Continuous Time Structural Equation Modelling With R Package ctsem

Charles C. Driver
Max Planck Institute for Human Development

Johan H. L. Oud
Radboud University Nijmegen

Manuel C. Voelkle
Max Planck Institute for Human Development

Abstract

We introduce **ctsem** (Driver, Oud, and Voelkle 2015), an R package for continuous time structural equation modelling of panel ($N > 1$) and time series ($N = 1$) data, using either full information maximum likelihood (FIML) or the Kalman filter. Most dynamic models for longitudinal data in the social and behavioural sciences are discrete time models. An assumption of discrete time models is that time intervals between measurements are equal, and that all subjects were assessed at the same intervals. Violations of this assumption are regularly ignored due to the difficulty of accounting for varying time intervals, therefore parameter estimates can be severely biased. By using stochastic differential equations and estimating an underlying continuous process, continuous time models allow for any pattern of measurement occasions. By interfacing to a general purpose SEM package (**OpenMx**), **ctsem** combines the flexible specification of structural equation models with the enhanced data gathering opportunities and improved estimation of continuous time models. **ctsem** can estimate relationships over time for multiple latent processes, measured by multiple noisy indicators with varying time intervals between observations. Within and between effects are estimated simultaneously by modelling both observed covariates and unobserved heterogeneity. Exogenous shocks with different shapes, group differences, higher order diffusion effects and oscillating processes can all be simply modelled. We first briefly introduce and define continuous time models, then show how to specify and estimate a range of continuous time models using **ctsem**.

Keywords: time series, panel data, state Space, structural equation modelling, continuous time, stochastic differential equation, dynamic models, Kalman filter, R.

1. Introduction

Dynamic models, such as the well known vector autoregressive model, are widely used in the social and behavioural sciences. They allow us to see how fluctuations in processes relate to later values of those processes, the effect of an input at a particular time, how the various factors relate to average levels of the processes, and many other possibilities. Some examples with panel data include the impact of European institutional changes on business cycles (Canova, Ciccarelli, and Ortega 2012), the coupling between sensory and intellectual functioning (Ghisletta and Lindenberger 2005), or the analysis of bidirectional links between children's delinquency and the quality of parent-child relationships (Keijsers, Loeber, Branje, and Meeus 2011). Examples of single subject approaches are studies on the decline in pneumonia rates in the USA after a vaccine introduction (Grijalva, Nuorti, Arbogast, Martin, Edwards, and Griffin 2007), or the lack of a relationship between antidepressant sales and public health in Iceland (Helgason, Tómasson, and Zoega 2004). At present, applications of dynamic models in the social and behavioural sciences are almost exclusively limited to *discrete time models*. In discrete time models it is assumed that time progresses in discrete steps, that time intervals between measurement occasions are equal, and that, in case of panel data, all subjects are assessed with the same time intervals. In many cases, these assumptions are not met, resulting in biased parameter estimates. This concept is illustrated in Figure 1. In the upper panel, Figure 1 shows a true autoregressive effect of .80 between observed variables (represented by squares), assuming equal intervals of length $\Delta t = 1$ (represented by equal distances between observed variables), while the lower panel shows a process with two intervals of $\Delta t = 1$ and one interval $\Delta t = 2$. In the top panel, the meaning of the estimate of .80 is clear – it refers to the autoregression estimate for 1 unit of time. In the lower case, however, the autoregression estimate of .73 is ambiguous – it is too low to characterise the relation between the first three occasions (correct value of .80 is in brackets) and too high between the last two occasions (correct value of .64).

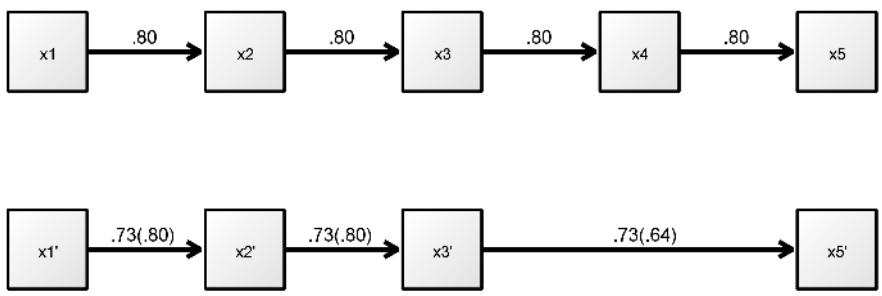

Figure 1: Two autoregressive processes, each exhibiting a true autoregressive effect of .80 for 1 unit of time. The top process is measured with equal time intervals (represented by the space between observations) of 1 unit, while the lower process has unequal intervals.

Obviously, parameter estimates and, thus, scientific conclusions, are biased when observation intervals vary and this is not adequately accounted for. In simple cases, such as the example in Figure 1, additional variables – so called phantom variables (Rindskopf 1984), with missing values for all individuals – could be added in order to artificially create equally spaced time intervals. For example, an additional variable could be specified at t4, resulting in equal time intervals and permitting the use of standard discrete time models. For complex patterns of individually varying time intervals, however, this approach quickly becomes untenable (Voelkle and Oud 2013). Furthermore, based on discrete time models it is difficult to compare results obtained from different studies with unequal time intervals, which poses a limitation to the production of cumulative knowledge in science (Voelkle, Oud, Davidov, and Schmidt 2012).

Continuous time models overcome these problems, offering researchers the possibility to estimate parameters free from bias due to unequal intervals, easily compare between studies and datasets with different observation schedules, and gather data with variable time intervals between observations. Although continuous time models have a long history (Coleman 1964; Hannan and Tuma 1979), their use in the social sciences is still uncommon. At least in parts, this is due to a lack of suitable software to specify and estimate continuous time models. With the introduction of **ctsem** in this article, we want to overcome this limitation. Although we will define continuous time models in the next section and provide several examples in the sections thereafter, a comprehensive treatment of continuous time models is beyond the scope of this article. For a more general introduction to continuous time models by means of SEM, the reader is referred to Voelkle *et al.* (2012). For additional information on the technical details we refer the reader to Oud and Jansen (2000).

The R package **ctsem** interfaces to **OpenMx** (Boker *et al.* 2011), a powerful general purpose SEM package for R (R Core Team 2014). This allows **ctsem** to capitalize on all the possibilities of structural equation models, including manifest and latent variables, flexibility in imposing parameter constraints, multiple group functionality, as well as the use of different fitting functions and highly non-linear constraints. Most importantly, by interfacing to **OpenMx** the user may tailor standard continuous time models to his or her specific needs.

The remainder of this article is organised as follows: in Section 2, we provide a formal definition of continuous time models. In Section 3 we will show how to install **ctsem** and give an overview of the package. In Section 4, we will review different data structures and discuss the role of time in continuous time models. In Section 5, we will show how to specify continuous time models in **ctsem**, followed by a discussion of model estimation and testing in Section 6. In Section 7 we will discuss various extensions of basic continuous time models, including unobserved heterogeneity, time dependent and time independent exogenous predictors, time series, multiple group models, additional models of the diffusion process, and the analysis of oscillations. We will end with some discussion of various specification options and tips for model fitting in Section 8, and point to current limitations and future research and development directions in Section 9.

2. Continuous time models: fundamentals

The class of continuous time models implemented in **ctsem** is represented by the multivariate stochastic differential equation:

$$\frac{\mathrm{d}\boldsymbol{\eta}(t)}{\mathrm{d}t} = \mathbf{A}\boldsymbol{\eta}(t) + \mathbf{B}\mathbf{Z} + \mathbf{M}\mathbf{X}(t) + \boldsymbol{\xi} + \mathbf{G}\frac{\mathrm{d}\mathbf{W}(t)}{\mathrm{d}t} \tag{1}$$

Vector $\boldsymbol{\eta}(t) \in \mathbb{R}^{v\times 1}$ is a v-variable vector of the processes of interest at time t. The $v \times v$ matrix $\mathbf{A}$ represents the so-called drift matrix, with auto effects on the diagonal and cross effects on the off-diagonals characterising the temporal relationships of the processes.

