Introduction to CTmeta: functions for lagged effects model parameters

R. M. Kuiper

Contents

Introduction

CTmeta is created to preform meta-analysis on time-interval dependent discrete-time (DT) lagged parameters estimates (e.g., CLPM or VAR(1) estimates). Its method utilizes an underlying continuous-time (CT) model; hence, the name CTmeta. Since its creation, many functions have been added to the package `CTmeta`, including functions to transform, standardize, and plot the time-interval dependent discrete-time (DT) lagged parameters.

More specifically, some of functionalities include:

- Rendering plots that illustrate how the VAR(1) model lagged parameters (Phi) and residual covariance matrix (SigmaVAR) vary as a function of the time interval (DeltaT). See the functions `PhiPlot` and `SigmaVARPlot`, respectively.
- Determining the time interval (DeltaT) for which each element of Phi(DeltaT) reaches its minimum or maximum. See the function `MaxDeltaT`.
- Rendering the time interval for which the VAR(1) model residual covariance matrix is a diagonal matrix. See the function `DiagDeltaT`.
- Rendering standardized lagged relationships from their unstandardized counterparts; or from cross-correlations and their covariance matrix. See the functions `StandPhi` and `TransPhi_Corr`, respectively. These functions also render the univariate (i.e., simultaneous) and multivariate (i.e., elliptical) confidence intervals of the (un)standardized lagged relationships.
- Transforming (un)standardized VAR(1) model lagged parameters and the residual covariance matrix to reflect scenarios in which another time interval is used. See the function `StandTransPhi`. This function also renders the univariate (i.e., simultaneous) and multivariate (i.e., elliptical) confidence intervals of the (un)standardized lagged relationships.
- Transforming VAR(1) model lagged parameters and the residual covariance matrix into their (un)standardized counterparts in the underlying CT(1) model and vice versa. See the functions `CTMparam` and `VARparam`, respectively.

Below, one can find examples demonstrating the application of these functions.

More details

More details about the methods and derivations can be found in:

- Kuiper, R. M., and Hamaker, E.L. (unpublished). Correlated residuals in a VAR model: What they do (not) represent.
- Kuiper, R. M., & Ryan, O. (2020). Meta-analysis of Lagged Regression Models: A Continuous-time Approach. Structural Equation Modeling: A Multidisciplinary Journal, 27(3), 396-413. https://doi.org/10.1080/10705511.2019.1652613
- Kuiper, R. M. (2021). Evaluating Causal Dominance of CTmeta-Analyzed Lagged Regression Estimates, Structural Equation Modeling: A Multidisciplinary Journal, 28(6), 951-963. https://doi-org.proxy.library.uu.nl/10.1080/10705511.2020.1823228
- Altınısık, Y., Van Lissa, C. J., Hoijtink, H., Oldehinkel, A. J., and Kuiper, R. M. (2021). Evaluation of inequality constrained hypotheses using a generalization of the AIC. Psychological Methods, 26(5), 599-621. https://doi.org/10.1037/met0000406.

Installation and descriptions

```
# Install R package
# Note: Make sure you have Rtools
#       (and a version which is compatible with your R version).
library(devtools)
install_github("rebeccakuiper/CTmeta")

# Load package
library(CTmeta)

# In case you use functions from this CTmeta package, please cite it:
citation("CTmeta")

# To look at the description of a function including example code,
# use ?functionname:
?Area
?ChecksCTM
?CTmeta
?CTMparam
?DiagDeltaT
?Gamma.fromCTM
?Gamma.fromVAR
?ggPhiPlot
?MaxDeltaT
?PhiPlot
?SigmaVARPlot
?StandPhi
?StandTransPhi
?TransPhi_Corr
?VARparam

# To obtain an overview of all functions in the package and their arguments:
lsf.str("package:CTmeta")
```

Usage

Area(DeltaT = 1, Phi, ...)

ChecksCTM(Drift, Sigma)

CTmeta(N, DeltaT, DeltaTStar, Phi, SigmaVAR, ...)

CTMparam(DeltaT, Phi, SigmaVAR, ...)

DiagDeltaT <- function(Phi, SigmaVAR, ...)

Gamma.fromCTM(Drift, Sigma)

Gamma.fromVAR(Phi, SigmaVAR)

ggPhiPlot(DeltaT = 1, Phi, ...)

MaxDeltaT(DeltaT = 1, Phi, ...)

PhiPlot(DeltaT = 1, Phi, ...)

SigmaVARPlot(DeltaT = 1, Phi, SigmaVAR, ...)

StandPhi(N = NULL, Phi, SigmaVAR, ...)

StandTransPhi(DeltaTStar, DeltaT = 1, N = NULL, Phi, SigmaVAR, ...)

TransPhi_Corr(DeltaTStar, DeltaT = 1, N = NULL, corr_YXYX, ...)

VARparam(DeltaT = 1, Drift, Sigma, ...)

Examples

CTmeta

This section shows the functionality of `CTmeta` demonstrated by using a simple example with $S = 3$ primary studies. These three studies investigate the (cross-)lagged relationship between $q = 2$ variables: Stress and Anxiety. Thus, each study generates a 2 x 2 lagged relationships matrix. All studies used different samples, different sample sizes, and collected the data using different time intervals.

The function `CTmeta` requires the following arguments as input: N, DeltaT, DeltaTStar, Phi, and one of SigmaVAR or Gamma, details are included below.

