAFullFit_6
```

Format

An object of class `CoTiMAFit` of length 10.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

CoTiMAFullFit_6_new *ctmaFit-object with a 'full' CoTiMA of 6 studies*

Description

ctmaFit-object with a 'full' CoTiMA of 6 studies

Usage

```
CoTiMAFullFit_6_new
```

Format

An object of class `CoTiMAFit` of length 11.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

CoTiMAFullInv23Fit_6 *1st fitted ctmaFit-object in a series of 2 to test equality of 2 cross effects*

Description

1st fitted ctmaFit-object in a series of 2 to test equality of 2 cross effects

Usage

```
CoTiMAFullInv23Fit_6
```

Format

An object of class `CoTiMAFit` of length 12.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

CoTiMAFullInvEq23Fit_6 *2nd fitted ctmaFit-object in a series of 2 to test equality of 2 cross effects*

Description

2nd fitted ctmaFit-object in a series of 2 to test equality of 2 cross effects

Usage

```
CoTiMAFullInvEq23Fit_6
```

Format

An object of class `CoTiMAFit` of length 11.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

CoTiMAInitFit_3 *ctmaInit-object with of 3 primary studies*

Description

ctmaInit-object with of 3 primary studies

Usage

```
CoTiMAInitFit_3
```

Format

An object of class `CoTiMAFit` of length 17.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

CoTiMAInitFit_6 *ctmaInit-object with 6 primary studies*

Description

ctmaInit-object with 6 primary studies

Usage

```
CoTiMAInitFit_6
```

Format

An object of class `CoTiMAFit` of length 18.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

CoTiMAInitFit_6_new *ctmaInit-object with 6 primary studies*

Description

ctmaInit-object with 6 primary studies

Usage

```
CoTiMAInitFit_6_new
```

Format

An object of class `CoTiMAFit` of length 18.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

CoTiMAInitFit_6_NUTS *ctmaInit-object with a 'full' CoTiMA of 6 studies using NUTS sampler*

Description

ctmaInit-object with a 'full' CoTiMA of 6 studies using NUTS sampler

Usage

```
CoTiMAInitFit_6_NUTS
```

Format

An object of class `CoTiMAFit` of length 17.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

CoTiMAInitFit_D_BO *ctmaInit-object created by Guthier et al. (2020) with 48 primary studies*

Description

ctmaInit-object created by Guthier et al. (2020) with 48 primary studies

Usage

```
CoTiMAInitFit_D_BO
```

Format

An object of class `CoTiMAFit` of length 12.

Author(s)

C. Guthier, C. Dormann & J. Cortina <CoTiMA@uni-mainz.org>

CoTiMAMod1onFullFit_6 *ctmaFit-object with a categorical moderator of the full drift matrix*

Description

ctmaFit-object with a categorical moderator of the full drift matrix

Usage

```
CoTiMAMod1onFullFit_6
```

Format

An object of class `CoTiMAFit` of length 13.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

CoTiMAMod1onFullFit_6_cats12 *ctmaFit-object with a categorical moderator of the full drift matrix*

Description

ctmaFit-object with a categorical moderator of the full drift matrix

Usage

```
CoTiMAMod1onFullFit_6_cats12
```

Format

An object of class `CoTiMAFit` of length 11.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

CoTiMAMod2on23Fit_6 *ctmaFit-object with a continuous moderator of 2 cross effects*

Description

ctmaFit-object with a continuous moderator of 2 cross effects

Usage

```
CoTiMAMod2on23Fit_6
```

Format

An object of class `CoTiMAFit` of length 13.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

`CoTiMAPart134Inv3Fit_6` *ctmaFit-object with with only one cross effect and this one set equal across primary studies*

Description

ctmaFit-object with with only one cross effect and this one set equal across primary studies

Usage

```
CoTiMAPart134Inv3Fit_6
```

Format

An object of class `CoTiMAFit` of length 13.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

`CoTiMAPower_D_BO` *ctmaPower-object reproducing results of Guthier et al. (2020)*

Description

ctmaPower-object reproducing results of Guthier et al. (2020)

Usage

```
CoTiMAPower_D_BO
```

Format

An object of class `CoTiMAFit` of length 10.

Author(s)

C. Guthier, C. Dormann & J. Cortina <CoTiMA@uni-mainz.org>

CoTiMAStanctArgs *This are preset arguments*

Description

This are preset arguments

object created to store standard parameters passed forward to ctStanFit

Usage

```
CoTiMAStanctArgs

CoTiMAStanctArgs
```

Format

An object of class `list` of length 37.

An object of class `list` of length 37.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

CoTiMAstudyList_3 *ctmaPrep-object created with 3 primary studies*

Description

ctmaPrep-object created with 3 primary studies

Usage

```
CoTiMAstudyList_3
```

Format

An object of class `CoTiMAFit` of length 28.

Author(s)

C. Guthier, C. Dormann & J. Cortina <CoTiMA@uni-mainz.org>

CoTiMAstudyList_6 *ctmaPrep-object created with 6 primary studies*

Description

ctmaPrep-object created with 6 primary studies

Usage

```
CoTiMAstudyList_6
```

Format

An object of class `CoTiMAFit` of length 30.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

CoTiMAstudyList_6_new *ctmaPrep-object created with 6 primary studies*

Description

ctmaPrep-object created with 6 primary studies

Usage

```
CoTiMAstudyList_6_new
```

Format

An object of class `CoTiMAFit` of length 30.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

country1 *country1 example vector*

Description

country1 example vector

Usage

```
country1
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

country128 *country128 example vector*

Description

country128 example vector

Usage

```
country128
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

country18 *country18 example vector*

Description

country18 example vector

Usage

```
country18
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

country201 *country201 example vector*

Description

country201 example vector

Usage

```
country201
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

country313 *country313 example vector*

Description

country313 example vector

Usage

```
country313
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

country32 *country32 example vector*

Description

country32 example vector

Usage

```
country32
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

country4 *country4 example vector*

Description

country4 example vector

Usage

```
country4
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

ctmaAllInvFit *ctmaAllInvFit*

Description

Fit a CoTiMA model with all params (drift, T0var, diffusion) invariant across primary studies

Usage

```
ctmaAllInvFit(
  ctmaInitFit = NULL,
  activeDirectory = NULL,
  activateRPB = FALSE,
  digits = 4,
  drift = drift,
  coresToUse = c(1),
  n.manifest = 0,
  indVarying = FALSE,
  scaleTime = NULL,
  optimize = TRUE,
  priors = FALSE,
  finishsamples = NULL,
  iter = NULL,
  chains = NULL,
  verbose = NULL,
  loadAllInvFit = c(),
  saveAllInvFit = c(),
```

```
  silentOverwrite = FALSE,
  customPar = FALSE,
  T0means = 0,
  manifestMeans = 0,
  CoTiMAStanctArgs = NULL,
  lambda = NULL,
  manifestVars = NULL,
  indVaryingT0 = NULL
)
```

Arguments

ctmaInitFit	ctmaInitFit
activeDirectory	activeDirectory
activateRPB	activateRPB
digits	digits
drift	Labels for drift effects. Have to be either of the type V1toV2 or 0 for effects to be excluded, which is usually not recommended)
coresToUse	coresToUse
n.manifest	Number of manifest variables of the model (if left empty it will assumed to be identical with n.latent).
indVarying	Allows ct intercepts to vary at the individual level (random effects model, accounts for unobserved heterogeneity)
scaleTime	scaleTime
optimize	optimize
priors	priors (FALSE)
finishsamples	finishsamples
iter	iter
chains	chains
verbose	verbose
loadAllInvFit	loadAllInvFit
saveAllInvFit	saveAllInvFit
silentOverwrite	silentOverwrite
customPar	logical. If set TRUE (default) leverages the first pass using priors and ensure that the drift diagonal cannot easily go too negative (helps since ctsem > 3.4)
T0means	Default 0 (assuming standardized variables). Can be assigned labels to estimate them freely.
manifestMeans	Default 0 (assuming standardized variables). Can be assigned labels to estimate them freely.
CoTiMAStanctArgs	parameters that can be set to improve model fitting of the ctStanFit Function

lambda	R-type matrix with pattern of fixed (=1) or free (any string) loadings.
manifestVars	define the error variances of the manifests with a single time point using R-type lower triangular matrix with nrow=n.manifest & ncol=n.manifest.
indVaryingT0	Forces T0MEANS (T0 scores) to vary interindividually, which undos the nesting of T0(co-)variances in primary studies (default = TRUE). Was standard until Aug. 2022. Could provide better/worse estimates if set to FALSE.

Value

returns a fitted CoTiMA object, in which all drift parameters, Time 0 variances and covariances, and diffusion parameters were set invariant across primary studies

---

`ctmaBiG` *ctmaBiG*

Description

Analysis of publication bias and generalizability. The function takes a CoTiMA fit object (created with `ctmaInit`) and estimates fixed and random effects of single drift coefficients, heterogeneity (Q, I square, H square, tau square), PET-PEESE corrections, Egger's tests, and z-curve analysis yielding expected replication and detection rates (ERR, EDR).

Usage

```
ctmaBiG(
  ctmaInitFit = NULL,
  activeDirectory = NULL,
  PETPEESEalpha = 0.1,
  activateRPB = FALSE,
  digits = 4,
  zcurve = FALSE,
  undoTimeScaling = TRUE,
  dt = NULL
)
```

Arguments

ctmaInitFit	fit object created with `ctmaInit` containing the fitted ctsem model of each primary study
activeDirectory	the directory where to save results (if not specified, it is taken from ctmaInitFit)
PETPEESEalpha	probability level (condition) below which to switch from PET to PEESE (cf. Stanley, 2017, p. 582, below Eq. 2; default p = .10)
activateRPB	if TRUE, messages (warning, finished) could be send to smart phone (default = FALSE)
digits	rounding (default = 4)

zcurve — performs z-curve analysis. Could fail if too few studies (e.g. around 10) are supplied. default=FALSE

undoTimeScaling — if TRUE, the original time scale is used (timeScale argument possibly used in ctmaInit is undone)

dt — A scalar indicating a time interval across which discrete time effects should be estimated and then used for ctmaBiG.

Value

ctmaBiG returns a list containing some arguments supplied, the results of analyses of publication bias and generalizability, model type, and the type of plot that could be performed with the returned object. The arguments in the returned object are activeDirectory, and coresToUse. Further arguments, which are just copied from the init-fit object supplied, are, n.studies, n.latent, studyList, statisticsList, modelResults (all parameter estimates and their standard error), and parameter names. All new results are returned as the list element "summary", which is printed if the summary function is applied to the returned object. The summary list element comprises a title (model='Analysis of Publication Bias & Generalizability') and "estimates", which is another list comprising "Fixed Effects of Drift Coefficients", "Heterogeneity", "Random Effects of Drift Coefficients", "PET-PEESE corrections", "Egger's tests" (constant of the WLS regression of drift coefficients on their standard errors (SE) with 1/SE^2 as weights), "Egger's tests Alt. Version" (constant of the OLS regression of the standard normal deviates of the drift coefficients on their precision), and "Z-Curve 2.0 Results". Plot type is plot.type=c("funnel", "forest") and model.type="BiG".