$\mathbf{B}$ is a $v \times p$ matrix with parameters describing the effect of the vector of time independent predictors $\mathbf{Z} \in \mathbb{R}^{p\times 1}$ on $\boldsymbol{\eta}(t)$.

$\mathbf{M}$ is a $v \times l$ matrix with parameters describing the effect of time dependent predictors $\mathbf{X}(t) \in \mathbb{R}^{l\times 1}$ on $\boldsymbol{\eta}(t)$.

The base level of the processes is captured by the v-length vector of random variables $\boldsymbol{\xi} \sim \mathrm{N}(\boldsymbol{\kappa}, \boldsymbol{\phi}_{\boldsymbol{\xi}})$, where the value $\boldsymbol{\kappa}$ denotes the continuous time intercept, and value $\boldsymbol{\phi}_{\boldsymbol{\xi}}$ the variance of this parameter across units of analysis (e.g., individuals). The continuous time intercepts set the long-term level of the processes – without it the processes of a stable model would always trend towards zero in the long-run. The variance term may be set to 0, resulting in a fixed rather than random intercept model.

$\mathbf{W}(t)$ represents the diffusion process, specifically here the Wiener process, a random-walk in continuous time. This is multiplied by $\mathbf{G}$ which determines the variance and covariance. $\mathbf{Q}$, where $\mathbf{Q} = \mathbf{G}\mathbf{G}^\top$, represents the variance-covariance matrix of the diffusion process in continuous time.

The solution of the stochastic differential Equation 1 for any time interval $t - t_0$ is:

$$\begin{aligned}\boldsymbol{\eta}(t) = \mathrm{e}^{\mathbf{A}(t-t_0)}\boldsymbol{\eta}(t_0) +& \\ &\mathbf{A}^{-1}[\mathrm{e}^{\mathbf{A}(t-t_0)} - \mathbf{I}]\mathbf{B}\mathbf{Z} + \\ &\int_{t_0}^{t} \mathrm{e}^{\mathbf{A}(t-s)}\mathbf{M}\mathbf{X}(s)\mathrm{d}s + \\ &\mathbf{A}^{-1}[\mathrm{e}^{\mathbf{A}(t-t_0)} - \mathbf{I}]\boldsymbol{\xi} + \\ &\int_{t_0}^{t} \mathrm{e}^{\mathbf{A}(t-s)}\mathbf{G}\mathrm{d}\mathbf{W}(s)\end{aligned} \tag{2}$$

The five terms of this equation correspond to the five terms of Equation 1, and give the link between the continuous model and discrete instantiations of the process. The last term, the integral of the diffusion over the given time interval, exhibits covariance matrix:

$$\mathrm{cov}\left[\int_{t_0}^{t} \mathrm{e}^{\mathbf{A}(t-s)}\mathbf{G}\mathrm{d}\mathbf{W}(s)\right] = \int_{t_0}^{\top} \mathrm{e}^{\mathbf{A}(t-s)}\mathbf{Q}\mathrm{e}^{\mathbf{A}^\top(t-s)}\mathrm{d}s = \mathrm{irow}\left\{\mathbf{A}_{\#}^{-1}[\mathrm{e}^{\mathbf{A}_{\#}(t-t_0)} - \mathbf{I}]\,\mathrm{row}\mathbf{Q}\right\} \tag{3}$$

Where $\mathbf{A}_{\#} = \mathbf{A} \otimes \mathbf{I} + \mathbf{I} \otimes \mathbf{A}$, with $\otimes$ denoting the Kronecker-product, row is an operation that takes elements of a matrix rowwise and puts them in a column vector, and irow is the inverse of the row operation.

As will be discussed in Section 7.2, we distinguish between *impulse* and *level change* time dependent predictors. Both are different solutions for the integral of the third term in Equation 2. An impulse is an effect on a single infinitesimal moment with integrated effect 1. A level change predictor takes different values across time and has constant effect M. See Section 7.2.2 for an illustration and further details. For an impulse predictor at time u, the third term becomes:

$$\mathrm{e}^{\mathbf{A}(t-u)}\mathbf{M}\mathbf{X}(u) \tag{4}$$

For a level change predictor that changes at time u, the third term would be handled by applying Equation 2 in two steps, first from initial time t_0 to u, then from time u to time t of interest. In the first step from $\boldsymbol{\eta}(t_0)$ to $\boldsymbol{\eta}(u)$, $\mathbf{A}^{-1}[\mathrm{e}^{\mathbf{A}(u-t_0)} - \mathbf{I}]\mathbf{M}\mathbf{X}(t_0)$ is used as the third term in Equation 2. In the second step from $\boldsymbol{\eta}(u)$ to $\boldsymbol{\eta}(t)$, $\mathbf{A}^{-1}[\mathrm{e}^{\mathbf{A}(t-u)} - \mathbf{I}]\mathbf{M}\mathbf{X}(u)$ is used as the third term in Equation 2.

The process vector $\boldsymbol{\eta}(t)$ may be directly observed or latent with measurement model

$$\mathbf{Y}(t) = \boldsymbol{\Gamma} + \boldsymbol{\Lambda}\boldsymbol{\eta}(t) + \boldsymbol{\delta}, \quad \text{where } \boldsymbol{\delta} = \mathrm{N}(\mathbf{0}, \boldsymbol{\Theta}), \quad \text{and } \boldsymbol{\Gamma} = \mathrm{N}(\boldsymbol{\tau}, \boldsymbol{\epsilon}) \tag{5}$$

where c-length vector $\boldsymbol{\tau}$ is the expected value of manifest intercept $\boldsymbol{\Gamma}$, which has $c \times c$ covariance matrix $\boldsymbol{\epsilon}$ (referred to later as *manifest traits* – see Section 7.1). $\boldsymbol{\Lambda}$ is a $c \times v$ matrix of factor loadings, $\mathbf{Y}(t) \in \mathbb{R}^{c \times 1}$ is a c-variable vector of manifest variables, and $\boldsymbol{\Theta}$ is a $c \times c$ covariance matrix of residual vector $\boldsymbol{\delta} \in \mathbb{R}^{c \times 1}$.

2.1. Continuous time and SEM

Continuous time models have already been implemented as structural equation models, using either non-linear algebraic constraints (Oud and Jansen 2000) or linear approximations of the matrix exponential (Oud 2002). Our formulation uses either the SEM RAM (reticular action model) specification as per McArdle and McDonald (1984), or the state space form recently added to **OpenMx**(Neale *et al.* 2015; Hunter 2014). For details on the equivalence and differences between SEM and state space modelling techniques, see Chow, Ho, Hamaker, and Dolan (2010). Expectation matrices are generated for each individual according to the specified inputs, constraints, and observed timing data. Optimization using either full information maximum likelihood or the Kalman filter within **OpenMx** is then used to estimate the parameters. For more detailed information on the specification of continuous time structural equation models, the reader is referred to Oud and Jansen (2000); Arnold (1974); Singer (1998); Voelkle *et al.* (2012). Note that while earlier incarnations of continuous time modelling focused on approaches to implement the matrix exponential, **OpenMx** now includes a form of the exponential recommended in computational contexts, the scaling and squaring approach with Pade approximation (Higham 2009), which has been implemented in **ctsem** accordingly.