Setting up the arguments for `CTmeta`:

- N: the sample size of each study in a S x 1 matrix.

```
N <- matrix(c(643, 651, 473))
```

- DeltaT: the time interval used in each study in a S x 1 matrix.

```
DeltaT <- matrix(c(2, 3, 1))
```

- DeltaTStar: the constant representing the time interval of interest (preferably, falling within the range of the inspected time intervals).

```
DeltaTStar <- 1
```

- Phi: the stacked discrete-time lagged relationships matrices. For each of the $S = 3$ studies, retrieve the q x q lagged relationships matrix; in this specific example, there are three 2 x 2 lagged relationships matrices. Then, stack the three matrices to form a matrix (Phi) with dimensions $S * q$ x q. This argument has an example matrix stored in the package:

```
Phi <- myPhi

Phi
>       [,1] [,2]
> [1,] 0.25 0.10
> [2,] 0.20 0.36
> [3,] 0.35 0.20
> [4,] 0.30 0.46
> [5,] 0.15 0.00
> [6,] 0.10 0.26
```

Note: For each study, the lagged relationships matrix (Phi) varies as a function of the time interval (DeltaT), hence the notation Phi(DeltaT). For instance, if Stress and Anxiety are measured every hour, Phi(1) reflects the lagged relationship between initial Stress and Anxiety levels and Stress and Anxiety levels after one hour. This relationship is stronger than (different from) a scenario in which Stress and Anxiety are measured daily (24 hours).

- SigmaVAR: the stacked residual covariance matrices. For each of the $S = 3$ studies, retrieve the q x q residual covariance matrix. Then, stack the matrices to form a matrix (SigmaVAR) with dimensions $S * q$ x q. This argument has an example matrix stored in the package:

```
SigmaVAR <- mySigmaVAR
```

- Gamma: the stacked stationary covariance matrices. For each of the $S = 3$ studies, generate the q x q stationary covariance matrix. Then, stack the matrices to form a matrix (Gamma) with dimensions $S * q$ x q. This argument has an example matrix stored in the package:

```
Gamma <- myGamma # Note: CTmeta does not need both SigmaVAR and Gamma
```

The `CTmeta` function will standardize these matrices (Phi, SigmaVAR, Gamma) to make the comparisons and weighted averages of lagged relationships estimates meaningful.

As with regular meta-analysis, one can choose between a fixed-effects and random-effects model, and moderators can also be included. Notably, the default is a fixed-effects model, but if one wants to generalize the results beyond the included studies, then one should use a random-effects model ('FEorRE = 2'). In case the lagged relationships matrices stacked in Phi are expected to be different / incomparable due to some study characteristics, then these should be included in the model. If, for example, some studies investigated the Stress-Anxiety relationship among students while other studies did this in a business context, a moderator can be included by adding a dummy variable for context. Note that when the parameter estimate for this moderator is significant, then there is a significant difference in the Stress-Anxiety relationship between the two subgroups.

The following code will demonstrate the use of CTmeta for four different types of meta-analyses. Note that only the output for the last model (the random-effects model with moderators) is shown. Subsequently, code is shown for six different types of output that can be requested.

```
### Example without moderators ###

## Fixed effects model (default) ##
CTmeta(N, DeltaT, DeltaTStar, Phi, SigmaVAR)

## Random effects model ##
CTmeta(N, DeltaT, DeltaTStar, Phi, SigmaVAR, FEorRE = 2)


### Example with moderators ###
#
```

```
# 1 moderator
Mod <- matrix(c(64,65,47))
#
# two moderators, in each column 1
#Mod <- matrix(cbind(c(64,65,47), c(78,89,34)), ncol = q);
#colnames(Mod) <- c("Mod1", "Mod2")

## Fixed effects model (default) ##
CTmeta(N, DeltaT, DeltaTStar, Phi, SigmaVAR, Moderators = 1, Mod = Mod)
```

```
## Random effects model ##
CTmeta(N, DeltaT, DeltaTStar, Phi, SigmaVAR,
       Moderators = 1, Mod = Mod, FEorRE = 2)
>
> The overall estimates obtained with CTmeta and their 95% elliptical/multivariate confidence interval:
>
>              Overall_Phi LB     UB
> overallPhi11 -1.019      -1.924 -0.114
> overallPhi12 -0.340      -0.869  0.188
> overallPhi21 -0.125      -0.605  0.354
> overallPhi22 -0.846      -1.577 -0.114
>
> Note: A random-effects model is used. The tau^2 values can be obtained via '$tau2'; see '$summaryMeta
>
> CTmeta Messages:
> - All eigenvalues are positive and real. Hence, the Phi's are transformed to Phi(DeltaT*) to account
> - For each study, the covariance matrix is positive definite. Hence, a multivariate approach is used.
```