Examples

```
## Not run:
# perform analyses of publication bias and generalizability
CoTiMAInitFit_D_BO$activeDirectory <- "/Users/tmp/" # adapt!
CoTiMABiG_D_BO <- ctmaBiG(ctmaInitFit=CoTiMAInitFit_D_BO, zcurve=FALSE)

## End(Not run)

# display results
summary(CoTiMABiG_D_BO)

## Not run:
# get funnel & forest plots
CoTiMABiG_D_BO$activeDirectory <- "/Users/tmp/" # adapt!
plot(CoTiMABiG_D_BO)

## End(Not run)
```

---

ctmaBiGOMX *ctmaBiGOMX*

Description

Analysis of publication bias and fixed and ranom effects analysis of single drift coefficients if OLD OpenMx fit files are supplied

Usage

```
ctmaBiGOMX(
  ctmaInitFit = NULL,
  activeDirectory = NULL,
  PETPEESEalpha = 0.1,
  activateRPB = FALSE,
  digits = 4
)
```

Arguments

ctmaInitFit	fit object created with ctmaInti containing the fitted ctsem model of each primary study
activeDirectory	the directory where to save results (if not specified, it is taken from ctmaInitFit)
PETPEESEalpha	# probability level (condition) below which to switch from PET to PEESE (Stanley, 2017, SPPS,p. 582, below Eq. 2; (default p = .10)
activateRPB	if TRUE, messages (warning, finishs) could be send to smart phone (default = FALSE)
digits	rounding (default = 4)

Value

returns a CoTiMA fit object with results of publication bias analysis, fixed and random effect analysis, Egger's tests, PET-PEESE corrections.

---

ctmaCombPRaw *ctmaCombPRaw*

Description

Combine Pseudo Raw Data (extract them from 'CoTiMAFit object'$studyFitList)

Usage

```
ctmaCombPRaw(listOfStudyFits = NULL, moderatorValues = NULL)
```

Arguments

listOfStudyFits	"Listobject of Studyfits"
moderatorValues	"Moderators

Value

returns a pseudo raw data set that combines pseudo raw data and moderators of primary studies

---

`ctmaCompFit` *ctmaCompFit*

Description

Performs log-liklihood ratio tests to compare the fit of 2 models (CoTiMAFit objects created with `ctmaFit` or `ctmaEqual`), i.e., the difference between the two -2 times LLs between the first model and the more constrained second model. The nested structure of the two models is assumed to be given and not checked.

Usage

```
ctmaCompFit(model1 = NULL, model2 = NULL)
```

Arguments

`model1`	Model 1
`model2`	Model 2

Value

Returns the the difference between the two -2 times LLs (Diff_Minus2LL), the associated difference in degrees of freedom (Diff_df (= Diff_n.params)), and the probability (prob).

Examples

```
minus2llDiffTest <- ctmaCompFit(CoTiMAFullInv23Fit_6,
                                CoTiMAFullInvEq23Fit_6)
summary(minus2llDiffTest)
```

`ctmaCorRel` *ctmaCorRel*

Description

Disattenuates the entries in a correlation matrix using a vector of reliabilities.

Usage

```
ctmaCorRel(empcov = NULL, alphas = NULL)
```

Arguments

empcov	Empirical correlation matrix
alphas	Vector reliabilities

Value

A corrected correlation matrix (corEmpcov). Corrections leading to r > 1.0 are set to 1.0.

Examples

```
empcov313new <- ctmaCorRel(empcov=empcov313, alphas=alphas313)
```

---

ctmaEmpCov *ctmaEmpCov*

Description

changes a full covariance matrix by selecting target variables, recoding them, combining them (compute the mean of two or more variables), and by adding rows/columns with NA if focal variables are not available.

Usage

```
ctmaEmpCov(
  targetVariables = NULL,
  recodeVariables = c(),
  combineVariables = c(),
  combineVariablesNames = c(),
  missingVariables = c(),
  n.latent = NULL,
  Tpoints = NULL,
  sampleSize = NULL,
  pairwiseN = NULL,
  empcov = NULL
)
```

Arguments

targetVariables	(col-/row-) number or names of the target variables
recodeVariables	(col-/row-) number or names of the target variables require inverse coding
combineVariables	list of vectors, which put together the targeted variables that should be used for composite variables

combineVariablesNames
: new names for combined variables - not really important

missingVariables
: missing variables

n.latent
: number of (latent) variables - actually it is the number of all variables

Tpoints
: number of time points.

sampleSize
: sample size

pairwiseN
: matrix of same dimensions as empcov containing possible pairwiseN.

empcov
: empirical correlation matrix

Value

returns a list with two elements. The first element (results$r) contains the adapted correlation matrix, and the second element (results$pairwiseNNew) an adapted version of a matrix of pairwise N if pariwiseN was provided for the original correlation matrix supplied.

Examples

```
source17 <- c()
delta_t17 <- c(12)
sampleSize17 <- 440
empcov17 <- matrix(
  c( 1.00, -0.60, -0.36,  0.20,  0.62, -0.47, -0.18,  0.20,
    -0.60,  1.00,  0.55, -0.38, -0.43,  0.52,  0.27, -0.21,
    -0.36,  0.55,  1.00, -0.47, -0.26,  0.37,  0.51, -0.28,
     0.20, -0.38, -0.47,  1.00,  0.15, -0.28, -0.35,  0.56,
     0.62, -0.43, -0.26,  0.15,  1.00, -0.63, -0.30,  0.27,
    -0.47,  0.52,  0.37, -0.28, -0.63,  1.00,  0.55, -0.37,
    -0.18,  0.27,  0.51, -0.35, -0.30,  0.55,  1.00, -0.51,
     0.20, -0.21, -0.28,  0.56,  0.27, -0.37, -0.51,  1.00),
 nrow=8, ncol=8)
moderator17 <- c(3, 2)
rownames(empcov17) <- colnames(empcov17) <-
  c("Workload_1", "Exhaustion_1", "Cynicism_1", "Values_1",
    "Workload_2", "Exhaustion_2", "Cynicism_2", "Values_2")
targetVariables17 <-
  c("Workload_1", "Exhaustion_1", "Cynicism_1",
    "Workload_2", "Exhaustion_2", "Cynicism_2")
recodeVariables17 <- c("Workload_1", "Workload_2")
combineVariables17 <- list("Workload_1", c("Exhaustion_1", "Cynicism_1"),
                           "Workload_2", c("Exhaustion_2", "Cynicism_2"))
combineVariablesNames17 <- c("Demands_1",  "Burnout_1",
                             "Demands_2",  "Burnout_2")
missingVariables17 <- c();
results17 <- ctmaEmpCov(targetVariables = targetVariables17,
                        recodeVariables = recodeVariables17,
                        combineVariables = combineVariables17,
                        combineVariablesNames = combineVariablesNames17,
                        missingVariables = missingVariables17,
                        n.latent = 2, sampleSize = sampleSize17,
```

```
                        Tpoints = 2, empcov = empcov17)
empcov17 <- results17$r
```

`ctmaEqual` *ctmaEqual*

Description

test if the two or more invariant drift parameters in the CoTiMAFit object supplied are equal. The supplied CoTiMA fit-object (ctmaInvariantFit) has to be a model fitted with `ctmaFit` where at least two parameters were set invariant across primary studies (e.g., 2 cross effects). All parameters that are set invariant in the supplied model are then constrained to be equal by ctmaEqual (no user action required), the model is fitted, and a log-liklihood ratio test is performed informing about the probability that equality applies.

Usage

```
ctmaEqual(
  ctmaInvariantFit = NULL,
  activeDirectory = NULL,
  activateRPB = FALSE,
  digits = 4,
  coresToUse = 2
)
```

Arguments

`ctmaInvariantFit`	object to which a CoTiMA fit has been assigned to (i.e., what has been returned by `ctmaFit`). In most cases probably a model in which (only) two effects were specified with invariantDrift.
`activeDirectory`	defines another active directory than the one used in ctmaInvariantFit
`activateRPB`	set to TRUE to receive push messages with CoTiMA notifications on your phone
`digits`	Number of digits used for rounding (in outputs)
`coresToUse`	If neg., the value is subtracted from available cores, else value = cores to use

Value

returns a model where two or more parameters were set equal across primary studies and a log-likelihood difference test informing about the probability that the equality assumption is correct.

Examples

```
# Fit a CoTiMA with a set of parameters set equal that were set
# invariant in a previous model (of which the fit object is
# supplied in argument ctmaInvariantFit)
## Not run:
CoTiMAFullInv23Fit_6$activeDirectory <- "/Users/tmp/" # adapt!
CoTiMAFullInvEq23Fit_6 <- ctmaEqual(ctmaInvariantFit=CoTiMAFullInv23Fit_6)

## End(Not run)
```

---

ctmaFit *ctmaFit*

Description

Fits a ctsem model with invariant drift effects across primary studies, possible multiple moderators (but all of them of the the same type, either "cont" or "cat"), and possible cluster (e.g., countries where primary studies were conducted).

Usage

```
ctmaFit(
  activateRPB = FALSE,
  activeDirectory = NULL,
  allInvModel = FALSE,
  binaries = NULL,
  catsToCompare = NULL,
  chains = NULL,
  cint = 0,
  cluster = NULL,
  coresToUse = c(2),
  CoTiMAStanctArgs = NULL,
  ctmaInitFit = NULL,
  customPar = FALSE,
  digits = 4,
  drift = NULL,
  driftsToCompare = NULL,
  equalDrift = NULL,
  finishsamples = NULL,
  fit = TRUE,
  ind.mod.names = NULL,
  ind.mod.number = NULL,
  ind.mod.type = "cont",
  indVarying = FALSE,
  indVaryingT0 = NULL,
  inits = NULL,
```

```
  invariantDrift = NULL,
  iter = NULL,
  lambda = NULL,
  manifestMeans = 0,
  manifestVars = 0,
  mod.names = NULL,
  mod.number = NULL,
  mod.type = "cont",
  moderatedDrift = NULL,
  modsToCompare = NULL,
  optimize = TRUE,
  primaryStudyList = NULL,
  priors = FALSE,
  randomIntercepts = FALSE,
  sameInitialTimes = FALSE,
  scaleClus = TRUE,
  scaleMod = TRUE,
  scaleTI = TRUE,
  scaleTime = NULL,
  T0means = 0,
  T0var = "auto",
  transfMod = NULL,
  useSampleFraction = NULL,
  verbose = 0,
  WEC = FALSE
)
```