3. ctsem package overview and installation

3.1. Package overview

Estimating continuous time models via **ctsem** comprises five steps: First, **ctsem** must be correctly configured, which we detail in Section 3.2. Second, the data must be adequately prepared (Section 4). Next, the continuous time model must be specified by creating a **ctsem** model object using the function `ctModel` (Sections 5 and 7). After specification, the model must be fit to the data using the function `ctFit`, after which `summary` and `plot` methods may be used to examine parameter estimates, standard errors, and fit statistics (Section 6). We will discuss these steps in the following.

3.2. Package installation

As **ctsem** is an R package, it requires R to be installed, available from `www.r-project.org` (R Core Team 2014). To install **ctsem** within R, run the following line of code within R:

```
R> source(file = 'https://bitbucket.org/charles_driver/ctsem_public/raw/
+   master/installctsem.R')
```

In the event of difficulties with the above script, **ctsem** can be installed via the following steps: Install version 2.0 or greater of the package **OpenMx** (Neale *et al.* 2015), available from web address: `http://openmx.psyc.virginia.edu/`.

Download, then install within R, the package of **ctsem** from: http://bitbucket.org/charles_driver/ctsem_public/src/master/ctsem_current.tar.gz

Then load the **ctsem** library via `library('ctsem')`.

4. Data structure

The internal functions of `ctFit` use data in a wide layout, with all data for each individual in a single row, including the time intervals between measurement occasions for this individual. Because this is the format used internally when fitting, for the sake of transparency it is also required as the input format, and is detailed below in Section 4.1. In some cases it may however be simpler to maintain data in a long format, and use the `ctLongToWide` and `ctIntervalise` functions we provide to convert from long format with absolute times to wide format with time intervals. This functionality is discussed in Section 4.2. The choice of time scale and treatment of the initial time point can influence results and will be discussed in Section 4.3, though first time users may find it easier to return to later.

4.1. Wide format

This is the data format required when fitting a model with **ctsem**. The example data below depicts two individuals, observed at three occasions, on three manifest variables, one time dependent predictor, and two time independent predictors. A corresponding path diagram of one possible model for this data is shown in Figure 2. The data are ordered into blocks as follows: Manifest process variables, time dependent predictors, time intervals, time independent predictors. Manifest variables are grouped by *measurement occasion* and ordered within this by *variable*. In the example there are three manifest variables (Y1, Y2, Y3) assessed across three measurement occasions. In this case, the first three columns of the data (Y1_T0, Y2_T0, Y3_T0) represent the three manifest variables at the first measurement occasion, time point 0, followed by the columns of the second measurement occasion and so on. Note that measurement occasions subsequent to the first may occur at different times for different individuals. Also note the naming convention, wherein the variable name is followed by an underscore and T, followed by an integer denoting the measurement occasion. After the manifest variables, any time dependent predictors (there need not be any) are grouped by *variable* and ordered within this by *measurement occasion*. This change of ordering compared to the dependent variables reflects the fact that relations between exogenous predictors are not of interest. If they are, then they should instead be included as additional latent processes. Note also that in continuous time modelling a cause must always precede an effect in time, precluding instantaneous effects. For this reason, no time dependent predictors may be included at the last measurement occasion, because there would be no time for an effect to occur. In the data below and the model in Figure 2, there is only one time dependent predictor, TD1, though a second could be added by inserting its' two columns directly after TD1_T1). After the time dependent predictors, T-1 time intervals are specified in chronological order, with column names dT followed by the number of the measurement occasion occurring *after* the interval. That is, dT1 refers to the time interval between the first measurement occasion, T0, and the second, T1. In continuous time modelling it is imperative to know the time point at which an observation takes place. Thus, missing values on time intervals are not allowed. Finally, two time independent predictors (TI1, TI2 – the naming here is only with variable names) are contained in the last two columns of the data structure.

```
  Y1_T0 Y2_T0 Y3_T0 Y1_T1 Y2_T1 Y3_T1 Y1_T2 Y2_T2 Y3_T2 TD1_T0 TD1_T1 dT1 dT2
1   2.5    NA   2.2   2.3   5.1   2.3   4.8   3.9   3.7      0    6.0   4   9
2   5.6   3.7   5.0    NA   7.0   4.4   2.5   4.6   4.7      0    5.3   1  14
   TI1  TI2
1 -0.6 -0.6
2 -0.5 -0.2
```

4.2. Conversion from long format with absolute times

Although **ctsem** uses the wide format as default data input, often data are stored in long format, that is, each subject has multiple rows of data, with each row reflecting a particular measurement occasion. In addition, time intervals may not be readily available at the individual level, instead the *absolute time* when a measurement took place is recorded. To convert from long format, the data must contain a subject identification column, columns for every observed variable, and a time variable. In the example below, three manifest variables of interest (Y1, Y2, Y3) have been observed across a number of occasions, along with one time dependent predictor (TD1) and two time independent predictors (TI1, TI2). The variable 'Time' contains the time when the measurement took place (e.g., in weeks from the beginning of the study).

```
     subject   Y1   Y2   Y3      TD1 Time    TI1    TI2
[1,]       1 3.13   NA 4.59  0.32625    0     NA -0.609
[2,]       1 2.30 5.11 2.31 -0.00972    1 -0.607 -0.609
[3,]       1 3.85 4.45   NA       NA   NA     NA     NA
[4,]       1 3.05 3.02 3.11 -0.26912    2 -0.607 -0.609
[5,]       2 5.26 6.06 5.55  0.07493    3 -0.529 -0.233
[6,]       2 5.68 5.64 7.96 -0.63566    4 -0.529 -0.233
[7,]       2 4.68 5.43 4.79  0.26527    5 -0.529 -0.233
```

Given the specific wide structure required by **ctsem**, and that the time points of measurement may vary across individuals, restructuring from long to wide can be complicated, so we have included functions to manage this. First, the long format data with information on the absolute time of measurement must be converted to the wide format, using the `ctLongToWide` function. Then, subject specific time intervals based on the absolute time information must be generated, using the function `ctIntervalise`. One should take care that the defaults used by `ctIntervalise` for structuring the data and handling missing time information are appropriate.[1]

```
R> data('longexample')
R> wideexample <- ctLongToWide(datalong = longexample, id = "subject",
```

[1] By default, when timing information is missing, variables measured at that time are also set to NA for the individual missing the information. Once this is done the actual time of measurement no longer influences parameter estimates or likelihood, so we can set it to an arbitrary minimum interval. By default, the `mininterval` argument to `ctIntervalise` is set to .001. This argument must be set *lower* than the minimum time interval recorded in the data, so that later observations can be adjusted without problems.