Next, some code is shown for six different types of output that can be requested.

```
## Different output options are possible ##
CTma <- CTmeta(N, DeltaT, DeltaTStar, Phi, SigmaVAR,
               Moderators = 1, Mod = Mod, FEorRE = 2)
```

```
# 1
CTma
>
> The overall estimates obtained with CTmeta and their 95% elliptical/multivariate confidence interval:
>
>              Overall_Phi LB     UB
> overallPhi11 -1.019      -1.924 -0.114
> overallPhi12 -0.340      -0.869  0.188
> overallPhi21 -0.125      -0.605  0.354
> overallPhi22 -0.846      -1.577 -0.114
>
> Note: A random-effects model is used. The tau^2 values can be obtained via '$tau2'; see '$summaryMeta
>
> CTmeta Messages:
> - All eigenvalues are positive and real. Hence, the Phi's are transformed to Phi(DeltaT*) to account
> - For each study, the covariance matrix is positive definite. Hence, a multivariate approach is used.
```

```
# 2
print(CTma)
>
> The overall estimates obtained with CTmeta and their 95% elliptical/multivariate confidence interval:
```

```
>
>              Overall_Phi LB     UB
> overallPhi11 -1.019      -1.924 -0.114
> overallPhi12 -0.340      -0.869  0.188
> overallPhi21 -0.125      -0.605  0.354
> overallPhi22 -0.846      -1.577 -0.114
>
> Note: A random-effects model is used. The tau^2 values can be obtained via '$tau2'; see '$summaryMeta
>
> CTmeta Messages:
> - All eigenvalues are positive and real. Hence, the Phi's are transformed to Phi(DeltaT*) to account
> - For each study, the covariance matrix is positive definite. Hence, a multivariate approach is used.
```

```
# 3
summary(CTma)
> +------------------+-------------+---------+---------+
> |                  | Overall_Phi |      LB |      UB |
> +==================+=============+=========+=========+
> | overallPhi11     |      -1.019 |  -1.924 | -0.1139 |
> +------------------+-------------+---------+---------+
> | overallPhi12     |     -0.3405 | -0.8688 |  0.1878 |
> +------------------+-------------+---------+---------+
> | overallPhi21     |     -0.1254 | -0.6053 |  0.3545 |
> +------------------+-------------+---------+---------+
> | overallPhi22     |     -0.8457 |  -1.577 | -0.1144 |
> +------------------+-------------+---------+---------+
```

```
# 4
print(CTma, digits = 4)
>
> The overall estimates obtained with CTmeta and their 95% elliptical/multivariate confidence interval:
>
>              Overall_Phi LB      UB
> overallPhi11 -1.0189     -1.9240 -0.1139
> overallPhi12 -0.3405     -0.8688  0.1878
> overallPhi21 -0.1254     -0.6053  0.3545
> overallPhi22 -0.8457     -1.5770 -0.1144
>
> Note: A random-effects model is used. The tau^2 values can be obtained via '$tau2'; see '$summaryMeta
>
> CTmeta Messages:
> - All eigenvalues are positive and real. Hence, the Phi's are transformed to Phi(DeltaT*) to account
> - For each study, the covariance matrix is positive definite. Hence, a multivariate approach is used.
```

```
# 5
summary(CTma, digits = 4)
> +------------------+-------------+---------+---------+
> |                  | Overall_Phi |   LB    |   UB    |
> +==================+=============+=========+=========+
> |   overallPhi11   |   -1.0189   | -1.9240 | -0.1139 |
> +------------------+-------------+---------+---------+
> |   overallPhi12   |   -0.3405   | -0.8688 | 0.1878  |
> +------------------+-------------+---------+---------+
> |   overallPhi21   |   -0.1254   | -0.6053 | 0.3545  |
> +------------------+-------------+---------+---------+
```

```
> |   overallPhi22   |   -0.8457   | -1.5770 | -0.1144 |
> +------------------+-------------+---------+---------+
```

```
# 6
# In Rstudio, use 'CTma$' to see what output options are available.
# For example:
CTma$summaryMetaAnalysis
>
> Multivariate Meta-Analysis Model (k = 12; method: ML)
>
>   logLik  Deviance       AIC       BIC      AICc
>  25.2754    9.0869  -14.5508   -5.8225  669.4492
>
> Variance Components:
>
> outer factor: Study      (nlvls = 3)
> inner factor: overallPhi (nlvls = 4)
>
>             estim    sqrt  k.lvl  fixed  level
> tau^2.1    0.0039  0.0627      3     no     11
> tau^2.2    0.0003  0.0162      3     no     12
> tau^2.3    0.0000  0.0000      3     no     21
> tau^2.4    0.0027  0.0516      3     no     22
>
>      rho.11   rho.12   rho.21  rho.22    11  12  21  22
> 11        1                             -   3   3   3
> 12   1.0000        1                   no   -   3   3
> 21  -1.0000  -1.0000        1          no  no   -   3
> 22   1.0000   1.0000  -1.0000       1   no  no  no   -
>
> Test for Residual Heterogeneity:
> QE(df = 4) = 28.1058, p-val < .0001
>
> Test of Moderators (coefficients 1:8):
> QM(df = 8) = 404.0755, p-val < .0001
>
> Model Results:
>
>                    estimate      se     zval    pval    ci.lb    ci.ub
> overallPhi11        -1.0189  0.3143  -3.2419  0.0012  -1.6350  -0.4029   **
> overallPhi12        -0.3405  0.1891  -1.8004  0.0718  -0.7112   0.0302    .
> overallPhi21        -0.1254  0.1696  -0.7396  0.4596  -0.4578   0.2070
> overallPhi22        -0.8457  0.2732  -3.0958  0.0020  -1.3812  -0.3103   **
> overallPhi11:Mod.    0.0248  0.0052   4.7253  <.0001   0.0145   0.0350  ***
> overallPhi12:Mod.    0.0072  0.0031   2.3397  0.0193   0.0012   0.0133    *
> overallPhi21:Mod.    0.0047  0.0027   1.7351  0.0827  -0.0006   0.0101    .
> overallPhi22:Mod.    0.0234  0.0045   5.1741  <.0001   0.0146   0.0323  ***
>
> ---
> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

This results, among other things, in a 2 x 2 overall Phi matrix (if $q = 2$). Although the varying time intervals are corrected for, this overall Phi matrix is also dependent on the chosen time interval (i.e., DeltaTStar). To obtain insight into this, a Phi-plot of the resulting overall Phi can be made. This is demonstrated in Section

1.1.

A major point of interest with regards to lagged relationships is to determine the 'causal dominant' variable, that is, the variable with the highest predictive power. In this example, we are interested in the question: Is current Stress a stronger predictor for Anxiety on the next time interval or is current Anxiety a stronger predictor for Stress on the next time interval? To evaluate the dominance of (overall) lagged relationships, an AIC-type criterion called the GORICA can be used (Altinisik, . . . , Kuiper, 2021). This method is included in the `restriktor` package and is demonstrated in Section 1.2.

Finally, in the example above, the lagged relationships matrices (stacked in Phi) were obtained for each primary study. It is also possible that some or all primary studies report a (lagged) correlation matrix. Section 1.3 demonstrates how (lagged) correlation matrices can be used in `CTmeta`.

Phi-plot of resulting overall Phi

Make customized Phi-plot of resulting overall Phi:

```
# Option 1: Use the plot option in the 'CTmeta' function.