Arguments

activateRPB	set to TRUE to receive push messages with 'CoTiMA' notifications on your phone
activeDirectory	defines another active directory than the one used in ctmaInitFit
allInvModel	estimates a model with all parameters invariant (DRIFT, DIFFUSION, T0VAR) if set TRUE (defautl = FALSE)
binaries	which manifest is a binary. Still experimental
catsToCompare	when performing contrasts for categorical moderators, the categories (values, not positions) for which effects are set equal
chains	number of chains to sample, during HMC or post-optimization importance sampling.
cint	default 'auto' (= 0). Are set free if random intercepts model with varying cints is requested (by indVarying='cint')
cluster	vector with cluster variables (e.g., countries). Has to be set up carfully. Will be included in ctmaPrep in later 'CoTiMA' versions.
coresToUse	if negative, the value is subtracted from available cores, else value = cores to use
CoTiMAStanctArgs	parameters that can be set to improve model fitting of the ctStanFit Function

ctmaInitFit	object to which all single ctsem fits of primary studies has been assigned to (i.e., what has been returned by ctmaInit)
customPar	logical. If set TRUE leverages the first pass using priors and ensure that the drift diagonal cannot easily go too negative (helps since ctsem > 3.4)
digits	Number of digits used for rounding (in outputs)
drift	labels for drift effects. Have to be either of the type 'V1toV2' or '0' for effects to be excluded.
driftsToCompare	when performing contrasts for categorical moderators, the (subset of) drift effects analyzed
equalDrift	Constrains all listed effects to be equal (e.g., equalDrift = c("V1toV2", "V2toV1")). Note that this is not required for testing the assumption that two effects are equal in the population. Use the invariantDrift argument and then ctmaEqual)
finishsamples	number of samples to draw (either from hessian based covariance or posterior distribution) for final results computation (default = 1000).
fit	TRUE (default) fits the requested model. FALSE returns the ctsem code CoTiMA uses to set up the model, the ctsemmodelbase which can be modified to match users requirements, and the data set (in long format created). The model can then be fitted using ctStanFit)
ind.mod.names	vector of names for individual level (!) moderators used in output
ind.mod.number	which in the vector of individual level (!) moderator values shall be used (e.g., 2 for a single moderator or 1:3 for 3 moderators simultaneously)
ind.mod.type	'cont' or 'cat' of the individual level (!) moderators (mixing them in a single model not yet possible)
indVarying	allows continuous time intercepts to vary at the individual level (random intercepts model, accounts for unobserved heterogeneity)
indVaryingT0	deprecated. Automatically set to NULL.
inits	vector of start values
invariantDrift	drift labels for drift effects that are set invariant across primary studies (default = all drift effects).
iter	number of iterations (defaul = 1000). Sometimes larger values could be required fom Bayesian estimation
lambda	R-type matrix with pattern of fixed (=1) or free (any string) loadings.
manifestMeans	default = 0. Are automatically set free if indVarying is set to TRUE. Can be assigned labels to estimate them freely.
manifestVars	define the error variances (default = 0) of the manifests with a single time point using R-type lower triangular matrix with nrow=n.manifest & ncol=n.manifest.
mod.names	vector of names for moderators used in output
mod.number	which in the vector of moderator values shall be used (e.g., 2 for a single moderator or 1:3 for 3 moderators simultaneously)
mod.type	'cont' or 'cat' (mixing them in a single model not yet possible)
moderatedDrift	labels for drift effects that are moderated (default = all drift effects)

`modsToCompare` when performing contrasts for categorical moderators, the moderator numbers (position in mod.number) that is used

`optimize` if set to FALSE, Stan's Hamiltonian Monte Carlo sampler is used (default = TRUE = maximum a posteriori / importance sampling) .

`primaryStudyList`
could be a list of primary studies compiled with `ctmaPrep` that defines the subset of studies in ctmaInitFit that should actually be used

`priors` if FALSE, any priors are disabled – sometimes desirable for optimization

`randomIntercepts`
(default = FALSE) Experimental. Overrides ctsem's default mode for modelling indVarying cints.

`sameInitialTimes`
Only important for raw data. If TRUE (default=FALSE), T0MEANS occurs for every subject at the same time, rather than just at the earliest observation.

`scaleClus` scale vector of cluster indicators - TRUE (default) yields avg. drift estimates, FALSE yields drift estimates of last cluster

`scaleMod` scale moderator variables - TRUE (default) recommended for continuous and categorical moderators, to separate withing and betwen efeccts

`scaleTI` scale TI predictors - not recommended until version 0.5.3.1. Does not change aggregated results anyways, just interpretation of effects for dummies representing primary studies.

`scaleTime` scale time (interval) - sometimes desirable to improve fitting

`T0means` Default 0 (assuming standardized variables). Can be assigned labels to estimate them freely.

`T0var` (default = 'auto')

`transfMod` more general option to change moderator values. A vector as long as number of moderators analyzed (e.g., c("mean(x)", "x - median(x)"))

`useSampleFraction`
to speed up debugging. Provided as fraction (e.g., 1/10).

`verbose` integer from 0 to 2. Higher values print more information during model fit – for debugging

`WEC` (default = FALSE) Experimental. Uses weighted effect coding of TIpred representing the dummies of the primary studies. Returns drift matrices for all primary studies.

Value

ctmaFit returns a list containing somearguments supplied, the fitted model, different elements summarizing the main results, model type, and the type of plot that could be performed with the returned object. The arguments in the returned object are activeDirectory, coresToUse, moderator names (mod.names), and moderator type (mod.type). Further arguments, which are just copied from the init-fit object supplied, are, n.latent, studyList, parameterNames, and statisticsList. The fitted model is found in studyFitList, which is a large list with many elements (e.g., the ctsem model specified by CoTiMA, the rstan model created by ctsem, the fitted rstan model etc.). Further

results returned are n.studies = 1 (required for proper plotting), data (created pseudo raw data), and a list with modelResults (i.e., DRIFT=model_Drift_Coef, DIFFUSION=model_Diffusion_Coef, T0VAR=model_T0var_Coef, CINT=model_Cint_Coef, MOD=modTI_Coeff, and CLUS=clusTI_Coeff). Possible invariance constraints are included in invariantDrift. The number of moderators simultaneously analyzed are included in ' n.moderators. The most important new results are returned as the list element "summary", which is printed if the summary function is applied to the returned object. The summary list element comprises "estimates" (the aggregated effects), possible randomEffects (not yet fully working), the minus2ll value and its n.parameters, the opt.lag sensu Dormann & Griffin (2015) and the max.effects that occur at the opt.lag, clus.effects and mod.effects, and possible warning messages (message). Plot type is plot.type=c("drift") and model.type="stanct" ("omx" was deprecated).

Examples

```
## Not run:
# Example 1. Fit a CoTiMA to all primary studies previously fitted one by one
# with the fits assigned to CoTiMAInitFit_6
CoTiMAFullFit_6 <- ctmaFit(ctmaInitFit=CoTiMAInitFit_6)
summary(CoTiMAFullFit_6)

## End(Not run)

## Not run:
# Example 2. Fit a CoTiMA with only 2 cross effects invariant (not the auto
# effects) to all primary studies previously fitted one by one with the fits
# assigned to CoTiMAInitFit_6
CoTiMAInitFit_6$activeDirectory <- "/Users/tmp/" # adapt!
CoTiMAFullInv23Fit_6 <- ctmaFit(ctmaInitFit=CoTiMAInitFit_6,
                          invariantDrift=c("V1toV2", "V2toV1"))
summary(CoTiMAFullInv23Fit_6)

## End(Not run)

## Not run:
# Example 3. Fit a moderated CoTiMA
CoTiMAInitFit_6$activeDirectory <- "/Users/tmp/" # adapt!
CoTiMAMod1onFullFit_6 <- ctmaFit(ctmaInitFit=CoTiMAInitFit_6,
                                 mod.number=1, mod.type="cont",
                                 mod.names=c("Control"))
summary(CoTiMAMod1onFullFit_6)

## End(Not run)
```

`ctmaFitList` *ctmaFitList*

Description

Combines CoTiMAFit objects into a list with class CoTiMAFit to inform generic functions what to do

Usage

```
ctmaFitList(...)
```

Arguments

...	any number of CoTiMAFit objects

Value

a list that combines all objects supplied and is assigned the class 'CoTiMAFit'

Examples

```
## Not run:
CoTiMAInitFit_3$activeDirectory <- "/Users/tmp/" # adapt!
CoTiMAFullFit_3$activeDirectory <- "/Users/tmp/" # adapt!
plot(ctmaFitList(CoTiMAInitFit_3, CoTiMAFullFit_3),
     timeUnit="Months",
     timeRange=c(1, 144, 1) )

## End(Not run)
```

---

ctmaFitToPrep *ctmaFitToPrep*

Description

Extracts information from fitted CoTiMA objects to (re-)crearte list of primary studies originally created with ctmaPrep

Usage

```
ctmaFitToPrep(ctmaFitObject = NULL, reUseEmprawData = FALSE)
```

Arguments

ctmaFitObject	ctmaFitObject
reUseEmprawData	whether data should be transferred (will be re-used in subsequent fit attempts)

Value

list that could be used for fitting new CoTiMA models with ctmaInit or ctmaFit.

Examples

```
newStudyList <- ctmaFitToPrep(CoTiMAInitFit_3)
```

ctmaGetPub *ctmaGetPub*

Description

Retrieves publication and citation information from google scholar based on the supplied author names and their google ID (user)

Usage

```
ctmaGetPub(authorList = NULL, flush = FALSE, yearsToExclude = NULL)
```

Arguments

authorList	list of authors and googe scholar addresses
flush	if TRUE, the cache will be cleared and the data reloaded from Google.
yearsToExclude	the years to be excluded (default = current year)

Value

list with (cumulative) frequencies and (cumulative) citations in google scholar

Note

Set flush=TRUE only if retrieving is necessary (e.g., first retrieval on a day)

Examples

```
pubList_8 <- ctmaGetPub(authorList = list( c("J; de Jonge",
              "https://scholar.google.de/citations?hl=de&user=0q27IckAAAAJ"),
              c("Arnold B.; Bakker", "user=FTl3bwUAAAAJ"),
              c("Evangelia; Demerouti", "user=9mj5LvMAAAAJ"),
              c("Joachim; Stoeber", "user=T9xdVusAAAAJ"),
              c("Claude; Fernet", "user=KwzjP4sAAAAJ"),
              c("Frederic; Guay", "user=99vnhX4AAAAJ"),
              c("Caroline; Senecal", "user=64ArFWQAAAAJ"),
              c("Stéphanie; Austin", "user=PPyTI7EAAAAJ")),
              flush=FALSE)
summary(pubList_8)
```

ctmaInit *ctmaInit*

Description

Fits ctsem models to each primary study in the supplied list of primary studies prepared by ctmaPrep.

Usage

```
ctmaInit(
  activateRPB = FALSE,
  activeDirectory = NULL,
  binaries = NULL,
  chains = NULL,
  checkSingleStudyResults = FALSE,
  cint = 0,
  coresToUse = c(2),
  CoTiMAStanctArgs = NULL,
  customPar = FALSE,
  diff = NULL,
  digits = 4,
  doPar = 1,
  drift = NULL,
  experimental = FALSE,
  finishsamples = NULL,
  fit = TRUE,
  indVarying = FALSE,
  indVaryingT0 = NULL,
  iter = NULL,
  lambda = NULL,
  loadSingleStudyModelFit = c(),
  manifestMeans = 0,
  manifestVars = NULL,
  n.latent = NULL,
  n.manifest = 0,
  optimize = TRUE,
  primaryStudies = NULL,
  priors = FALSE,
  randomIntercepts = FALSE,
  sameInitialTimes = FALSE,
  saveRawData = list(),
  saveSingleStudyModelFit = c(),
  scaleTI = NULL,
  scaleTime = NULL,
  silentOverwrite = FALSE,
  T0means = 0,
  T0var = "auto",
```

```
  useSV = FALSE,
  verbose = 0
)
```

Arguments

activateRPB	set to TRUE to receive push messages with 'CoTiMA' notifications on your phone
activeDirectory	defines another active directory than the one used in ctmaPrep
binaries	which manifest is a binary. Still experimental
chains	number of chains to sample, during HMC or post-optimization importance sampling.
checkSingleStudyResults	Displays estimates from single study ctsem models and waits for user input to continue. Useful to check estimates before they are saved.
cint	default 'auto' (= 0). Are set free if random intercepts model with varying cints is requested (by indVarying='cint')
coresToUse	if neg., the value is subtracted from available cores, else value = cores to use
CoTiMAStanctArgs	parameters that can be set to improve model fitting of the ctStanFit Function
customPar	logical. If set TRUE leverages the first pass using priors and ensure that the drift diagonal cannot easily go too negative (helps since ctsem > 3.4)
diff	labels for diffusion effects. Have to be either of the character strings of the type "diff_eta1" or "diff_eta2_eta1" (= freely estimated) or values (e.g., 0 for effects to be excluded, which is usually not recommended)
digits	number of digits used for rounding (in outputs)
doPar	parallel and multiple fitting if single studies. A value > 1 will fit each study doPar times in parallel mode during which no output is generated (screen remains silent). Useful to obtain best fit.
drift	labels for drift effects. Have to be either of the character strings of the type V1toV2 (= freely estimated) or values (e.g., 0 for effects to be excluded, which is usually not recommended)
experimental	used for debugging puposes (default = FALSE)
finishsamples	number of samples to draw (either from hessian based covariance or posterior distribution) for final results computation (default = 1000).
fit	TRUE (default) fits the requested model. FALSE returns the ctsem code CoTiMA uses to set up the model, the ctsemmodelbase which can be modified to match users requirements, and the data set (in long format created). The model can then be fitted using ctStanFit)
indVarying	control for unobserved heterogeneity by having randomly (inter-individually) varying manifest means
indVaryingT0	deprecated. Automatically set to NULL.