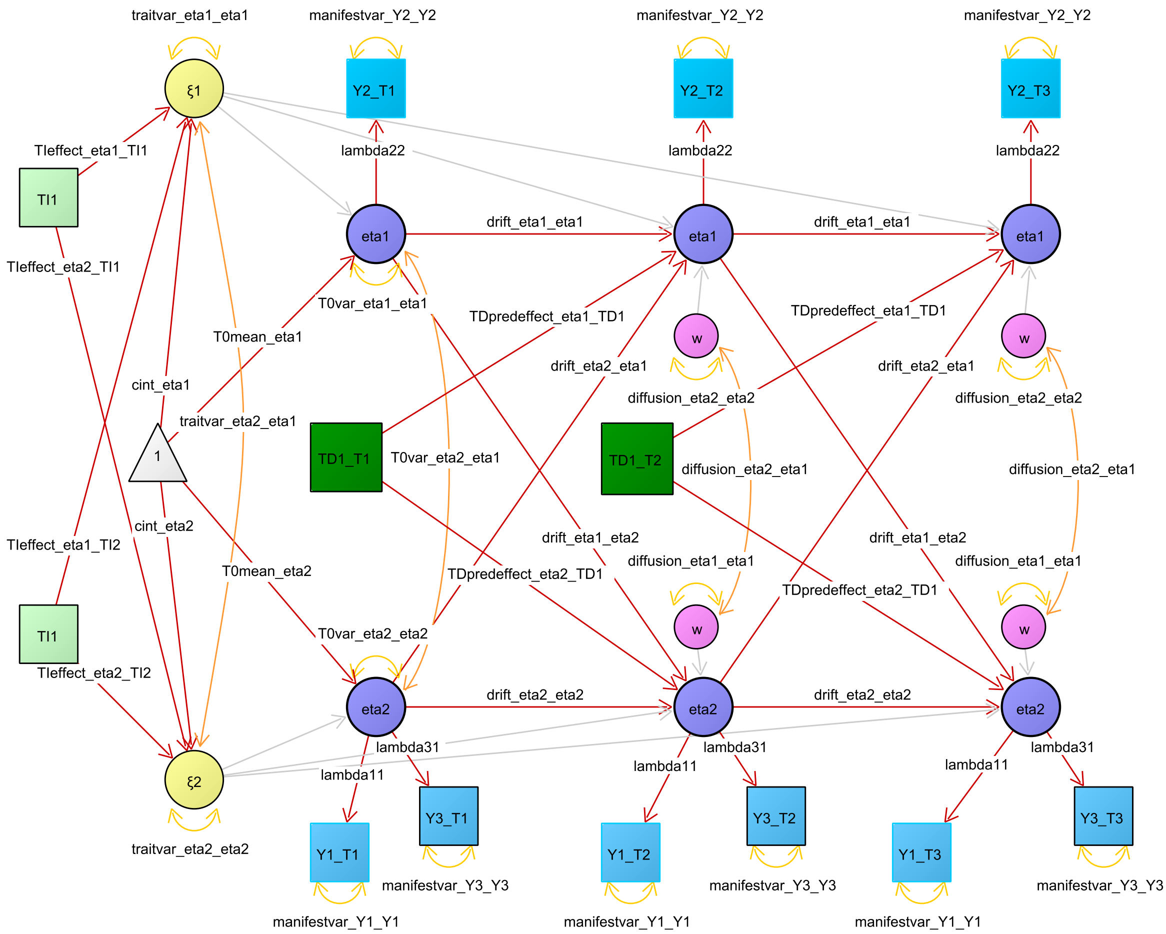

Figure 2: The first three time points of a two process continuous time model, with three manifest indicators (blue) measuring 2 latent processes (purple), one time dependent predictor(dark green), and two time independent predictors (light green). Variance / covariance paths are in orange, regressions in red. Light grey paths indicate those that are constrained to a function of other parameters. Note that the value of parameters for all paths to latents at time 2 and higher do not directly represent the effect, rather, the effect depends on a function of the shown parameter and the time interval Δt. Covariances between the time dependent predictor and traits (yellow) are not drawn.

```
+   time = "Time", manifestNames = c("Y1", "Y2", "Y3"),
+   TDpredNames = "TD1", TIpredNames = c("TI1", "TI2"))
R> wide <- ctIntervalise(datawide = wideexample, Tpoints = 4, n.manifest = 3,
+   n.TDpred = 1, n.TIpred = 2, manifestNames = c("Y1", "Y2", "Y3"),
+   TDpredNames = "TD1", TIpredNames = c("TI1", "TI2") )
```

4.3. Choice of initial time point and time scale

Choice of initial time point: Pre-determined or stationary?

An important aspect of continuous time modelling is the choice of how to handle the initial time point. In principle, there are two different ways to do so. One approach is to treat the first time point as *predetermined*, where no assumptions are made about the process prior to the initial time point. In this case, all parameters for the first time point (means, variances, effect of predictors and unit level heterogeneity) are freely estimated. This is the default in **ctsem**, though requires some constraining if fitting a single individual.[2] When treating the first time point as predetermined, it is important to choose a meaningful starting point, as the process will gradually transition from the variances and means of the initial parameters, towards those of the parameters when the model is stationary. In principle, the initial time point does not have to reflect the first measurement occasion, and can also be set to any time prior. For example it may be meaningful to set T0 to the beginning of the school year, although the first measurement was only taken two weeks after start of school. This can be specified using the `startoffset` argument to `ctIntervalise`, specifying the amount of time prior to the first observed measurement occasion. The other approach is to assume a stationary model, that is, a model where the first observations are merely random instantiations of a long term process with time-invariant mean and variance expectations. Or, put another way, we assume that sufficient time has elapsed from the unobserved, hypothetical start of the process to our first measurement occasion, such that whatever the start values were, they no longer influence the process. Strictly speaking, this requires an infinite length of time, or a process that began in a stationary state. However, in some practical cases without clear trends in the data it is possible that the improvement in estimation due to the stationarity assumption outweighs related losses (this may also be tested). To implement the stationarity assumption the means and variances of the first measurement occasion are constrained according to the model predicted means and variances across all time points. This is specified by including a character vector of the T0 matrices to constrain in the `ctFit` arguments: `stationary = c('T0VAR', 'T0MEANS', 'T0TRAITEFFECT', 'T0TIPREDEFFECT')` constrains all means and variances to stationarity. These matrices do not all have to be included however, one could allow differing means at T0 but constrained variances, or vice versa. The `ctModel` specification of any matrices that are constrained to stationarity is ignored.

Choice of time scale: Individual or sample relative time?

An additional consideration when treating the first time point as predetermined is necessary in cases of individually varying time intervals. Here, two alternatives need to be distinguished. The default option is to treat the observation times as *relative to the individual*, the other is

[2] Either T0VAR or T0MEANS must be fixed, see Section 7.3.

to treat them as *relative to the sample*. When we treat time as relative to the individual, the first observation of every individual is set to measurement occasion T0, even though different individuals may have been recorded many years apart. However if we treat time as relative to the sample, every individual's observation times are set relative to the very first observation in the entire sample. This may result in a larger and sparser data matrix, potentially with only a single observation at the first measurement occasion. To specify sample relative time when converting from absolute time to intervals, set the argument `individualRelativeTime = FALSE` in the `ctIntervalise` function. The choice between the individual or sample relative time may influence parameter estimates when the processes are not stationary. One way of deciding between the two may be to observe whether the changes of the individuals' processes is more closely aligned with the sample relative or individual relative time. The change in processes may be more aligned with individual relative time when we expect that the activity of measurement relates to changes in the process. Consider for instance the relation between abstinence behaviour and mood among individuals attending an alcohol addiction clinic. Different individuals may come and go from the clinic over many years, but the mean level of abstinence is likely related to when each individual began attending the clinic and being measured – not the specific date the observation took place. In contrast, sample relative time could be more appropriate for a study of linguistic abilities in a cohort of schoolchildren over the years, with some individuals observed early and some only observed later, once they are older and more developed. In this case, we may expect changes in the average linguistic ability related to sample time. Another example that becomes conveniently available with continuous time models and these functions is to arrange the data in individual relative fashion but using age as the timing variable. In this case, age-related developmental trajectories may be studied. When considering these options one should be aware that consistent up or down trends over time may confound dynamic parameter estimates, if the innovation (latent residual) at t is correlated with the process at $t-1$. Pre-processing approaches that remove trend components, such as controlling for age or year, removing a linear trend, or differencing scores, *may* provide some check on model estimates, but the ramifications of these should be carefully considered. Alternatively one may wish to explicitly model the diffusion process, discussed in Section 7.5.