CTmeta(N, DeltaT, DeltaTStar, Phi, SigmaVAR, PrintPlot = TRUE)
>
> The overall estimates obtained with CTmeta and their 95% elliptical/multivariate confidence interval:
>
>              Overall_Phi LB     UB
> overallPhi11 0.519       0.4771 0.562
> overallPhi12 0.139       0.0964 0.181
> overallPhi21 0.198       0.1638 0.233
> overallPhi22 0.624       0.5897 0.659
>
> CTmeta Messages:
> - All eigenvalues are positive and real. Hence, the Phi's are transformed to Phi(DeltaT*) to account f
> - For each study, the covariance matrix is positive definite. Hence, a multivariate approach is used.
```

```
# Option 2: Use the interactive web application:
# (\url{https://www.uu.nl/staff/RMKuiper/Websites\%20\%2F\%20Shiny\%20apps}).
```

```
# Option 3: Use the 'PhiPlot' function.

# First, extract the (q times q) overall Phi matrix
out_CTmeta <- CTmeta(N, DeltaT, DeltaTStar, Phi, SigmaVAR)
# resulting overall Phi:
overallPhi <- out_CTmeta$Overall_standPhi

# Make Phi-plot:
Title <- as.list(expression(Phi(Delta[t])~plot),
 "How do the overall lagged parameters vary as a function of the time interval")
PhiPlot(DeltaTStar, overallPhi, Min = 0, Max = 40, Step = 0.5, Title = Title)
> NULL
```

```
# Option 4: The function 'ggPhiPlot' can be used instead of 'PhiPlot'.
phi_plot <- ggPhiPlot(DeltaTStar, overallPhi,
                      Min = 0, Max = 40, Step = 0.5, Title = Title)
print(phi_plot$PhiPlot)
```

Evaluate dominance of overall Phi using GORICA

Evaluate dominance of overall lagged relationships matrix (overallPhi) with the GORICA using the `goric` function in the `restriktor` package:

```
# Specify hypothesis
H1 <- "overallPhi12 < overallPhi21"
#H2 <- "overallPhi12 > overallPhi21" or:
# vs its complement (default in case of one hypothesis)
#
# Btw if signs can be negative one perhaps better use:
# H1 <- "abs(overallPhi12) < abs(overallPhi21)"

# Evaluate dominance of cross-lagged using the GORICA
if (!require("restriktor")) install.packages("restriktor")
> Loading required package: restriktor
> This is restriktor 0.5-90
> Please report any bugs to info@restriktor.org
```

```
# Use restriktor package for function goric().
# Authors of goric(): Vanbrabant and Kuiper.
library(restriktor)
set.seed(123) # for reproducability of results and possible sensitivity check of penalty
#
# Option 1:
# Extract the vectorized overall standardized Phi matrix
# and its covariance matrix using the functions coef() and vcov(), respectively:
#out_CTmeta <- CTmeta(N, DeltaT, DeltaTStar, Phi, SigmaVAR)
#est <- coef(out_CTmeta)  # or: est <- out_CTmeta$Overall_vecStandPhi_DeltaTStar
#VCOV <- vcov(out_CTmeta) # or: VCOV <- out_CTmeta$CovMx_OverallPhi_DeltaTStar
## GORICA (default)
#goric(est, VCOV = VCOV, hypotheses = list(H1))
#
# Option 2
# GORICA (default)
goricaResult <- goric(out_CTmeta, hypotheses = list(H1))
#summary(goricaResult)
goricaResult
> restriktor (0.5-90): generalized order-restricted information criterion approximation:
>
> Results:
>         model  loglik  penalty   gorica  loglik.weights  penalty.weights
> 1          H1  12.370    3.500  -17.739           0.936            0.500
> 2  complement   9.690    3.500  -12.379           0.064            0.500
>    gorica.weights
> 1           0.936
> 2           0.064
> ---
> The order-restricted hypothesis 'H1' has 14.58 times more support than its complement.
```

From this, one concludes that H1: overallPhi12 < overallPhi21 is more supported / more likely than its complement (here, overallPhi12 > overallPhi21). Hence, there is support in favor of our theory w.r.t. predictive strength / the ordering of the size of the cross-lagged relationships.

A (lagged) correlation matrix as input

If primary studies report a (lagged) correlation matrix rather than reporting the lagged relationships, then do the following:

```
q <- 2 # Recall there are q=2 variables of interest.

# Suppose all primary studies reported the following lagged correlation matrix:
corr_YXYX <- matrix(c(1.00, 0.40, 0.63, 0.34,
                      0.40, 1.00, 0.31, 0.63,
                      0.63, 0.31, 1.00, 0.41,
                      0.34, 0.63, 0.41, 1.00), byrow = T, ncol = 2*q)

# In this example, the previous N and DeltaT(Star) values are used:
N <- matrix(c(643, 651, 473))
DeltaT <- matrix(c(2, 3, 1))
DeltaTStar <- 1

# First, use the function 'TransPhi_Corr' to calculate the corresponding
# standardized lagged relationships matrix for each of the 3 primary studies:

# first study
out_1 <- TransPhi_Corr(DeltaTStar = DeltaT[1], DeltaT = 1, N = N[1], corr_YXYX)
Phi_1 <- out_1$standPhi_DeltaTStar
SigmaVAR_1 <- out_1$standSigmaVAR_DeltaTStar

# second study
out_2 <- TransPhi_Corr(DeltaTStar = DeltaT[2], DeltaT = 1, N = N[2], corr_YXYX)
Phi_2 <- out_2$standPhi_DeltaTStar
SigmaVAR_2 <- out_2$standSigmaVAR_DeltaTStar

# third study
out_3 <- TransPhi_Corr(DeltaTStar = DeltaT[3], DeltaT = 1, N = N[3], corr_YXYX)
Phi_3 <- out_3$standPhi_DeltaTStar
SigmaVAR_3 <- out_3$standSigmaVAR_DeltaTStar

# Note: one can already make the time intervals equal via
#       the arguments DeltaTStar and DeltaT in the TransPhi_Corr function,
#       but CTmeta can do this as well.