`iter` number of interation (defaul = 1000). Sometimes larger values could be required fom Bayesian estimation

`lambda` R-type matrix with pattern of fixed (=1) or free (any string) loadings.

`loadSingleStudyModelFit`
load the fit of single study ctsem models

`manifestMeans` Default 0 (assuming standardized variables). Can be assigned labels to estimate them freely.

`manifestVars` define the error variances of the manifests within a single time point using R-type lower triangular matrix with nrow=n.manifest & ncol=n.manifest.

`n.latent` number of latent variables of the model (hast to be specified)!

`n.manifest` number of manifest variables of the model (if left empty it will assumed to be identical with n.latent).

`optimize` if set to FALSE, Stan's Hamiltonian Monte Carlo sampler is used (default = TRUE = maximum a posteriori / importance sampling) .

`primaryStudies` list of primary study information created with `ctmaPrep`

`priors` if FALSE, any priors are disabled – sometimes desirable for optimization

`randomIntercepts`
(default = FALSE) Experimental. Overrides ctsem's default mode for modelling indVarying cints.

`sameInitialTimes`
Only important for raw data. If TRUE (default=FALSE), T0MEANS occurs for every subject at the same time, rather than just at the earliest observation.

`saveRawData` save (created pseudo) raw date. List: saveRawData$studyNumbers, $fileName, $row.names, col.names, $sep, $dec

`saveSingleStudyModelFit`
save the fit of single study ctsem models (could save a lot of time afterwards if the fit is loaded)

`scaleTI` scale TI predictors

`scaleTime` scale time (interval) - sometimes desirable to improve fitting

`silentOverwrite`
overwrite old files without asking

`T0means` Default 0 (assuming standardized variables). Can be assigned labels to estimate them freely.

`T0var` (default = 'auto')

`useSV` if TRUE (default=FALSE) start values will be used if provided in the list of primary studies

`verbose` integer from 0 to 2. Higher values print more information during model fit - for debugging

Value

ctmaFit returns a list containing some arguments supplied, the fitted models, different elements summarizing the main results, model type, and the type of plot that could be performed with the returned object. The arguments in the returned object are activeDirectory, coresToUse, n.latent, n.manifest, and primaryStudyList. The study count is returned as n.studies, the created matrix of loadings of manifest on latent factors is returned as lambda, and a re-organized list of primary studies with some information ommited is returned as studyList. The fitted models for each primary study are found in studyFitList, which is a large list with many elements (e.g., the ctsem model specified by CoTiMA, the rstan model created by ctsem, the fitted rstan model etc.). Further results returned are emprawList (containing the pseudo raw data created), statisticsList (comprising baisc stats such as average sample size, no. of measurement points, etc.), a list with modelResults (i.e., DRIFT=model_Drift_Coef, DIFFUSION=model_Diffusion_Coef, T0VAR=model_T0var_Coef, CINT=model_Cint_Coef), and the paramter names internally used. The summary list, which is printed if the summary function is applied to the returned object, comprises "estimates" (the aggregated effects), possible randomIntercepts,confidenceIntervals, the minus2ll value and its n.parameters, and possible warning messages (message). Plot type is plot.type=c("drift") and model.type="stanct" ("omx" was deprecated).

Examples

```
# Fit a ctsem model to all three primary studies summarized in
# CoTiMAstudyList_3 and save the three fitted models
## Not run:
CoTiMAInitFit_3 <- ctmaInit(primaryStudies=CoTiMAstudyList_3,
                            n.latent=2,
                            checkSingleStudyResults=FALSE,
                            activeDirectory="/Users/tmp/") # adapt!
summary(CoTiMAInitFit_3)

## End(Not run)
```

`ctmaLabels` *ctmaLabels*

Description

used for consistent labeling of names and parameters

Usage

```
ctmaLabels(
  n.latent = NULL,
  n.manifest = 0,
  lambda = NULL,
  manifestVars = NULL,
  drift = NULL,
  diff = NULL,
```

```
  invariantDrift = NULL,
  moderatedDrift = NULL,
  equalDrift = NULL,
  T0means = 0,
  manifestMeans = 0
)
```

Arguments

n.latent	n.latent
n.manifest	n.manifest
lambda	lambda
manifestVars	manifestVar
drift	drift
diff	diffusion
invariantDrift	invariantDrift
moderatedDrift	moderatedDrift
equalDrift	equalDrift
T0means	T0means
manifestMeans	manifestMeans

Value

returns consistently named parameters (e.g., "V1toV2") as well es their symbolic values, which are used to fix or free parameters when fitting a 'CoTiMA' model

---

`ctmaLCS` *ctmaLCS*

Description

Transforms estimates obtained with `ctmaFit` into LCS (latent change score) terminology. LCS models can be estimated with CT CLPM, but results have to be transformed. When time intervals vary much between and within persons, LCS models are virtually impossible to fit. However, CT CLPM models can be fitted, and the results - after transformation - show what LCS estimates would have been (cf Voelke & Oud, 2015; their terminology to label LCS effects is used in the output created by ctmaLCS)

Usage

```
ctmaLCS(
  CoTiMAFit = NULL,
  undoTimeScaling = TRUE,
  digits = 4,
  activateRPB = FALSE
)
```

Arguments

CoTiMAFit	Fitted CoTiMA object.
undoTimeScaling	Whether (TRUE) or not (FALSE) LCS results should be provided ignoring the scaleTime argument used in ctmaFit.
digits	Number of digits used for rounding (in outputs)
activateRPB	set to TRUE to receive push messages with 'CoTiMA' notifications on your phone

Value

Returns LCS effects derived from CT CoTiMA CLPM estimates.

Examples

```
## Not run:
LCSresults <- ctmaLCS(CoTiMAFullFit_6)

## End(Not run)
```

---

ctmaOptimizeFit *ctmaOptimizeFit*

Description

Replaces deprecated `ctmaOptimizeInit`, which was limited to initial fitting (i.e., applies `ctmaInit`) of a primary study reFits times to capitalize on chance for obtaining a hard-to-find optimal fit. Now, optimizing a CoTiMA model generated with `ctmaFit` can also be done. Using `ctmaOptimizeFit` could be helpful if a model yields out-of-range estimates, which could happen if the fitting algorithm unfortunately used random start values that resulted in a locally but not globally optimal fit. Essentially, using `ctmaOptimizeFit` is like gambling, hoping that at least one set of starting values (the number it tries is specified in the reFits argument) enables finding the global optimal fit.

Usage

```
ctmaOptimizeFit(
  activateRPB = FALSE,
  activeDirectory = NULL,
  coresToUse = c(2),
  CoTiMAStanctArgs = NULL,
  ctmaFitFit = NULL,
  ctmaInitFit = NULL,
  customPar = FALSE,
  finishsamples = NULL,
  iter = 5000,
```

```
  primaryStudies = NULL,
  problemStudy = NULL,
  randomPar = FALSE,
  randomScaleTI = FALSE,
  randomScaleTime = c(1, 1),
  saveModelFits = FALSE,
  shuffleStudyList = FALSE,
  reFits = NULL,
  scaleTime = NULL,
  scaleTI = NULL,
  verbose = 1
)
```

Arguments

activateRPB	set to TRUE to receive push messages with 'CoTiMA' notifications on your phone
activeDirectory	activeDirectory
coresToUse	if neg., the value is subtracted from available cores, else value = cores to use
CoTiMAStanctArgs	parameters that can be set to improve model fitting of the ctStanFit Function
ctmaFitFit	a object fitted with ctmaFit
ctmaInitFit	the ctmaInitFit object that was used to create the ctmaFitFit object with ctmaFit
customPar	logical. If set TRUE leverages the first pass using priors and ensure that the drift diagonal cannot easily go too negative (helps since ctsem > 3.4)
finishsamples	number of samples to draw (either from hessian based covariance or posterior distribution) for final results computation (default = 1000).
iter	number of iterations (default = 5000)
primaryStudies	list of primary study information created with ctmaPrep or ctmaFitToPrep
problemStudy	number (position in list) where the problem study in primaryStudies is found
randomPar	logical (default = FALSE). Overrides arguments used for customPar and randomly sets customPar either TRUE or FALSE
randomScaleTI	logical (default = FALSE). Overrides arguments used for scaleTI and randomly sets scaleTI either TRUE or FALSE
randomScaleTime	lower and upper limit (default = c(1,1)) of uniform distribution from which timeScale argument for ctmaInit is uniformly shuffled (integer)
saveModelFits	save the fit of each Fit attempt (default = FALSE).
shuffleStudyList	(default = FALSE) randomly re-arranges studies in primaryStudyList. We encountered a few cases where this mattered, even though it should not. Only works if ctmaFit is optimized.
reFits	how many reFits should be done

scaleTime	scale time (interval) - sometimes desirable to improve fitting
scaleTI	scale TI predictors - not recommended until version 0.5.3.1. Does not change aggregated results anyways, just interpretation of effects for dummies representing primary studies.
verbose	integer from 0 to 2. Higher values print more information during model fit – for debugging

Value

returns a list with bestFit (= the best fit achieved), all_minus2ll (= all -2ll values for all fitted models), and summary, which is printed if the summary function is applied to the returned object, and which shows the summary information of the ctsem model with the best fit.

Examples

```
## Not run:
optimFit313 <- ctmaOptimizeFit(primaryStudies=CoTiMAstudyList_3,
                               activeDirectory="/Users/tmp/",  # adapt!
                              problemStudy=which(CoTiMAstudyList_3$studyNumbers == 313),
                               reFits=10,
                               n.latent=2)
summary(optimFit313)

## End(Not run)
```

`ctmaOptimizeInit` *ctmaOptimizeInit*

Description

Initial fitting (i.e., applies `ctmaInit`) to a primary study reFit times to capitalize on chance for obtaining a hard-to-find optimal fit. This could be very helpful if a primary yields out-of-range estimates, which could happen if the fitting algorithm unfortunately used random start values that resulted in a locally but not globally optimal fit. Essentially, using ctmaOptimizeInit is like gambling, hoping that at leas one set of starting values (the number is tries is specified in the reFits argument) eneables finding the global optimal fit. On unix-like machines (e.g. MacOS), this could be done in parallel mode if coresToUse > 1.