5. Model specification

Continuous time models are specified via the `ctModel` function. This function takes as input a series of arguments and parameter matrices, and outputs a list object containing matrices to be later evaluated by the ctFit function. The `ctModel` function contains many defaults that should be generally applicable and safe, in that most parameters are specified to be freely estimated, with a few exceptions.[3] However, as with all default settings, they should be checked as they may not be applicable. The arguments to the `ctModel` function and the relation to equations in Section 2 are shown in Table 5 (required specification) and Table 5 (optional specification). The matrices can be specified with either character labels, to indicate free parameter names, or numeric values, which indicate fixed values. A mixture of both in

[3] `ctModel` defaults that *may* not be considered safe, as they are not freely estimated, are the `MANIFESTMEANS`, `TRAITVAR` and `MANIFESTTRAITVAR` matrices. These were set in such a way to ensure that the defaults are appropriate also for single indicator and $N = 1$ cases, and because generally only one of the two trait matrices can be set at once. See Section 7.3 regarding manifest means, and Section 7.1 regarding the trait matrices.

one matrix is fine. These generally need to be set when constraining parameters to equality (same character label), when fixing certain parameters to specific values (for instance, when you do not wish to have a certain parameter in the model, or when testing if an effect is different from 0), or when assigning non-standard names to output parameters. An example

Argument	Sign	Meaning
n.manifest	c	Number of manifest indicators per individual at each measurement occasion.
n.latent	v	Number of latent processes.
Tpoints		Number of time points, or measurement occasions, in the data.
LAMBDA	$\mathbf{\Lambda}$	n.manifest $\times$ n.latent loading matrix relating latent to manifest variables.

Table 1: Optional arguments for `ctModel`.

model specification relying heavily on the defaults is:

```
R> examplemodel <- ctModel(n.latent = 2, n.manifest = 2, Tpoints = 3,
+   LAMBDA = diag(2))
```

A visual representation of this model is shown in Figure 3. With `n.latent = 2`, we have specified a model with 2 latent processes, shown in purple. Each of these is measured by a single manifest indicator (in blue), for a total of 2 manifest variables, specified with `n.manifest = 2`. Loadings between latents and manifests are fixed to 1.00 (indicated by the 2×2 diagonal LAMBDA matrix) at 3 measurement occasions, specified by `Tpoints = 3`. Because no other parameters are specified, the model defaults are used, resulting in a bivariate latent process model where each manifest variable has a measurement error variance (manifestvar_Y1_Y1, manifestvar_Y2_Y2), and a mean fixed to 0. The initial latent variables of each process each have a freely estimated mean (T0mean_eta1, T0mean_eta2), variance (T0var_eta1_eta1, T0var_eta2_eta2), and covariance (T0var_eta2_eta1). Subsequent latent variables of each process all have an innovation term, with the variance dependent on a function of diffusion matrix (variances diffusion_eta1_eta1, diffusion_eta2_eta2, covariance diffusion_eta2_eta1), drift matrix, and time interval Δt. Each latent variable in our two processes has continuous auto effects on itself according to the drift_eta1_eta1 and drift_eta2_eta2 parameters (the diagonals of the drift matrix), and cross effects to the other process according to the drift_eta1_eta2 and drift_eta2_eta1 parameters (the off diagonals). This drift matrix combines with time interval Δt to generate the auto and cross regressions shown in the diagram. As usual, the first process listed in the parameter name represents the row of the drift matrix, and the second the column, with the direction of effects flowing from column to row – so the parameter drift_eta1_eta2 represents the effect of a change in process 2 on later values of process 1. Each process also has a continuous intercept (cint_eta1, cint_eta2), which, in combination with the drift matrix, sets the level to which each process asymptotes.

6. Model estimation

The `ctFit` function estimates the specified model, calling the data in wide format along with the **ctsem** model object. For an example, we can fit a similar model to that defined in Section 5. We first load an example dataset contained in the **ctsem** package, then use the `ctFit` function for parameter estimation. Output information can be obtained via the `summary` function. The dataset used in this example, is a simulation of the relation between leisure time and happiness

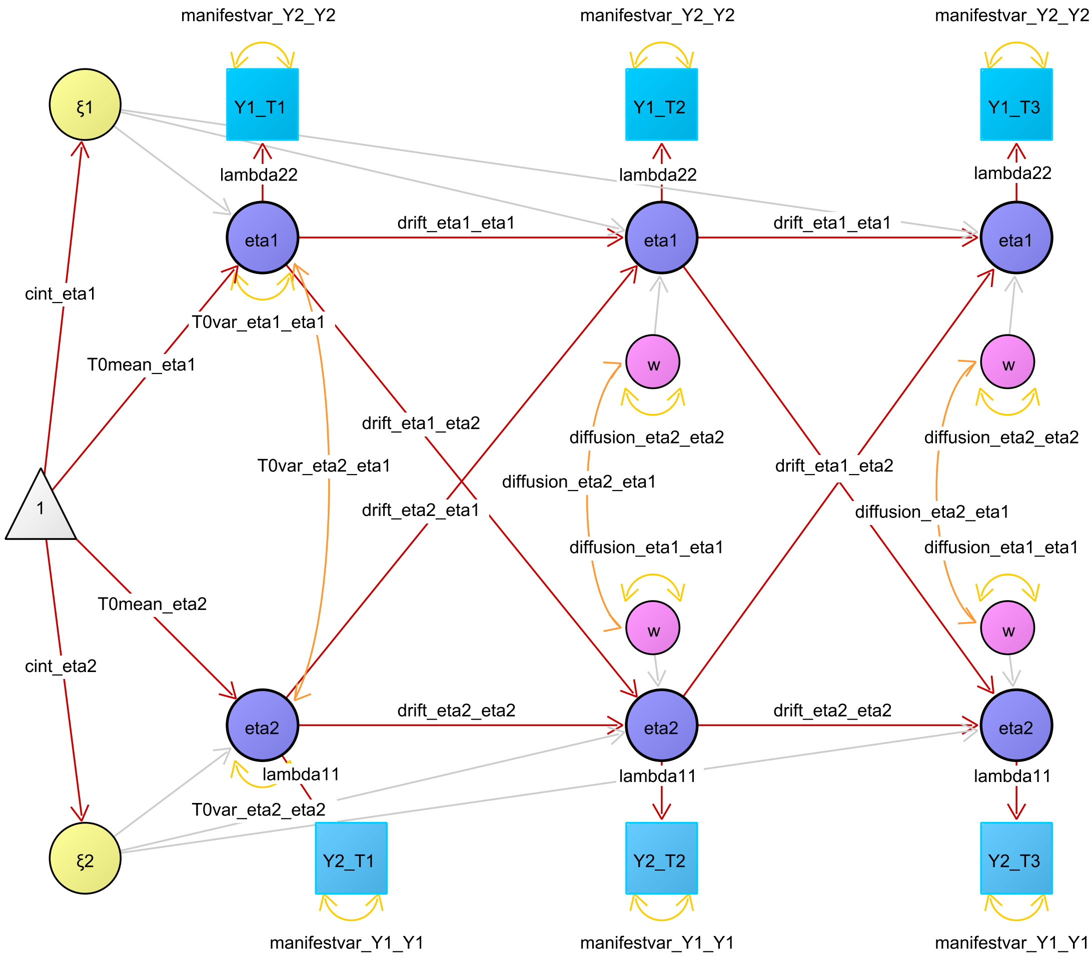

Figure 3: A two process continuous time model with manifest indicators (blue) measuring latent processes (purple). Variance / covariance paths are in orange, regressions in red. Light grey paths indicate those that are either fixed to certain values or constrained to other parameters. Note that the value of the parameters for all paths to latents at time 2 and higher do not directly represent the effect, rather, the effect depends on a function of the shown parameter and the time interval Δt.