# In this example, the time intervals are deliberately unequal such that
# - the example is in line with the input (i.e., DeltaT <- matrix(c(2, 3, 1)))
# - the resulting overall Phi should equal the Phi that underlies this lagged
#   correlation matrix.

# Next, make the stacked matrices Phi and SigmaVAR, both of size (S*q) * q:
Phi <- rbind(Phi_1, Phi_2, Phi_3)
SigmaVAR <- rbind(SigmaVAR_1, SigmaVAR_2, SigmaVAR_3)

# Run CTmeta:
CTmeta(N, DeltaT, DeltaTStar, Phi, SigmaVAR)

# Finally, retrieve the overall q x q (here, 2x2) lagged relationships matrix:
out_CTmeta <- CTmeta(N, DeltaT, DeltaTStar, Phi, SigmaVAR)
out_CTmeta$Overall_standPhi
```

Time-interval dependency

This section focuses on the time-interval dependency of (cross-)lagged relationships for one study. However, it can also be used for the overall (cross-)lagged relationships resulting from `CTmeta`.

Let us investigate the (cross-)lagged relationship between Stress and Anxiety (i.e., between q=2 variables), represented by a 2 x 2 discrete-time lagged relationships matrix, Phi, and a 2 x 2 discrete-time residual covariance matrix, SigmaVAR (called Psi in Kuiper and Hamaker).

Note: Both the lagged relationships matrix (Phi) and the residual covariance matrix (SigmaVAR) vary with the time interval (DeltaT). For example, if Stress and Anxiety were measured every hour, Phi would reflect the lagged relationship between current Stress and Anxiety levels and the Stress and Anxiety levels over one hour. Likewise, the residual covariance matrix (SigmaVAR) would reflect the scenario in which the measurements of Stress and Anxiety were taken every hour.

Thus, it follows that for a long enough time interval, lagged relationships damp out, that is, the autoregressive and cross-lagged relationships are zero. Consequently, the residual covariance matrix equals the stationary covariance matrix (Gamma) which equals the (contemporaneous) covariance matrix of the contemporaneous variables.

In this section, we will make use of the following matrices:

```
## Discrete-time matrices

# Phi(DeltaT) & SigmaVARDeltaT)
DeltaT <- 1            # Specify time observed time interval (DeltaT)
Phi <- myPhi[1:2,1:2]  # Select part that corresponds to one primary study
SigmaVAR <- diag(2) # for ease

# Fitted object of class "varest"
DeltaT <- 1
data <- myData
#if (!require("vars")) install.packages("vars")
library(vars)
> Loading required package: MASS
>
> Attaching package: 'MASS'
> The following object is masked from 'package:dplyr':
>
>     select
> Loading required package: strucchange
> Loading required package: zoo
>
> Attaching package: 'zoo'
> The following objects are masked from 'package:base':
>
>     as.Date, as.Date.numeric
> Loading required package: sandwich
> Loading required package: urca
> Loading required package: lmtest
```

```
out_VAR <- VAR(data, p = 1)


## Continuous-time equivalents Drift & Sigma
# obtained via the function CTMparam(), see a later section as well.
CTparam <- CTMparam(DeltaT, Phi, SigmaVAR)
```

```
Drift <-CTparam$Drift
Sigma <-CTparam$Sigma
```

2.1 Plots

To visualize the time-interval dependency, there are functions that plot the elements of these matrices for a range of time intervals:

```
### Make Phi-plot ###

## Example 1 ##

# Example 1.1: unstandardized Phi #
PhiPlot(DeltaT, Phi)
> NULL
```

```
# Example 1.2: standardized Phi #
SigmaVAR <- diag(2) # for ease
PhiPlot(DeltaT, Phi, Stand = 1, SigmaVAR = SigmaVAR)
> NULL
```

```
## Example 2: input from fitted object of class "varest" ##

# Example 2.1: unstandardized Phi #
PhiPlot(DeltaT, out_VAR)

# Example 2.2: standardized Phi #
PhiPlot(DeltaT, out_VAR, Stand = 1)
```

```
## Example 3: Change plot options ##
# Note: use Phi from Example 1
q <- dim(Phi)[1]

WhichElements <- matrix(1, ncol = q, nrow = q) # Now, all elements are 1
diag(WhichElements) <- 0 # Now, the autoregressive parameters are excluded
                         # by setting the diagonals to 0.
Lab <- c("12", "21")
Labels <- NULL
for(i in 1:length(Lab)){
 e <- bquote(expression(Phi(Delta[t])[.(Lab[i])]))
 Labels <- c(Labels, eval(e))
}

Col <- c(1,2)
Lty <- c(1,2)
# Standardized Phi
ggPhiPlot(DeltaT = 1, Phi, Stand = 1, SigmaVAR = SigmaVAR,
        Min = 0, Max = 10, Step = 0.05,
        WhichElements = WhichElements, Labels = Labels, Col = Col, Lty = Lty)

# Note that when you use 'ggPhiPlot',
# then you can customize the plot like you would do with a regular ggplot.
```

```
### Make Psi-plot/SigmaVAR-plot ###
```

```
# Example 1.1: unstandardized Phi&SigmaVAR #
SigmaVARPlot(DeltaT, Phi, SigmaVAR)

# Example 1.2: standardized Phi&SigmaVAR #
SigmaVARPlot(DeltaT, Phi, SigmaVAR, Stand = 1)

# Notes:
# - Like in the Phi-plot, a 'varest' object can be used.
# - Like in the Phi-plot, the plot can be customized
#   (but there is no ggplot variant)
```

One can also use the following continuous-time matrices to generate Phi- and SigmaVAR-plots: Drift, the underlying continuous-time lagged relationships matrix, and Sigma, the continuous-time residual covariance matrix (also called the diffusion matrix). Then, one should use the following code:

```
# Phi-plot: unstandardized Drift #
PhiPlot(DeltaT, Drift = Drift, Min = 0, Max = 10, Step = 0.01)