Usage

```
ctmaOptimizeInit(
  primaryStudies = NULL,
  activeDirectory = NULL,
  problemStudy = NULL,
  reFits = NULL,
  finishsamples = NULL,
  n.latent = NULL,
```

```
  coresToUse = c(1),
  indVarying = FALSE,
  randomScaleTime = c(1, 1),
  activateRPB = FALSE,
  checkSingleStudyResults = FALSE,
  customPar = FALSE,
  T0means = 0,
  manifestMeans = 0,
  manifestVars = NULL,
  CoTiMAStanctArgs = NULL,
  scaleTime = NULL
)
```

Arguments

primaryStudies	list of primary study information created with ctmaPrep or ctmaFitToPrep
activeDirectory	activeDirectory
problemStudy	number (position in list) where the problem study in primaryStudies is found
reFits	how many reFits should be done
finishsamples	number of samples to draw (either from hessian based covariance or posterior distribution) for final results computation (default = 1000).
n.latent	number of latent variables of the model (hast to be specified)!
coresToUse	if neg., the value is subtracted from available cores, else value = cores to use
indVarying	control for unobserved heterogeneity by having randomly (inter-individually) varying manifest means
randomScaleTime	lower and upper limit of uniform distribution from which timeScale argument for ctmaInit is uniformly shuffled (integer)
activateRPB	set to TRUE to receive push messages with 'CoTiMA' notifications on your phone
checkSingleStudyResults	displays estimates from single study 'ctsem' models and waits for user input to continue. Useful to check estimates before they are saved.
customPar	logical. If set TRUE (default) leverages the first pass using priors and ensure that the drift diagonal cannot easily go too negative (helps since ctsem > 3.4)
T0means	Default 0 (assuming standardized variables). Can be assigned labels to estimate them freely.
manifestMeans	Default 0 (assuming standardized variables). Can be assigned labels to estimate them freely.
manifestVars	define the error variances of the manifests with a single time point using R-type lower triangular matrix with nrow=n.manifest & ncol=n.manifest.
CoTiMAStanctArgs	parameters that can be set to improve model fitting of the ctStanFit Function
scaleTime	scale time (interval) - sometimes desirable to improve fitting

Value

returns a list with bestFit (= the best fit achieved), all_minus2ll (= all -2ll values for all fitted models), and summary, which is printed if the summary function is applied to the returned object, and which shows the summary information of the ctsem model with the best fit.

Note

All but one of multiple cores are used on unix-type machines for parallel fitting

During fitting, not output is generated. Be patient.

Examples

```
## Not run:
optimFit313 <- ctmaOptimizeInit(primaryStudies=CoTiMAstudyList_3,
                                  activeDirectory="/Users/tmp/",  # adapt!
                                problemStudy=which(CoTiMAstudyList_3$studyNumbers == 313),
                                  reFits=10,
                                  n.latent=2)
summary(optimFit313)

## End(Not run)
```

---

ctmaPlot *ctmaPlot*

Description

Forest plot, funnel plots, plots of discrete time cross-lagged and autoregressive effect, and plots of required sample sizes

Usage

```
ctmaPlot(
  ctmaFitObject = NULL,
  activeDirectory = NULL,
  saveFilePrefix = "ctmaPlot",
  activateRPB = FALSE,
  plotCrossEffects = TRUE,
  plotAutoEffects = TRUE,
  timeUnit = "timeUnit (not specified)",
  timeRange = c(),
  yLimitsForEffects = c(),
  mod.number = 1,
  mod.values = -2:2,
  aggregateLabel = "",
  xLabels = NULL,
```

```
  undoTimeScaling = TRUE,
  ...
)
```

Arguments

ctmaFitObject	'CoTiMA' Fit object
activeDirectory	defines another active directory than the one used in ctmaInitFit
saveFilePrefix	Prefix used for saved plots
activateRPB	set to TRUE to receive push messages with 'CoTiMA' notifications on your phone
plotCrossEffects	logical
plotAutoEffects	logical
timeUnit	label for x-axis when plotting discrete time plots
timeRange	vector describing the time range for x-axis as sequence from/to/stepSize (e.g., c(1, 144, 1))
yLimitsForEffects	range for y-axis
mod.number	moderator number that should be used for plots
mod.values	moderator values that should be used for plots
aggregateLabel	label to indicate aggregated discrete time effects
xLabels	labes used for x-axis
undoTimeScaling	if TRUE, the original time scale is used (timeScale argument possibly used in ctmaInit is undone)
...	arguments passed through to plot()

Value

depending on the CoTiMA fit object supplied, generates funnel plots, forest plots, discrete time plots of autoregressive and cross-lagged effects, plots of required samples sizes across a range of discrete time intervals to achieve desired levels of statistical power, and post hoc power of primary studies. Plots are saved to disk.

Examples

```
## Not run:
# cannot run without proper activeDirectory specified. Adapt!
CoTiMAFullFit_3$activeDirectory <- "/Users/tmp/" # adapt!
plot(ctmaFitList(CoTiMAInitFit_3, CoTiMAFullFit_3),
     timeUnit="Months", timeRange=c(1, 144, 1),
     plotAutoEffects=FALSE)
```

```
## End(Not run)

## Not run:
# cannot run without proper activeDirectory specified. Adapt!
CoTiMABiG_D_BO$activeDirectory <- "/Users/tmp/" # adapt!
plot(CoTiMABiG_D_BO)

## End(Not run)
```

---

ctmaPlotCtsemMod *ctmaPlotCtsemMod*

Description

Plots moderator models using ctsem fit objects

Usage

```
ctmaPlotCtsemMod(
  ctStanFitObject = NULL,
  fitSummary = NULL,
  activeDirectory = NULL,
  TIpred.pos = 1,
  saveFilePrefix = "Moderator Plot ",
  scaleTime = 1,
  mod.sd.to.plot = -1:1,
  digits = 4,
  timeUnit = "not specified",
  timeRange = NULL,
  mod.type = "cont",
  no.mod.cats = NULL,
  n.x.labels = NULL,
  plot = TRUE,
  plot.xMin = 0,
  plot.xMax = NULL,
  plot.yMin = -1,
  plot.yMax = 1,
  plot..type = "l",
  plot.lty = 1,
  plot.col = "grey",
  plot.lwd = 1.5,
  dot.plot.type = "b",
  dot.plot.col = "black",
  dot.plot.lwd = 0.5,
  dot.plot.lty = 3,
  dot.plot.pch = 16,
  dot.plot.cex = 3
)
```

Arguments

ctStanFitObject	The fit object with moderator (TIpred) effects to be plotted
fitSummary	Mainl ofr debugging purpose. Saves computation time if provided in addition to the fit object
activeDirectory	defines the active directory (where to save plots)
TIpred.pos	the Tipred that represents the moderator. Could be more than one in case of dummy variables made from categorical moderators (e.g., TIpred.pos = c(3,4))
saveFilePrefix	Prefix used for saving plots
scaleTime	Factor to increase or decrease the time scale (e.g., 1/12 if estimates were based on yearly intervals and figure should show monthly intervals)
mod.sd.to.plot	The standard deviation vlaues (default -1, 0, +1) for which the drift effects are plotted
digits	number of digits used for rounding
timeUnit	Label for the x-axis
timeRange	time range across which drift effects are plotted
mod.type	Could be either "cont" or "cat"
no.mod.cats	Need to be specified if type = "cat". The number of categories should usually be equal the number of dummy variables used to represent the categorical moderator + 1.
n.x.labels	How many values to be used for indicating time points on the x-axis (0 is automatically added and should not be counted)
plot	plots figures if TRUE (default) otherwise only return moderated drift matrices
plot.xMin	default = 0
plot.xMax	default = NULL
plot.yMin	default = -1
plot.yMax	default = 1
plot..type	default = "l", # 2 dots .. are correct
plot.lty	default = 1
plot.col	default = "grey"
plot.lwd	default = 1.5
dot.plot.type	default = "b" for the dots indicating the moderator values
dot.plot.col	default ="black" for the dots indicating the moderator values
dot.plot.lwd	default = .5 for the dots indicating the moderator values
dot.plot.lty	default = 3 for the dots indicating the moderator values
dot.plot.pch	default = 16 for the dots indicating the moderator values
dot.plot.cex	default = 3 for the dots indicating the moderator values

Value

writes png figures to disc using the path specified in the activeDirectory arguments.

Examples

```
#Plot a categorical moderator
## Not run:
ctmaPlotCtsemMod(ctStanFitObject = ctsemFit,
                 activeDirectory=NULL,
                 mod.sd.to.plot = NULL,
                 timeUnit = "Months",
                 timeRange = c(0, 12, .1),
                 mod.type = "cat",
                 no.mod.cats = NULL

## End(Not run)
```

`ctmaPower` *ctmaPower*

Description

Fits a full invariant model to a list of primary studies and performs analyses of expected (post hoc) power and required sample sizes.

Usage

```
ctmaPower(
  ctmaInitFit = NULL,
  activeDirectory = NULL,
  statisticalPower = c(),
  failSafeN = NULL,
  failSafeP = NULL,
  timeRange = NULL,
  useMBESS = FALSE,
  coresToUse = 1,
  digits = 4,
  indVarying = FALSE,
  activateRPB = FALSE,
  silentOverwrite = FALSE,
  loadAllInvFit = c(),
  saveAllInvFit = c(),
  loadAllInvWOSingFit = c(),
  saveAllInvWOSingFit = c(),
  skipScaling = TRUE,
  useSampleFraction = NULL,
  optimize = TRUE,
```

```
  priors = FALSE,
  finishsamples = NULL,
  iter = NULL,
  chains = NULL,
  verbose = NULL,
  customPar = FALSE,
  scaleTime = NULL
)
```

Arguments

ctmaInitFit	object to which all single 'ctsem' fits of primary studies has been assigned to (i.e., what has been returned by ctmaInit)
activeDirectory	defines another active directory than the one used in ctmaInit
statisticalPower	vector of requested statistical power values
failSafeN	sample size used to determine across which time intervals effects become non-significant
failSafeP	p-value used to determine across which time intervals effects become non-significant
timeRange	vector describing the time range for x-axis as sequence from/to/stepSize (e.g., c(1, 144, 1))
useMBESS	use 'MBESS' package to calculate statistical power (slower)
coresToUse	if negative, the value is subtracted from available cores, else value = cores to use
digits	number of digits used for rounding (in outputs)
indVarying	Allows continuous time intercepts to vary at the individual level (random effects model, accounts for unobserved heterogeneity)
activateRPB	set to TRUE to receive push messages with 'CoTiMA' notifications on your phone
silentOverwrite	overwrite old files without asking
loadAllInvFit	load the fit of fully constrained 'CoTiMA' model
saveAllInvFit	save the fit of fully constrained 'CoTiMA' model
loadAllInvWOSingFit	load series of fits of fully constrained 'CoTiMA' model with single cross effects excluded, respectively
saveAllInvWOSingFit	save series of fits of fully constrained 'CoTiMA' model with single cross effects excluded, respectively
skipScaling	does not (re-)scale raw data (re-scaling of imported pseudo raw data achieves correlations = 1)
useSampleFraction	to speed up debugging. Provided as fraction (e.g., 1/10)

optimize	if set to FALSE, Stan's Hamiltonian Monte Carlo sampler is used (default = TRUE = maximum a posteriori / importance sampling) .
priors	if FALSE, any priors are disabled – sometimes desirable for optimization
finishsamples	number of samples to draw (either from hessian based covariance or posterior distribution) for final results computation (default = 1000).
iter	number of iterations (defaul = 1000). Sometimes larger values could be required fom Bayesian estimation
chains	number of chains to sample, during HMC or post-optimization importance sampling.
verbose	integer from 0 to 2. Higher values print more information during model fit – for debugging
customPar	logical. If set TRUE (default) leverages the first pass using priors and ensure that the drift diagonal cannot easily go too negative (helps since ctsem > 3.4)
scaleTime	scale time (interval) - sometimes desirable to improve fitting