Argument	Sign	Default	Meaning.
manifestNames		Y1, Y2, etc	n.manifest length character vector of manifest names.
latentNames		eta1, eta2, etc	n.latent length character vector of latent names.
T0VAR		free	symmetric n.latent × n.latent matrix of latent process initial variance / covariance.
T0MEANS		free	n.latent × 1 matrix of latent process means at first time point, T0.
MANIFESTMEANS	$\boldsymbol{\tau}$	0	n.manifest × 1 matrix of manifest means.
MANIFESTVAR	$\boldsymbol{\Theta}$	free diag	symmetric n.manifest × n.manifest matrix of variance / covariance between manifests (i.e., measurement error).
DRIFT	$\mathbf{A}$	free	n.latent × n.latent matrix of continuous auto and cross effects.
CINT	$\boldsymbol{\kappa}$	free	n.latent × 1 matrix of continuous intercepts.
DIFFUSION	$\mathbf{Q}$	free	symmetric n.latent × n.latent matrix of diffusion process variance / covariance.
TRAITVAR	$\boldsymbol{\phi}_{\xi}$	NULL	NULL if no trait variance, or symmetric n.latent × n.latent matrix of trait variance / covariance.
T0TRAITEFFECT		constrained	n.latent × n.latent matrix of effect of traits on latents at T0. 'constrained' sets to a fixed diagonal matrix, 'auto' freely estimates.
MANIFESTTRAITVAR	$\boldsymbol{\epsilon}$	NULL	NULL if no trait variance on manifest indicators, or symmetric n.manifest × n.manifest variance / covariance matrix.
n.TDpred	l	0	Number of time dependent predictors in the dataset.
TDpredNames		TD1, TD2, etc	n.TDpred length character vector of time dependent predictor names.
TDPREDMEANS		free	n.TDpred × (Tpoints-1) rows × 1 column matrix of time dependent predictor means.
TDPREDEFFECT	$\mathbf{M}$	free	n.latent × n.TDpred matrix of effects from time dependent predictors to latent processes.
T0TDPREDCOV		free	n.latent × ((Tpoints-1) × n.TDpred) covariance matrix between latents at time 1 and time dependent predictors.
TDPREDVAR		free	symmetric (n.TDpred × (Tpoints-1)) × (n.TDpred × (Tpoints-1)) variance / covariance matrix for time dependent predictors.
TRAITTDPREDCOV		free	n.latent rows × (n.TDpred × (Tpoints-1)) columns covariance matrix for latent traits and time dependent predictors.
TDTIPREDCOV		free	(n.TDpred × (Tpoints-1)) rows × n.TIpred columns covariance matrix between time dependent and independent predictors.
n.TIpred	p	0	Number of time independent predictors.
TIpredNames		TI1, TI2, etc	n.TIpred length character vector of time independent predictor names.
TIPREDMEANS		free	n.TIpred × 1 matrix of time independent predictor means.
TIPREDEFFECT	$\mathbf{B}$	free	n.latent × n.TIpred effect matrix of time independent predictors on latent processes.
T0TIPREDEFFECT		constrained	n.latent × n.TIpred effect matrix of time independent predictors on latents at T0. "constrained" replicates the TIPREDEFFECT matrix, forcing the T0 effect to be the same as the long run effect, 'auto' freely estimates.
TIPREDVAR		free	symmetric n.TIpred × n.TIpred variance / covariance matrix for time independent predictors.
inits		NULL	a 2 column matrix, first column with character strings of parameter labels used when specifying other matrices, second column with starting values for these parameters.

Table 2: Required arguments for `ctModel`.

for 100 individuals across 6 measurement occasions. Because our data here does not use the default manifest variable names of `Y1` and `Y2`, but rather `LeisureTime` and `Happiness`, we must include a `manifestNames` character vector in our model specification. Because each manifest directly measures a latent process, we can use the same character vector for the `latentNames` argument, though one could specify any character vector of length 2 here, or rely on the defaults of `eta1` and `eta2`.

```
R> data('ctExample1')
R> example1model <- ctModel(n.latent = 2, n.manifest = 2, Tpoints = 6,
+   manifestNames = c('LeisureTime', 'Happiness'),
+   latentNames = c('LeisureTime', 'Happiness'), LAMBDA = diag(2))
R> example1fit <- ctFit(datawide = ctExample1, ctmodelobj = example1model)
R> summary(example1fit)
```

The output of `summary` after fitting such a model includes a range of matrices representing the continuous time parameters (e.g., DRIFT), discrete time transformations of these parameters for the time interval $\Delta t = 1$ (e.g., discreteDRIFT), and when appropriate, asymptotic values for the parameters as the time interval Δt approaches ∞ (e.g., asymDIFFUSION). Matrices representing the model implied variance at T0 and as Δt approaches ∞ are also included, as T0TOTALVAR and asymTOTALVAR respectively. When appropriate, standardised matrices are output with the suffix 'std'.[4] The `$omxsummary` portion of the summary output contains information directly from the **OpenMx** summary function, including estimated parameters and fit information such as the -2LL, AIC, and BIC. See the **OpenMx** user guide (Boker *et al.* 2014) for further details.

```
$discreteDRIFTstd
            LeisureTime Happiness
LeisureTime      0.9698   -0.0438
Happiness        0.0131    0.8440
```

The output above shows the standardised discrete time equivalent of the DRIFT matrix for time interval $\Delta t = 1$ as reported by `summary`. This is provided for convenience, but one should note that it only represents the temporal effects given the specific interval of 1 unit of time. The unstandardised discreteDRIFT matrix may be calculated from the continuous drift matrix for any desired interval, with the code `omxExponential(summary(example1fit)$DRIFT * desiredinterval)`, see Equation 2. From the diagonals of the matrix we see that changes in the amount of leisure time one has tend to persist longer (indicated by a stronger autoregression) than happiness. The cross-regression in row 2 column 1 suggests that as leisure time increases, this tends to be followed by *increases* in happiness. However, the cross-regression in row 1 column 2 suggests that as happiness increases, this tends to be followed by *reductions* in leisure time. While these results are accurate for the specified model, they do not represent the underlying model for this data, which we explain more of in Section 7.1 on unobserved heterogeneity.

6.1. Comparing different models

[4]Standardisations are based on only the relevant variance, not the total. For instance, DRIFT parameters are standardised using only the within-subject variance, asymDIFFUSION, because DRIFT parameters are typically intended to represent individual, or average individual, temporal dynamics.