# Phi-plot: standardized Drift #
PhiPlot(DeltaT, Drift = Drift, Stand = 1, Sigma = Sigma)

# SigmaVAR-plot: unstandardized Drift & Sigma / Phi & SigmaVAR # .
SigmaVARPlot(DeltaT, Drift = Drift, Sigma = Sigma,
             Min = 0, Max = 10, Step = 0.01)
```

2.1.1 Area under curves in Phi-plot The area under a curve is the magnitude of the displacement, which is equal to the distance traveled (only for constant acceleration). As a comparison, when the plot shows the variation of a drug concentration as a function of time, the area under the curve (from zero to infinity) represents the total drug exposure across time. Such a measure might be interesting when comparing lagged relationships matrices using different formulations/operationalisations (e.g., in the drugs example, capsule vs tablet of same dose), but may also reflect which variable has overall more predictive strength – further research is needed to obtain more insight in the relevance of this measure.

Code to calculate the area under the curve for each of the elements in Phi:

```
Area(DeltaT, Phi)
> $Area
>           [,1]     [,2]
> [1,] 0.7699825 0.244806
> [2,] 0.4896121 1.039269
>
> $Area_range
>           [,1]     [,2]
> [1,] 0.7699825 0.244806
> [2,] 0.4896121 1.039269
```

```
# If, for instance, the time interval range from 1 to 2 should be inspected
# (and not 0 to infinity), then use:
Area(DeltaT, Phi, t_min = 1, t_max = 2)
> $Area
>           [,1]     [,2]
> [1,] 0.7699825 0.244806
> [2,] 0.4896121 1.039269
>
> $Area_range
```

```
>           [,1]       [,2]
> [1,] 0.1480669 0.08153651
> [2,] 0.1630730 0.23775709
```

```
# Notes:
# - A fitted object of the classes "varest" and "ctsemFit" can also be used.
# - The Drift matrix can also be used.
```

Note that these should not be seen as matrices. That is, it gives the area under the curve for each element separately. For ease, these are depicted in matrices.

2.1.2 Maximum or minimum of curves in Phi-plot To calculate for each element in Phi what the optimum is, thus either its maximum or minimum, together wit the corresponding time interval, the following function can be used:

```
MaxDeltaT(DeltaT, Phi)
> $DeltaT_MinOrMaxPhi
>            [,1]       [,2]
> [1,] 22.0973792  0.7992313
> [2,]  0.7992313 23.1126937
>
> $MinOrMaxPhi
>              [,1]         [,2]
> [1,] 9.617270e-09 1.025501e-01
> [2,] 2.051002e-01 9.275289e-09
>
> $DeltaT_MinOrMaxPhi_2
>          [,1]     [,2]
> [1,] 23.11269 23.11269
> [2,] 23.11269 23.11269
>
> $MinOrMaxPhi_2
>              [,1]         [,2]
> [1,] 4.340191e-09 4.486453e-09
> [2,] 8.972906e-09 9.275289e-09
>
> $DeltaT_MinOrMaxPhi_3
>          [,1]     [,2]
> [1,] 23.11269 23.11269
> [2,] 23.11269 23.11269
>
> $MinOrMaxPhi_3
>              [,1]         [,2]
> [1,] 4.340191e-09 4.486453e-09
> [2,] 8.972906e-09 9.275289e-09
```

```
# Notes:
# - A fitted object of the classes "varest" and "ctsemFit" can also be used.
# - The Drift matrix can also be used.
```

Note that these should not be seen as matrices. That is, it gives the optimum and corresponding time interval for each element separately. For ease, these are depicted in matrices.

2.2 Transformations

2.2.1 Discrete-time <-> continuous-time There are two types of lagged relationships models: the discrete-time (DT) and continuous-time (CT) models. Both are related (for more details see some of the references at the top). The following functions transform parameter matrices from one type to the other:

```
### From DT to CT ###

CTMparam(DeltaT, Phi, SigmaVAR)
> $Drift
>            [,1]       [,2]
> [1,] -1.5275304  0.3598189
> [2,]  0.7196378 -1.1317296
>
> $Sigma
>            [,1]       [,2]
> [1,]  3.2370355 -0.9260082
> [2,] -0.9260082  2.5960426
>
> $Gamma
>           [,1]      [,2]
> [1,] 1.0855262 0.1102123
> [2,] 0.1102123 1.2170170
>
> $standDrift
>            [,1]       [,2]
> [1,] -1.5275304  0.3809887
> [2,]  0.6796507 -1.1317296
>
> $standSigma
>            [,1]       [,2]
> [1,]  2.9819968 -0.8056499
> [2,] -0.8056499  2.1331194
>
> $standGamma
>            [,1]       [,2]
> [1,] 1.00000000 0.09588739
> [2,] 0.09588739 1.00000000
>
> $UniqueSolutionDrift
> [1] TRUE
>
> $UniqueSolutionDrift_message
> [1] "The resulting drift matrix Drift is unique (since the eigenvalues of Drift (and thus also Phi) ar
>
> $StableProcess
> [1] TRUE
>
> $StableProcess_message
> [1] "The process is stable, it is restore to its equilibrium (since all (real parts of) the eigenvalue
>
> $eigenvalueDrift
> [1] -1.8756189 -0.7836412
>
> $eigenvaluePhi_DeltaT
```

```
> [1] 0.4567399 0.1532601
```

```
# Notes:
# - A fitted object of the class "varest" can also be used.
# - The Gamma matrix can be used instead of SigmaVAR.
```

```
### From CT to DT ###

DeltaT <- 1
VARparam(DeltaT, Drift, Sigma)

# Notes:
# - A fitted object of the class "ctsemFit" can also be used.
# - The Gamma matrix can be used instead of SigmaVAR.
```

2.2.2 Gamma For both the DT and CT model, it holds that there are three types of matrices: 1) the lagged relationships matrix (Phi or Drift), 2) the residuals covariance matrix (SigmaVAR or Sigma), and 3) the stationary covariance matrix (Gamma - this is the same for both models, which makes sense since it is the covariance matrix of the contemporaneous variables). These three types of matrices are related: when you know two of them, the other one can be calculated. Next, the code for the functions that can calculate Gamma from the DT and CT models, respectively:

```
# Using DT matrices
Gamma.fromVAR(Phi, SigmaVAR)
>           [,1]      [,2]
> [1,] 1.0855262 0.1102123
> [2,] 0.1102123 1.2170170
```

```
# Using CT matrices
Gamma.fromCTM(Drift, Sigma)
>           [,1]      [,2]
> [1,] 1.0855262 0.1102123
> [2,] 0.1102123 1.2170170
```

2.2.3 Checks When you have the CT matrices, you can also do checks on these matrices. For example, the covariance matrices should be positive definite. These checks can be done via:

```
ChecksCTM(Drift, Sigma)
> $ChecksAreFine
> [1] TRUE
>
> $error
> [1] "No error: All matrices (Drift, Sigma, and Gamma) are fine."
>
> $Drift
>            [,1]       [,2]
> [1,] -1.5275304  0.3598189
> [2,]  0.7196378 -1.1317296
>
> $Sigma
>            [,1]       [,2]
> [1,]  3.2370355 -0.9260082
> [2,] -0.9260082  2.5960426
>
```

```
> $Gamma
>           [,1]      [,2]
> [1,] 1.0855262 0.1102123
> [2,] 0.1102123 1.2170170
>
> $EigenVal_Drift
> [1] 1.8756189 0.7836412
>
> $EigenVal_Sigma
> [1] 3.896442 1.936636
>
> $EigenVal_Gamma
> [1] 1.279604 1.022939
```

```
# Note: a fitted object of class "ctsemFit" can also be used.
```

2.2.4 Standardization The function 'StandPhi' renders standardized lagged relationships estimates from their unstandardized counterparts. Using this function, it is also possible to obtain the multivariate (elliptical) confidence intervals (CIs) for the lagged relationships estimates. Please note that most software renders univariate confidence intervals, thus, not taking into account covariance between estimates.

The following code demonstrates how to obtain standardized lagged relationships estimates from unstandardized lagged relationships estimates.

```
## Obtain only standardized lagged relationships ##
StandPhi(N = NULL, Phi, SigmaVAR)
> $Phi_DeltaT
>      [,1] [,2]
> [1,] 0.25 0.10
> [2,] 0.20 0.36
>
> $StandPhi_DeltaT
>           [,1]      [,2]
> [1,] 0.2500000 0.1058835
> [2,] 0.1888869 0.3600000
>
> $SigmaVAR_DeltaT
>      [,1] [,2]
> [1,]    1    0
> [2,]    0    1
>
> $standSigmaVAR_DeltaT
>           [,1]      [,2]
> [1,] 0.9212122 0.0000000
> [2,] 0.0000000 0.8216812
>
> $Gamma
>           [,1]      [,2]
> [1,] 1.0855262 0.1102123
> [2,] 0.1102123 1.2170170
>
> $standGamma
>            [,1]       [,2]
> [1,] 1.00000000 0.09588739
```

```
> [2,] 0.09588739 1.00000000
```

```
# or
#StandPhi(Phi = Phi, SigmaVAR = SigmaVAR)

## Obtain standardized lagged relationships and multivariate CIs ##
# In that case, input specifying the sample size (N) is needed as well.
N <- 643
StandPhi(N, Phi, SigmaVAR)
> $Phi_DeltaT
>      [,1] [,2]
> [1,] 0.25 0.10
> [2,] 0.20 0.36
>
> $StandPhi_DeltaT
>           [,1]      [,2]
> [1,] 0.2500000 0.1058835
> [2,] 0.1888869 0.3600000
>
> $vecStandPhi_DeltaT
> [1] 0.2500000 0.1058835 0.1888869 0.3600000
>
> $CovMx_vecStandPhi_DeltaT
>               [,1]          [,2]          [,3]          [,4]
> [1,]  0.0014504849 -0.0001390832  0.0000000000  0.0000000000
> [2,] -0.0001390832  0.0014504849  0.0000000000  0.0000000000
> [3,]  0.0000000000  0.0000000000  0.0012937694 -0.0001240562
> [4,]  0.0000000000  0.0000000000 -0.0001240562  0.0012937694
>
> $multiCI_vecStandPhi_DeltaT
>        Phi11      Phi12     Phi21    Phi22
> LB 0.1631628 0.01904629 0.1068749 0.277988
> UB 0.3368372 0.19272065 0.2708989 0.442012
>
> $SigmaVAR_DeltaT
>      [,1] [,2]
> [1,]    1    0
> [2,]    0    1
>
> $standSigmaVAR_DeltaT
>           [,1]      [,2]
> [1,] 0.9212122 0.0000000
> [2,] 0.0000000 0.8216812
>
> $Gamma
>           [,1]      [,2]
> [1,] 1.0855262 0.1102123
> [2,] 0.1102123 1.2170170
>
> $standGamma
>            [,1]       [,2]
> [1,] 1.00000000 0.09588739
> [2,] 0.09588739 1.00000000
```

```
# Notes:
# - A fitted object of the classes "varest" and "ctsemFit" can also be used.
# - The Gamma matrix can also be used instead of Sigma.
```

2.2.5 Same time interval To compare results from multiple studies, it may be necessary to transform the time-interval dependent matrices (i.e., Phi and SigmaVAR) such that they all reflect the same time interval.

The function 'StandTransPhi' offers the possibility to transform time-interval dependent matrices to reflect scenarios in which another time interval was used. For example, a lagged relationships matrix where measurements were taken at some time interval (DeltaT) may be transformed to reflect the scenario in which another time interval (DeltaTStar) was used.

The following code demonstrates the application of this function.