Value

ctmaPower returns a list containing some arguments supplied, a fitted model with all (!) parameters invariant across primary studies, different elements summarizing the main results, model type, and the type of plot that could be performed with the returned object. The arguments in the returned object are activeDirectory, coresToUse, n.latent, n.manifest, and primaryStudyList. A further result returned is n.studies = 1 (required for proper plotting). Further arguments, which are just copied from the init-fit object supplied, are, n.latent, studyList, and the statisticsList. The fitted model is found in studyFitList, which is a large list with many elements (e.g., the ctsem model specified by CoTiMA, the rstan model created by ctsem, the fitted rstan model etc.). Further results returned are a list with modelResults (i.e., DRIFT=DRIFT, DIFFUSION=DIFFUSION, T0VAR=T0VAR, CINT=NULL) and the paramter names internally used. The summary list, which is printed if the summary function is applied to the returned object, contains "estimates", which is itself a list comprising "Estimates of Model with all Effects Invariant", "Requested Statistical Power" (which just returns the argument statisticalPower), "Power (post hoc) for Drift Effects", "Required Sample Sizes" "Effect Sizes (based on discrete-time calcs; used for power calcs.)", and "Range of significant effects" (across which intervals effects were significant). Plot type is plot.type=c("power") and model.type="stanct" ("omx" was deprecated).

Examples

```
## Not run:
CoTiMAInitFit_D_BO$activeDirectory <- "/Users/tmp/" # adapt!
CoTiMAPower_D_BO <- ctmaPower(ctmaInitFit=CoTiMAInitFit_D_BO,
                              statisticalPower = c(.50, .80, .95),
                              finishsamples = 10000)
summary(CoTiMAPower_D_BO)

## End(Not run)
```

ctmaPRaw *ctmaPRaw*

Description

Converts empirical correlation matrices to pseudo raw data (i.e. random data, that perfectly reproduce the correlations)

Usage

```
ctmaPRaw(
  empCovMat = NULL,
  empNMat = matrix(0, 0, 0),
  empN = NULL,
  studyNumber = NULL,
  empMeanVector = NULL,
  empVarVector = NULL,
  activateRPB = FALSE,
  experimental = FALSE
)
```

Arguments

empCovMat	empirical primary study covariance matrix
empNMat	matrix of (possibly pairwise) N
empN	N (in case of listwise N)
studyNumber	internal number
empMeanVector	vector of means for all variables, usually 0
empVarVector	vector of variances for all variables, usually 1
activateRPB	set TRUE to receive push messages with 'CoTiMA' notifications on your phone
experimental	set TRUE to try new pairwise N function

ctmaPrep *ctmaPrep*

Description

Combines information of primary studies into a list object and returns this list. This list is then used as input to fit 'ctsem' models. Primary study information is expected to be assigned to 'numbered' objects. Some of these objects are pre-defined (e.g., 'empcov', 'ageM'). Most of the pre-defined objects could be empty, or they could be dropped by entering their names in the excludedElements-object (e.g., excludedElements = c('ageM')), but dropping them is not really necessary. Additional elements could also be added, which could be useful to put together all information about primary studies at the convenience of the researcher.

Usage

```
ctmaPrep(
  selectedStudies = NULL,
  excludedElements = NULL,
  addElements = NULL,
  digits = 4,
  moderatorLabels = NULL,
  moderatorValues = NULL,
  newRawDataDirectory = NULL,
  summary = TRUE,
  activeDirectory = NULL,
  ctmaPrepObject = NULL,
  excludedStudies = NULL
)
```

Arguments

`selectedStudies`
: Vector of primary study numbers (numeric values with no leading 0; e.g., '2' but not '02')

`excludedElements`
: Vector of predefined objects used to code primary study information. Some predefined objects are strongly defined; they have to be used in a special way because they are actually used in subsequent analyses. Some other objects could be used at the researcher's convenience (information is just collected). Strongly predefined objects are 'delta_t' (vector of time intervals; the only mandatory requirement; should be of the type c(NA, NA) in cases when raw data are provided), 'sampleSize' (single number), 'pairwiseN' (matrix of pairwise N; could be used if correlation matrix is based on pairwise N), 'empcov' (correlation matrix), 'moderator' (vector of numbers; could be continuous or categorical), 'startValues' (vector of start values), 'rawData' (information about file name and structure of raw data), 'empMeans' (means for variables; usually 0), and 'empVars' (varainces for variables; usually 1). Weakly predefined objects are 'studyNumber' (intended as a special number used for the outputs of subsequently fitted CoTiMA models), 'source' (intended as vector of authors' names and publication year), 'ageM' (intended as value indicating the mean age of participants in a primary study), 'malePercent' (intended as value indicating the percentage of male participants in a primary study), 'occupation' (intended as vector of character strings representing the occupations of participants in a primary study), 'country' (intended as single character string representing the country in which a primary study was conducted), 'alphas' (intended as vector of Cronbach's alphas of the variables of a primary study; not yet functional), and 'targetVariables' (intended as vector of character strings representing information about the variables used).'

`addElements`
: User-added objects that are handled as the weakly predefined objects. The major purpose is to collect information a researcher regards as important.

`digits`
: Rounding used for summary function

`moderatorLabels`
: character vector of names

`moderatorValues`
: list of character vectors

`newRawDataDirectory`
: (NULL = default) Change paths for all raw data files.

`summary`
: if TRUE (default) creates summary table and xlsx sheets. Could be set to FALSE in case of errors.

`activeDirectory`
: Mandatory. If subsequent fitting is done using different folders or on different computers, it can be

`ctmaPrepObject`
: previously created object with ctmaPrep, from which studies should be excluded. Only works in combination with the argument excludeStudies.

`excludedStudies`
: studies to be excluded from a previously created ctmaPrep-object changed so that raw data files can be loaded.

Value

List of primary studies and parameters for the following CoTiMA (plus StudyInformation which could be saved to Excel)

Note

The following example shows information a researcher has about three studies, which have the numbers '2', '4' and '17'. All information about these studies are stored in objects ending with '2', '4', and '17', respectively. In most instances, one relevant piece of information is the empirical correlation (or covariance) matrix reported in this study, which is stored in the objects 'empcov2', 'empcov4', and 'empcov17'. Note that full and symmetric matrices are required for ctmaPrep. Usually, sample sizes ('sampleSize2', 'sampleSize4', & 'sampleSize17') and time lags ('delta_t2', 'delta_t4', & 'delta_t17'), are required, too.

Examples

```
# First Study
empcov2 <- matrix(c(1.00, 0.45, 0.57, 0.18,
                    0.45, 1.00, 0.31, 0.66,
                    0.57, 0.31, 1.00, 0.40,
                    0.18, 0.66, 0.40, 1.00), nrow=4, ncol=4)
delta_t2 <- 12
sampleSize2 <- 148
moderator2 <- c(1, 0.72)
source2 <- c("Houkes, I,", "Janssen, P, P, M,", "de Jonge, J",
             "& Bakker, A, B", "Study1", "2003")
addedByResearcher2 <- "something you want to add"

# Second Study
empcov3 <- matrix(c(1.00, 0.43, 0.71, 0.37,
                    0.43, 1.00, 0.34, 0.69,
                    0.71, 0.34, 1.00, 0.50,
```

```
                    0.37, 0.69, 0.50, 1.00), nrow=4, ncol=4)
delta_t3 <- 12
sampleSize3 <- 88
moderator3 <- c(1, 0.72)
source3 <- c("Houkes, I,", "Janssen, P, P, M,", "de Jonge, J",
             "& Bakker, A, B", "Study2", "2003")
addedByResearcher3 <- ""

# Third Study
empcov313 <- matrix(c(1.00, 0.38, 0.54, 0.34, 0.60, 0.28,
                      0.38, 1.00, 0.34, 0.68, 0.28, 0.68,
                      0.54, 0.34, 1.00, 0.47, 0.66, 0.39,
                      0.34, 0.68, 0.47, 1.00, 0.38, 0.72,
                      0.60, 0.28, 0.66, 0.38, 1.00, 0.38,
                      0.28, 0.68, 0.39, 0.72, 0.38, 1.00), nrow=6, ncol=6)
delta_t313 <- c(1.5, 1.5)
sampleSize313 <- 335
moderator313 <- c(0.8,2.47)
source313 <- c("Demerouti", "Bakker", "& Bulters", "2004")
addedByResearcher313 <- "check correlation matrix"

# Add Labels and Values for Moderators (just for optional excel tables)
moderatorLabels <- c("Control", "Social Support")
moderatorValues <- list("continuous", c("1 = very low", "2 = low",
                        "3 = medium", "4 = high", "5 = very high"))

CoTiMAstudyList_3 <- ctmaPrep(selectedStudies = c(2, 3, 313),
                              activeDirectory="/user/",
                              excludedElements = "ageM",
                              addElements = "addedByResearcher",
                              moderatorLabels=moderatorLabels,
                              moderatorValues=moderatorValues)
```

---

ctmaPub *ctmaPub*

Description

Compute publication and citation scores for studies based on the (team of) authors' publication scores .

Usage

```
ctmaPub(
  getPubObj = NULL,
  primaryStudyList = NULL,
  yearsToExclude = 0,
  recency = 5,
  targetYear = NULL,
```

```
  indFUN = "sum",
  colFUN = "mean",
  addAsMod = FALSE
)
```

Arguments

getPubObj	publication information compiled with ctmaGetPub
primaryStudyList	vector with numbers of studies (e.g., c(1,3); requires source1 and source3 to be available)
yearsToExclude	years to exclude from publications
recency	years before targetYear that are considered for recency analysis
targetYear	year (default = last year) after which publications are ignored
indFUN	function (default = sum) how publications of each author within a collective (team) are summarized
colFUN	function (default = mean) how publications all authors of collective (team) are summarized
addAsMod	currently disabled. Add to existing moderator objects (or create them) in primaryStudyList, which is part of the returned object

Value

returns NEPP (= the *number* of studies published by the authors of the primary studies supplied UNTIL the year when the primary study was published), NEPPRecency (like NEPP, but limited to the number of years before the publication as specified with the recency argument), "Meaning of NEPP" and "Meaning of NEPPRecency" which explain what *number* exactly means (e.g., could be the mean of the sum of each author's publication, or the sum of the maximum publications per year of the authors), and "primaryStudyList(full)", which just returns the primaryStudyList supplied).