Suppose we wanted to test the model we fit above against a model where the effect of happiness on later leisure time (parameter drift_LeisureTime_Happiness) was constrained to 0. First we specify and fit the model under the null hypothesis by taking our previous model and fixing the desired parameter to 0:

```
R> testmodel <- example1model
R> testmodel$DRIFT[1, 2] <- 0
R> testfit <- ctFit(datawide = ctExample1, ctmodelobj = testmodel)
```

The result may then be compared to the original model with a likelihood ratio test, using the **OpenMx** function `mxCompare`. To use this function a base model fit object and a comparison model fit object must be specified, with the latter being a constrained version of the former. Note that **ctsem** stores the original **OpenMx** fit object under a `$mxobj` sub-object, which must be referenced when using **OpenMx** functions directly.

```
R> library('OpenMx')
R> mxCompare(example1fit$mxobj, testfit$mxobj)

   base comparison ep minus2LL   df AIC diffLL diffdf        p
1 ctsem       <NA> 16     2905 1184 537     NA     NA       NA
2 ctsem      ctsem 15     2920 1185 550   14.8      1 0.000117
```

According to the conventional $p < .05$ criterion, results show that the more constrained model fits the data significantly worse, that is, happiness has a significant effect on later leisure time for this model and data. An alternative to this approach is to estimate likelihood based 95% confidence intervals for our parameters of interest:

```
R> example1cifit <- ctFit(datawide = ctExample1, ctmodelobj = example1model,
+   confidenceintervals = 'DRIFT')

                              lbound estimate  ubound
drift_LeisureTime_LeisureTime -0.0529 -0.03037 -0.0111
drift_LeisureTime_Happiness   -0.1222 -0.07825 -0.0385
drift_Happiness_LeisureTime   -0.0148  0.00894  0.0352
drift_Happiness_Happiness     -0.2559 -0.16922 -0.1071
```

Now the `summary` function reports 95% confidence bounds for the continuous drift parameters, which in case of drift_LeisureTime_Happiness does not include 0. For complicated models, the estimation of confidence intervals may increase computation time considerably. Note that although the standard errors of parameter estimates may be automatically returned via **OpenMx** and displayed via `summary`, likelihood based confidence intervals are the recommended approach because confidence intervals are unlikely to be symmetric around the point estimate for many parameters in a continuous time model.

6.2. Plots

A visual depiction of the relationships between the processes over time can be obtained by using the `plot` function on any fit object created by `ctFit`. This will show the latent processes' mean trajectories, within-subject variance, stable between-subject variance (when applicable), autoregression, and cross regression plots. Autoregression plots show the impact of a 1 unit change in a process on later values of that process, while cross regression plots show the impact of a 1 unit change in one process on later values of other processes. Plots of the discrete time autoregressive and cross regression coefficients for the above fitted model are shown in the top row of Figure 4.

7. Continuous time models: extensions

7.1. Unobserved heterogeneity

When modelling panel data, the continuous intercept parameter $\boldsymbol{\kappa}$ reflects the expected value for continuous time intercept $\boldsymbol{\xi}$, which, along with the initial means, determines the average level and mean trajectory of a process. In panel data, however, it is common that *individuals* exhibit stable differences in the level. Within **ctsem** we call such stable differences *traits*, but they may also be thought of more abstractly as *unit level* or *between person* differences, or *unobserved heterogeneity*. When individuals exhibit this trait variance, fitting a model that fails to account for it will result in parameter estimates that will not reflect the processes of individual subjects, but will mix between and within-person information (Balestra and Nerlove 1966; Oud and Jansen 2000; Halaby 2004). To account for this bias, individual differences can be incorporated in two different ways. One way is to control for observed covariates as will be discussed in Section 7.2.1. As covariates are likely to be insufficient, one may also estimate the latent trait variance by estimating the variance and covariance $\boldsymbol{\phi}_{\xi}$ of the intercept parameters $\boldsymbol{\xi}$ across individuals.[5] In **ctsem**, freely estimated latent trait variance may be added with the argument `TRAITVAR = "auto"` to the `ctModel` command. If the user is interested in a specific variance-covariance structure, it is of course also possible to specify the n.latent $\times$ n.latent matrix of free or fixed parameters by hand. To illustrate the inclusion of trait variance, we fit the same model on simulated leisure time and happiness introduced above, but also model the trait variance.

```
R> data('ctExample1')
R> traitmodel <- ctModel(n.manifest = 2, n.latent = 2, Tpoints = 6,
+   LAMBDA = diag(2), manifestNames = c('LeisureTime', 'Happiness'),
+   latentNames = c('LeisureTime', 'Happiness'), TRAITVAR = "auto")
R> traitfit <- ctFit(datawide = ctExample1, ctmodelobj = traitmodel)
```

From Figure 4, we can see that after accounting for differences in the base levels of leisure time and happiness, the estimated auto and cross regression effects between latent processes are very different. Auto effects (persistence) have reduced, and the magnitude and sign of the cross effects have switched. Now, rather than a *decrease in leisure time* predicting an *increase*

[5] Note that this is a substantially different approach to achieve unbiased effect estimates than the common *fixed effects* approach (see for example Mundlak 1978), as our SEM specification, while essentially a *random effects* model which have typically been associated with bias for within effects, allows unbiased estimation of *within* and *between* effects at the same time. For further details on the estimation of unobserved heterogeneity in an SEM context, see Bollen and Brand (2010), and in the continuous time case Voelkle, Driver, and Oud (2015).

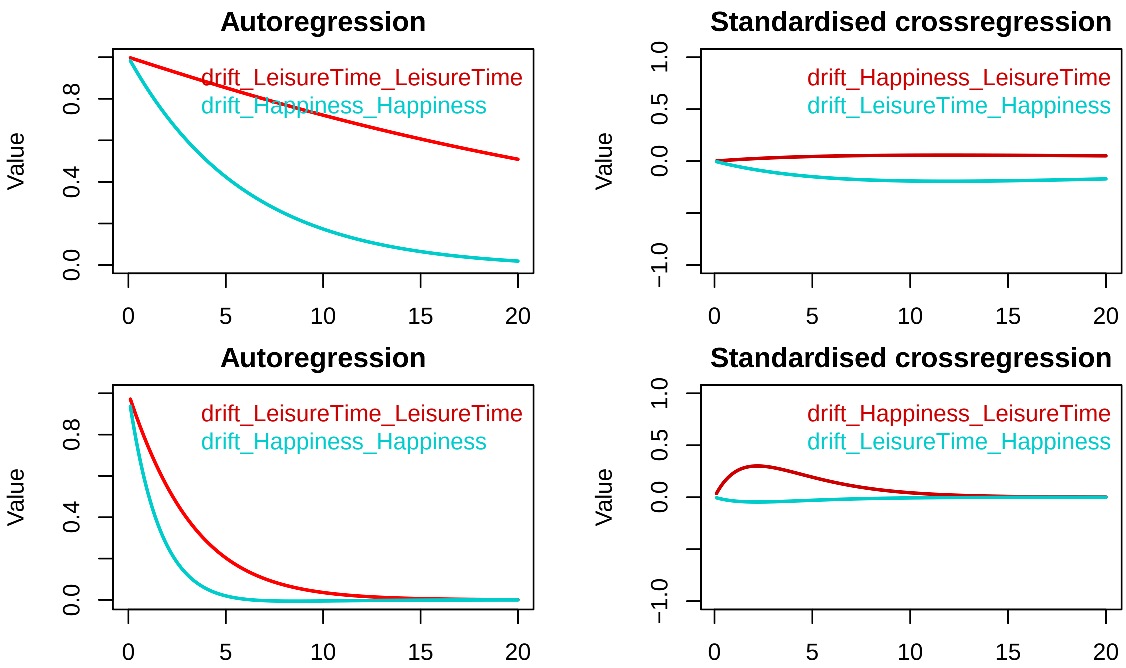

Figure 4: Top row shows parameter plots without accounting for trait variance, bottom row with trait variance accounted for.

in happiness, after controlling for unobserved heterogeneity we see instead that *increases in leisure time* predict later *increases in happiness*.