```
DeltaTStar <- 2     # Specify desired  time interval
DeltaT <- 1         # Specify observed time interval

## Obtain only (unstandardized) transformed lagged relationships ##
StandTransPhi(DeltaTStar, DeltaT, N = NULL, Phi)
# or
#StandTransPhi(DeltaTStar, DeltaT, Phi = Phi)

## obtain only (un)standardized transformed lagged relationships ##
StandTransPhi(DeltaTStar, DeltaT, N = NULL, Phi, SigmaVAR)
# or
#StandTransPhi(DeltaTStar, DeltaT, Phi = Phi, SigmaVAR = SigmaVAR)

## Obtain (un)standardized transformed lagged relationships
##                                                    and multivariate CIs ##
# In that case, input for the sample size (N) is needed as well.
N <- 643
StandTransPhi(DeltaTStar, DeltaT, N, Phi, SigmaVAR)

# Notes:
# - A fitted object of the classes "varest" and "ctsemFit" can also be used.
# - The Gamma matrix can be used instead of Sigma.
```

Correlated residuals

Even though the focus in lagged relationships models is the strength and sign of the lagged relationships, the residuals may be of interest as well (more details may be found in Kuiper and Hamaker). This section shows the options with respect to inspecting the residual covariance matrix SigmaVAR(DeltaT).

In this section, we will make use of the following discrete-time matrices:

```
# Phi(DeltaT) & SigmaVARDeltaT)
DeltaT <- 1            # Specify time observed time interval (DeltaT)
Phi <- myPhi[1:2,1:2]  # Select part that corresponds to one primary study
SigmaVAR <- diag(2) # for ease
```

As demonstrated above, via the SigmaVAR-plot (Psi-plot), the residual covariance matrix of the discrete-time model varies with the chosen time interval, like the lagged relationships (in Phi) do. Recall, the code to produce such a SigmaVAR-plot is given by:

```
### Make Psi-plot/SigmaVAR-plot ###
```

```
# Example 1.1: unstandardized Phi&SigmaVAR #
SigmaVARPlot(DeltaT, Phi, SigmaVAR)

# Example 1.2: standardized Phi&SigmaVAR #
SigmaVARPlot(DeltaT, Phi, SigmaVAR, Stand = 1)

# Notes:
# - Like in the Phi-plot, a 'varest' object can be used.
# - Like in the Phi-plot, the plot can be customized
#   (but there is no ggplot variant)
```

3.1: Time interval for uncorrelated discrete-time residuals

Since SigmaVAR varies with the chosen time interval DeltaT, there may exist a DeltaT for which SigmaVAR is diagonal. In that case, the discrete-time residuals are uncorrelated. Note that for DeltaT = 0, SigmaVAR is diagonal, but there may be more positive time intervals for which SigmaVAR is diagonal. The code to calculate this DeltaT (if it exists) is:

```
# Calculate DeltaT for which SigmaVAR is diagonal
DiagDeltaT(Phi, SigmaVAR = SigmaVAR)
> $DeltaT_diag
> [1] 1
>
> $message
> [1] "No message / warning / error. Hence, there is positive DeltaT for which the diagonals/variances
>
> $Phi_DeltaT_diag
>      [,1] [,2]
> [1,] 0.25 0.10
> [2,] 0.20 0.36
>
> $SigmaVAR_DeltaT_diag
>      [,1] [,2]
> [1,]    1    0
> [2,]    0    1
>
> $Gamma
>           [,1]      [,2]
> [1,] 1.0855262 0.1102123
> [2,] 0.1102123 1.2170170
>
> $StandPhi_DeltaT_diag
>           [,1]      [,2]
> [1,] 0.2500000 0.1058835
> [2,] 0.1888869 0.3600000
>
> $StandSigmaVAR_DeltaT_diag
>           [,1]      [,2]
> [1,] 0.9212122 0.0000000
> [2,] 0.0000000 0.8216812
>
> $Gamma_s
>            [,1]       [,2]
> [1,] 1.00000000 0.09588739
```

```
> [2,] 0.09588739 1.00000000
>
> $errorMatrices
> [1] "No error: All matrices (Drift, Sigma, and Gamma) are fine."
>
> $message_startvalues
> [1] "In case the Psi-plot/SigmaVAR-plot does show a solution (or another solution) for DeltaT such tha
```

```
# Notes:
# - The function 'SigmaVARPlot' can help to see whether there is a DeltaT for
#   which SigmaVAr(DeltaT) is diagonal.
#   The starting value of DeltaT ('xstart_DeltaT') can be altered if needed.
# - A 'varest' object can be used as well.
```

Please note that, even though this function calculates the DeltaT for which SigmaVAR is diagonal, it may not be that useful. Kuiper and Hamaker show that correlated discrete-time residuals are supposed to be an indication for omitted common causes (or an effect at a shorter time interval), but it actually is not. It also does not signal that lagged effects relationships are distorted, since omitted unique causes will not affect the DT residual correlations.

3.2: Continuous-time residual covariance matrix

It may be better to inspect continuous-time residuals, by looking at Sigma, instead of the DT residuals. Correlated continuous-time residuals signal one or more omitted relevant variables (so, common or unique omitted causes). Namely, correlated continuous-time residuals warn us that the found lagged relationships do not reflect the causal relationships (but solely the predicting relationships). Unfortunately, it is not a measure for the extent of the distortion. So, non-zero off-diagonals in Sigma are a red flag that non-causal relationships are found. Thus, when estimating the DT model, one may want to inspect Sigma as well.

As demonstrated in an example above, code to obtain Sigma from the discrete-time parameter matrices is given by:

```
### From DT to CT ###

CTM <- CTMparam(DeltaT, Phi, SigmaVAR)
CTM$Sigma
>            [,1]       [,2]
> [1,]  3.2370355 -0.9260082
> [2,] -0.9260082  2.5960426
```

```
# Notes:
# - A fitted object of the class "varest" can also be used.
# - The Gamma matrix can be used instead of SigmaVAR.
```