Examples

```
pubResults_6 <- ctmaPub(getPubObj=pubList_8,
                        primaryStudyList=CoTiMAstudyList_6)
summary(pubResults_6)
```

---

ctmaRedHet *ctmaRedHet*

Description

Computes the Reduction in Heterogeneity in drift effects after introducing study-level moderators

Usage

```
ctmaRedHet(
  activateRPB = FALSE,
  activeDirectory = NULL,
  ctmaFitObject = NULL,
  ctmaFitObjectMod = NULL,
  digits = 4,
  dt = NULL,
  undoTimeScaling = TRUE
)
```

Arguments

activateRPB	if TRUE, messages (warning, finished) could be send to smart phone (default = FALSE)
activeDirectory	the directory where to save results (if not specified, it is taken from ctmaInitFit)
ctmaFitObject	ctmaFit Object WITHOUT Moderators (obtained from ctmaFit with the arguments WEC=\'TRUE\' and scaleTI=FALSE)
ctmaFitObjectMod	ctmaFit Object WITH Moderators (obtained from ctmaFit with the arguments WEC=\'TRUE\' and scaleTI=FALSE)
digits	rounding (default = 4)
dt	A vector of scalars indicating a time interval across which discrete time effects should be estimated and then used for ctmaBiG.
undoTimeScaling	if TRUE, the original time scale is used (timeScale argument possibly used in ctmaInit is undone)

---

ctmaSaveFile *ctmaSaveFile*

Description

Internal fcuntion to save files

Usage

```
ctmaSaveFile(
  activateRPB,
  activeDirectory = activeDirectory,
  SaveObject,
  FileName,
  Directory,
  silentOverwrite = FALSE
)
```

Arguments

activateRPB	set TRUE to receive push messages with 'CoTiMA' notifications on your phone
activeDirectory	directory name
SaveObject	object to save
FileName	filename
Directory	directory to save file in
silentOverwrite	override old files without asking

Value

No return value. Just saves files

---

ctmaScaleInits *ctmaScaleInits*

Description

This function rescales inits for drifts and sets all other inits to 0 (because it is too complicated to re-scale inits for diffusions).It uses the internal trasnformations of ctStanFit (i.e., tforms) to transform the raw estimates, then re-scale them, and finally use the inverse of tfrom to supplie raw estimates as inits.

Usage

```
ctmaScaleInits(
  CoTiMAFit = NULL,
  ctsemFit = NULL,
  newTimeScale = NULL,
  autoRefit = FALSE
)
```

Arguments

CoTiMAFit	Fit object created with ctmaFit
ctsemFit	Fit object created with ctStanFit
newTimeScale	New Time scale ctStanFit
autoRefit	Whether to automatically refit the original model using the new inits

ctmaShapeRawData *ctmaShapeRawData*

Description

Raw data objects are re-shaped (dealing with missing time points, wrong time intervals etc)

Usage

```
ctmaShapeRawData(
  dataFrame = NULL,
  id = NULL,
  inputDataFrameFormat = NULL,
  inputTimeFormat = "time",
  missingValues = NA,
  n.manifest = NULL,
  manifest.per.latent = NULL,
  Tpoints = NULL,
  allInputVariablesNames = NULL,
  orderInputVariablesNames = NULL,
  targetInputVariablesNames = NULL,
  targetInputTDpredNames = NULL,
  targetInputTIpredNames = NULL,
  targetTimeVariablesNames = NULL,
  outputDataFrameFormat = "long",
  outputVariablesNames = "Y",
  outputTDpredNames = NULL,
  outputTIpredNames = NULL,
  outputTimeVariablesNames = "time",
  outputTimeFormat = "time",
  scaleTime = 1,
  minInterval = 1e-04,
  minTolDelta = NULL,
  maxTolDelta = NULL,
  negTolDelta = FALSE,
  min.val.n.Vars = 1,
  min.val.Tpoints = 1,
  standardization = "none"
)
```

Arguments

dataFrame	an R object containing data
id	the identifier of subjects if data are in long format
inputDataFrameFormat	"wide" or "long"

`inputTimeFormat`
: "time" (default) or "delta"

`missingValues`
: Missing value indicator, e.g., -999 or NA (default)

`n.manifest`
: Number of process variables (e.g, 2 in a bivariate model)

`manifest.per.latent`
: n.manifest per latent factor. Frequently 1 manifest per latent, but e.g. c(2,3,1) also possible for 6 manifest loading on 3 latents

`Tpoints`
: Number of time points in the data frame

`allInputVariablesNames`
: vector of all process variable names, time dependent predictor names, time independent predictor names, and names of times/deltas. Only required if the dataFrame does not have column names.

`orderInputVariablesNames`
: = "names" vs "time" (e.g., names: X1, X2, X3, Y1, Y2, X3 vs time: X1, Y1, X2, Y2, ...). For ctsem/CoTiMA, the output file will order by time.

`targetInputVariablesNames`
: = the process variables in the dataFrame that should be used (in "names" or in "times" order; e.g., c("X1", "X3", "Y1", "X3")). This is used to delete variables from the data frame that are not required.

`targetInputTDpredNames`
: The actual time dependent (TD) predictor variable names, e.g, 3, or 6, or 9, ... names if Tpoints = 3. Internally, each of the 3, 6, etc represents one TDpred. One typically does NOT have TD predictors in a CoTiMA.

`targetInputTIpredNames`
: time independet (TI) predictor names names in the dataFrame. One typically does NOT have TI predictors in CoTiMA except it uses raw data only, where TIpreds are avalaible for individual cases.

`targetTimeVariablesNames`
: The time variables names in the dataFrame. They also define which Tpoints will be included in the output file , e.g., c("Time4", "Time9").

`outputDataFrameFormat`
: "long" (default) or "wide"

`outputVariablesNames`
: "Y" (default; creates Y1_T0, Y2_T0, Y1_T1, Y2_T1, etc.), but can also be, e.g., c("X", "Y"; creates X_T0, Y_T0, X_T1, Y_T1, etc.).

`outputTDpredNames`
: Will become "TD" if not specified

`outputTIpredNames`
: Will become "TI" if not specified

`outputTimeVariablesNames`
: "time" (default)

`outputTimeFormat`
: "time" (default) or "delta"

`scaleTime`
: A scalar that is used to multiply the time variable. Typical use is rescaling primary study time to the time scale use in other primary studies. For example,

scaleTime=1/(60 x 60 x 24 x 365.25) rescales time provided in seconds (frequent case when imported from SPSS) into years (60sec x 60min x 24hrs x 365.25days incl. leap years).

`minInterval` A parameter (default = 0.0001) supplied to ctIntervalise. Set to smaller values than any possible observed measurement interval, but larger than 0.0001. The value is used for indicating unavailable time interval information (caused by missing values) because NA is technically not possible for time intervals.

`minTolDelta` Set, e.g. to 1/24, to delete variables from time points that are too close (e.g., 1hr; or even before) after another time point. Could be useful to delete values generated by unreliable responding, e.g., in diary studies. Note that minTolDelta applies to the time intervals AFTER the scaleTime argument has applied (i.e., scaleTime may need adaptation for each primary study, but minTolDelta does not).

`maxTolDelta` Set, e.g., to 7, to delete variables from time points that are too far after another time point (e.g., 7 days, if all participants should have responed within a week). Note that maxTolDelta applies to the time intervals AFTER the scaleTime argument has applied (i.e., scaleTime may need adaptation for each primary study, but minTolDelta does not).

`negTolDelta` FALSE (default) or TRUE. Delete entire cases that have at least one negative delta ('unreliable responding'; use minTolDelta to delete certain variables only)

`min.val.n.Vars` min.val.n.Vars = Minimum no. of valid variables. Default = 1 (retaines cases with only 1 valid variable), 0 would retain cases will all variables missing (not very useful). Retaining participants who provide a single valid variable is technically possible, but these participants contribute to the estimation of the variance/mean of this variable only. Since variance/mean are 1/0 in most CoTiMA applications, this is not very informative but at the cost of additional computational burden. Setting min.val.n.Vars = 2 is recommended.

`min.val.Tpoints`

Minimum no. of valid Tpoints (i.e. Tpoints where min.val.n.Vars is met). Default = 1 retains participants with full set of valid variables at least at one single Tpoint (which will become T0). Setting min.val.Tpoints = 2 or higher values retains participants which provide longitudinal information. Since T0 covariances are usually not too interesting, min.val.Tpoints = 2 may be more reasonable then the default = 1.

`standardization`

the way to standardize possible raw data ("none", "withinTimeA", "withinTimeB", "withinColumn", "withinPerson", or "overall"). Only applies if the list for specifying raw data information contains the list element 'standardize=TRUE'. 'WithinTimeA' standardizes within time points and deletes cases with missing T0 data. 'WithinTimeB' does not delete cases, and in subsequent ctsem or CoTiMA applications the user is adviced to use the argument 'sameInitialTimes=TRUE'.

Value

A reshaped raw data file

Examples

```
## Not run:
tmpData <- data.frame(matrix(c(1,  2,  1, 2,  1, 2,  11, 26, 1,
                               NA, NA, 3, NA, 3, NA, 12, 27, 1,
                               1,  2,  1, 2,  1, 2,  NA, 24, 0 ),
                             nrow=3, byrow=TRUE))
colnames(tmpData) <- c("first_T0", "second_T0", "first_T1", "second_T1",
                       "TD1_0", "TD1_1",
                      "time1", "time2", "sex")
shapedData <- ctmaShapeRawData(dataFrame=tmpData,
                               inputDataFrameFormat="wide",
                               inputTimeFormat="time",
                               n.manifest=2,
                               Tpoints=2,
                               orderInputVariablesNames="time",
                               targetInputVariablesNames=c("first_T0", "second_T0",
                                                           "first_T1", "second_T1"),
                               targetInputTDpredNames=c("TD1_0", "TD1_1"),
                               targetInputTIpredNames="sex",
                               targetTimeVariablesNames=c("time1", "time2"),
                               scaleTime=1/12,
                               maxTolDelta=1.2)
head(shapedData)

## End(Not run)
```

`ctmaStanResample` *ctmaStanResample*

Description

re-sample from a fitted stanct model to achieve desired number of finishsamples (could be useful to prevent exhausted memory)

Usage

```
ctmaStanResample(ctmaFittedModel = NULL, nsamples = 25, overallSamples = 500)
```

Arguments

`ctmaFittedModel`	a 'CoTiMA' fit object, usually with few 'finishsamples' to prevent memory exhaustion
`nsamples`	sample size per run
`overallSamples`	overall samples size to be achieved

Value

returns a CoTiMA fit object with an increased number of finish samples

---

`ctmaStdParams` *ctmaStdParams*

Description

Computes standardized drift effects from a CoTiMA or ctsem fit object. Can only handle CLPM or RI-CLPM fit objects.

Usage

```
ctmaStdParams(
  fit = NULL,
  times = 1,
  digits = 4,
  standardize = TRUE,
  oneTailed = FALSE
)
```

Arguments

`fit`	CoTiMA or ctsem fit object with or without random intercepts
`times`	scalar (1 by defualt) or vector of scalars defining the discrete time lags for which standardized drift effects are computed.
`digits`	rounding (4 by default)
`standardize`	logical. TRUE (default) or FALSE (does not standardize and just computes discrete time effects)
`oneTailed`	logical. FALSE (default) or TRUE. If TRUE, one-tailed CIs will be reported

Value

ctmaStdParams returns a list of standardized discrete time drift matrices for different time intervals.

Examples

```
## Not run:
ctmaStdParams(CoTiMAFullFit_3_orig, times=c(.1, 1, 2), digits=6, standardize=TRUE)

## End(Not run)
```

ctmaSV *ctmaSV*

Description

derives start values by average discrete time SEM effects, converting them to continuous time, and inversely apply transformations used by 'ctsem'

Usage

```
ctmaSV(
  ctmaInitFit = NULL,
  activeDirectory = NULL,
  primaryStudies = NULL,
  coresToUse = 1,
  replaceSV = TRUE
)
```

Arguments

ctmaInitFit	object to which all single 'ctsem' fits of primary studies has been assigned to (i.e., what has been returned by ctmaInit)
activeDirectory	defines another active directory than the one used in ctmaInit
primaryStudies	if ctmaInitFit does not contain the primaryStudies object created with ctmaPrep it could be added
coresToUse	if negative, the value is subtracted from available cores, else value = cores to use
replaceSV	if TRUE replaces startValues in primaryStudies, else it saves them as list element inits

Value

returns a modified list of primary studies with starting values added or replaced

Examples

```
## Not run:
newPrimaryStudyList <- ctmaSV(ctmaInitFit=CoTiMAInitFit_6)

## End(Not run)
```

delta_t1 *delta_t1 example vector*

Description

delta_t1 example vector

Usage

```
delta_t1
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

delta_t128 *delta_t128 example vector*

Description

delta_t128 example vector

Usage

```
delta_t128
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

delta_t18 *delta_t18 example vector*

Description

delta_t18 example vector

Usage

```
delta_t18
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

delta_t201 *delta_t201 example vector*

Description

delta_t201 example vector

Usage

```
delta_t201
```

Format

An object of class `numeric` of length 2.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

delta_t228 *delta_t228 example vector*

Description

delta_t228 example vector

Usage

```
delta_t228
```

Format

An object of class `logical` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

delta_t313 *delta_t313 example vector*

Description

delta_t313 example vector

Usage

```
delta_t313
```

Format

An object of class `numeric` of length 2.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

delta_t32 *delta_t32 example vector*

Description

delta_t32 example vector

Usage

```
delta_t32
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

delta_t4 *delta_t4 example vector*

Description

delta_t4 example vector

Usage

```
delta_t4
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

demands1 *demands1 example vector*

Description

demands1 example vector

Usage

```
demands1
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

demands128 *demands128 example vector*

Description

demands128 example vector

Usage

```
demands128
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

demands18 *demands18 example vector*

Description

demands18 example vector

Usage

```
demands18
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

demands201 *demands201 example vector*

Description

demands201 example vector

Usage

```
demands201
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

demands313 *demands313 example vector*

Description

demands313 example vector

Usage

```
demands313
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

demands32 *demands32 example vector*

Description

demands32 example vector

Usage

```
demands32
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

demands4 *demands4 example vector*

Description

demands4 example vector

Usage

```
demands4
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

dl_link *dl_link example path*

Description

dl_link example path

Usage

```
dl_link
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

empcov1 *empcov1 example matrix*

Description

empcov1 example matrix

Usage

```
empcov1
```

Format

An object of class `matrix` (inherits from `array`) with 4 rows and 4 columns.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

empcov128 *empcov128 example matrix*

Description

empcov128 example matrix

Usage

```
empcov128
```

Format

An object of class `matrix` (inherits from `array`) with 4 rows and 4 columns.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

empcov18 *empcov18 example matrix*

Description

empcov18 example matrix

Usage

```
empcov18
```

Format

An object of class `matrix` (inherits from `array`) with 4 rows and 4 columns.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

empcov201 *empcov201 example matrix*

Description

empcov201 example matrix

Usage

```
empcov201
```

Format

An object of class `matrix` (inherits from `array`) with 6 rows and 6 columns.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

empcov313 *empcov313 example matrix*

Description

empcov313 example matrix

Usage

```
empcov313
```

Format

An object of class `matrix` (inherits from `array`) with 6 rows and 6 columns.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

empcov32 *empcov32 example matrix*

Description

empcov32 example matrix

Usage

```
empcov32
```

Format

An object of class `matrix` (inherits from `array`) with 4 rows and 4 columns.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

empcov4 *empcov4 example matrix*

Description

empcov4 example matrix

Usage

```
empcov4
```

Format

An object of class `matrix` (inherits from `array`) with 4 rows and 4 columns.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

malePercent1 *malePercent1 example vector*

Description

malePercent1 example vector

Usage

```
malePercent1
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

malePercent128 *malePercent128 example vector*

Description

malePercent128 example vector

Usage

```
malePercent128
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

malePercent18 *malePercent18 example vector*

Description

malePercent18 example vector

Usage

```
malePercent18
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

malePercent201 *malePercent201 example vector*

Description

malePercent201 example vector

Usage

```
malePercent201
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

malePercent313 *malePercent313 example vector*

Description

malePercent313 example vector

Usage

```
malePercent313
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

malePercent32 *malePercent32 example vector*

Description

malePercent32 example vector

Usage

```
malePercent32
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

malePercent4 *malePercent4 example vector*

Description

malePercent4 example vector

Usage

```
malePercent4
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

moderator1 *moderator1 example vector*

Description

moderator1 example vector

Usage

```
moderator1
```

Format

An object of class `numeric` of length 2.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

moderator128 *moderator128 example vector*

Description

moderator128 example vector

Usage

```
moderator128
```

Format

An object of class `numeric` of length 2.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

moderator18 *moderator18 example vector*

Description

moderator18 example vector

Usage

```
moderator18
```

Format

An object of class `numeric` of length 2.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

moderator201 *moderator201 example vector*

Description

moderator201 example vector

Usage

```
moderator201
```

Format

An object of class `numeric` of length 2.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

moderator313 *moderator313 example vector*

Description

moderator313 example vector

Usage

```
moderator313
```

Format

An object of class `numeric` of length 2.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

moderator32 *moderator32 example vector*

Description

moderator32 example vector

Usage

```
moderator32
```

Format

An object of class `numeric` of length 2.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

moderator4 *moderator4 example vector*

Description

moderator4 example vector

Usage

```
moderator4
```

Format

An object of class `numeric` of length 2.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

moderatorLabels *moderatorLabels example vector*

Description

moderatorLabels example vector

Usage

```
moderatorLabels
```

Format

An object of class `character` of length 2.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

moderatorValues *moderatorValues example vector*

Description

moderatorValues example vector

Usage

```
moderatorValues
```

Format

An object of class `list` of length 2.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

occupation1 *occupation1 example vector*

Description

occupation1 example vector

Usage

```
occupation1
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

occupation128 *occupation128 example vector*

Description

occupation128 example vector

Usage

```
occupation128
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

occupation18 *occupation18 example vector*

Description

occupation18 example vector

Usage

```
occupation18
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

occupation201 *occupation201 example vector*

Description

occupation201 example vector

Usage

```
occupation201
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

occupation313 *occupation313 example vector*

Description

occupation313 example vector

Usage

```
occupation313
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

occupation32 *occupation32 example vector*

Description

occupation32 example vector

Usage

```
occupation32
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

occupation4 *occupation4 example vector*

Description

occupation4 example vector

Usage

```
occupation4
```

Format

An object of class `character` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

pairwiseN128 *pairwiseN128 example vector*

Description

pairwiseN128 example vector

Usage

```
pairwiseN128
```

Format

An object of class `matrix` (inherits from `array`) with 9 rows and 9 columns.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

plot.CoTiMAFit *plot.CoTiMAFit*

Description

call `ctmaPlot` if a CoTiMAFit object is supplied to plot()

Usage

```
## S3 method for class 'CoTiMAFit'
plot(x, ...)
```

Arguments

x	líst
...	further arguments to be passed through to summary()

Value

returns a call to 'ctmaPlot', which is used to plot CoTiMA fit objects

pubList_8 *pubList_8 example list*

Description

pubList_8 example list

Usage

```
pubList_8
```

Format

An object of class `CoTiMAFit` of length 9.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

rawData228 *rawData228 example list*

Description

rawData228 example list

Usage

```
rawData228
```

Format

An object of class `list` of length 7.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

recodeVariables128 *recodeVariables128 example vector*

Description

recodeVariables128 example vector

Usage

```
recodeVariables128
```

Format

An object of class `character` of length 2.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

results128 *results128 example list*

Description

results128 example list

Usage

```
results128
```

Format

An object of class `list` of length 3.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

sampleSize1 *sampleSize1 example vector*

Description

sampleSize1 example vector

Usage

```
sampleSize1
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

sampleSize128 *sampleSize128 example vector*

Description

sampleSize128 example vector

Usage

```
sampleSize128
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

sampleSize18 *sampleSize18 example vector*

Description

sampleSize18 example vector

Usage

```
sampleSize18
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

sampleSize201 *sampleSize201 example vector*

Description

sampleSize201 example vector

Usage

```
sampleSize201
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

sampleSize313 *sampleSize313 example vector*

Description

sampleSize313 example vector

Usage

```
sampleSize313
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

sampleSize32 *sampleSize32 example vector*

Description

sampleSize32 example vector

Usage

```
sampleSize32
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

sampleSize4 *sampleSize4 example vector*

Description

sampleSize4 example vector

Usage

```
sampleSize4
```

Format

An object of class `numeric` of length 1.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

source1 *source1 example vector*

Description

source1 example vector

Usage

```
source1
```

Format

An object of class `character` of length 6.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

source128 *source128 example vector*

Description

source128 example vector

Usage

```
source128
```

Format

An object of class `character` of length 4.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

source18 *source18 example vector*

Description

source18 example vector

Usage

```
source18
```

Format

An object of class `character` of length 4.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

source201 *source201 example vector*

Description

source201 example vector

Usage

```
source201
```

Format

An object of class `character` of length 6.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

source313 *source313 example vector*

Description

source313 example vector

Usage

```
source313
```

Format

An object of class `character` of length 4.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

source4 *source4 example vector*

Description

source4 example vector

Usage

```
source4
```

Format

An object of class `character` of length 6.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

summary.CoTiMAFit *summary.CoTiMAFit*

Description

defines summary for 'CoTiMA' fit objects

Usage

```
## S3 method for class 'CoTiMAFit'
summary(object, ...)
```

Arguments

`object`	one CoTiMAFit object or more as ctmaFitList(object1, object2, ...)
`...`	further arguments to be passed through to summary()

Value

returns a printed summary of a 'CoTiMA' fit object

targetVariables1 *targetVariables1 example vector*

Description

targetVariables1 example vector

Usage

```
targetVariables1
```

Format

An object of class `character` of length 4.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

targetVariables128 *targetVariables128 example vector*

Description

targetVariables128 example vector

Usage

```
targetVariables128
```

Format

An object of class `character` of length 7.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

targetVariables313 *targetVariables313 example vector*

Description

targetVariables313 example vector

Usage

```
targetVariables313
```

Format

An object of class `character` of length 6.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

targetVariables4 *targetVariables4 example vector*

Description

targetVariables4 example vector

Usage

```
targetVariables4
```

Format

An object of class `character` of length 4.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

variableNames128 *variableNames128 example vector*

Description

variableNames128 example vector

Usage

```
variableNames128
```

Format

An object of class `character` of length 9.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

variableNames18 *variableNames18 example vector*

Description

variableNames18 example vector

Usage

```
variableNames18
```

Format

An object of class `character` of length 4.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

variableNames201 *variableNames201 example vector*

Description

variableNames201 example vector

Usage

```
variableNames201
```

Format

An object of class `character` of length 6.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

variableNames32 *variableNames32 example vector*

Description

variableNames32 example vector

Usage

```
variableNames32
```

Format

An object of class `character` of length 4.

Author(s)

C. Dormann & M. Homburg <CoTiMA@uni-mainz.org>

Index