Traits at the indicator level

Beyond differences in the level of the latent process, it is also possible that stable individual differences in the level of some or all indicators of a process may exist, and as such may be better accounted for at the measurement level. Take for instance a latent process, happiness, estimated using three survey questions at 10 time points for multiple individuals. The means for question one are fixed to 0 to identify the model, while those for questions two and three are 2.50 and 1.20 respectively. According to the models we have described so far, the estimated (or fixed) manifest means apply equally to all individuals, and deviations from this are taken to represent genuine information about the latent process. However, consider that question three queries happiness with work, which may for some people be consistently high, independent of their actual latent happiness, and for some may be consistently low. Calculating the latent process using the same mean for happiness with work again confounds between and within person information, but we can account for this by using what we will refer to as *manifest traits* – an additional, time invariant variance-covariance structure on the measurement level. These are specified by including the `MANIFESTTRAITVAR` matrix in the `ctModel` specification, either as `MANIFESTTRAITVAR = "auto"` wherein time invariant variance and covariance for all indicators is freely estimated, or the n.manifest × n.manifest matrix can be specified explicitly as usual. Such a specification may allow for improved fit of factor models, more realistic estimates of the dynamics of individual processes, and the testing of measurement related hypotheses. Note however that because process level traits are likely to be difficult to identify in a model that also contains manifest level traits, it is important to decide at which level to account for unobserved heterogeneity, or how to constrain and identify the model.

7.2. Predictors

ctsem allows the inclusion of *time independent* as well as time *time dependent* predictors. Time independent predictors could be variables such as gender, personality or socio-demographic background variables that remain constant over time. An example of a time dependent predictor could be a financial crisis, which all individuals in the sample experience at the same time, or the death of a loved one, which only some individuals may experience and for whom the time point of the event may differ. Both events may be thought of as adding some relatively distinct and sudden change to an individual's life, which influences the processes of interest. Time dependent predictors are distinguished from the endogenous latent processes in that they are assumed to be independent of fluctuations in the processes – changes in the latent processes do not lead to changes in the predictor. Furthermore, no temporal structure between different time points is modelled. Because of these two assumptions, in any case where the time dependent predictor depends on earlier values of either itself or the latent process, it may be better to model it as an additional latent process.

Time independent predictors

Time independent predictors are added by including the data as per the structures shown in Section 4, and specifying the number of time independent predictors, `n.TIpred`, in the `ctmodel` arguments. If not using the default variable naming, a `TIpredNames` character vector should also be specified. For an example, we add the 'number of close friends' as a time independent predictor to the earlier leisure time and happiness model. Note that, just like in any conventional regression analysis, if time independent predictors are not centered around 0, the estimate of continuous intercept parameters depends on the mean of the predictor.

```
R> data('ctExample1TIpred')
R> tipredmodel <- ctModel(n.manifest = 2, n.latent = 2, n.TIpred = 1,
+   manifestNames = c('LeisureTime', 'Happiness'),
+   latentNames = c('LeisureTime', 'Happiness'),
+   TIpredNames = 'NumFriends',
+  Tpoints = 6, LAMBDA = diag(2), TRAITVAR = "auto")
R> tipredfit <- ctFit(datawide = ctExample1TIpred, ctmodelobj = tipredmodel)
```

```
$TIPREDEFFECT                          $asymTIPREDEFFECT
            NumFriends                             NumFriends
LeisureTime     0.0124                 LeisureTime   -0.00176
Happiness       0.1372                 Happiness      0.21819

$discreteTIPREDEFFECT                  $addedTIPREDVAR
            NumFriends                             LeisureTime Happiness
LeisureTime    0.00773                 LeisureTime  0.00000337 -0.000418
Happiness      0.10343                 Happiness   -0.00041791  0.051885
```

The matrices output from `summary(tipredfit)` will now include matrices related to time independent predictors, while the estimated parameters now also includes a range of variance, covariance, and effect parameters for time independent predictors. The parameters `TIpred_LeisureTime_NumFriends` and `TIpred_Happiness_NumFriends` reflect the continuous time effects of the number of close friends (TI1) on the processes of leisure time and happiness, however these effects may be more easily understood by referring to the matrix output. Matrix TIPREDEFFECT simply displays the continuous time parameters, however discreteTIPREDEFFECT shows the effect added to the processes for each unit of time, which may provide a useful comparison with discrete models. asymTIPREDEFFECT (Asymptotic time independent predictor effect) shows the expected increase in process means given an increase of 1 on the time independent predictor. From these matrices we see that while the number of close friends is negatively related to levels of leisure time, it is positively related to base levels of happiness. The final matrix, addedTIPREDVAR, displays the stable between-subject variance and covariance in the processes accounted for by the time independent predictors.

Time dependent predictors

ctsem allows the specification of two forms of time dependent predictors: The first one causes a sudden *impulse* to the system and then dissipates according to the drift matrix; the second causes a stable change in the *level* of the process. Figure 5 provides an example of the two different types of effects. A single time dependent predictor can be incorporated in a

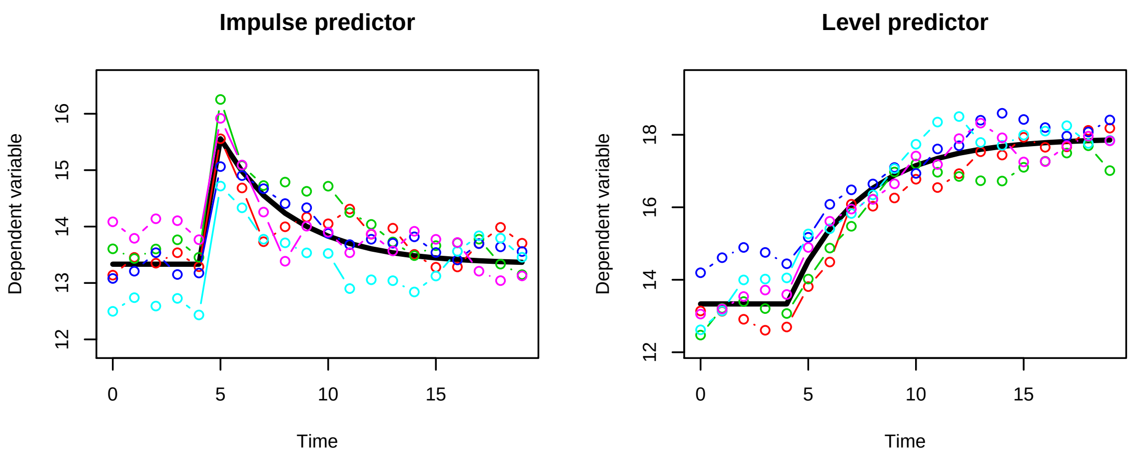

Figure 5: Two types of time dependent predictors: both plots show 5 selected individuals data, all experiencing a time dependent predictor at time point 5. The model-based expected trajectory of the predictor effect (including autoregression) is also shown as a solid black line. On the left, the processes spike up and then dissipate, reflecting a transient change, or *impulse*. On the right, the processes trend upwards towards a new equilibrium, reflecting a stable change in the *level*.

ctsem model by adding the argument `n.TDpred = 1` to the `ctModel` function, as well as a `TDpredNames` vector if not using the default variable naming in your data, then fitting as usual. In the following example, we use the same two simulated processes as above and include an intervention that all individuals experience at time 5. For example, let us assume everyone receives a large amount of money and we are interested in the impact of this monetary gift on leisure time and happiness. We expect that some short term increase in both leisure time and happiness may occur, as people may take holidays or enjoy the unexpected boon otherwise, but we also want to check whether the gift we provide may also cause a longer term adjustment in leisure time or happiness. To this end we first fit a model with the short-term impulse effect, coded in the data as a 1 when the intervention occurs and a 0 otherwise.[6]

```
R> data('ctExample2')
R> tdpredmodel <- ctModel(n.manifest = 2, n.latent = 2, n.TDpred = 1,
+   Tpoints = 8, manifestNames = c('LeisureTime', 'Happiness'),
+   TDpredNames = 'MoneyInt', latentNames = c('LeisureTime', 'Happiness'),
+   LAMBDA = diag(2), TRAITVAR = "auto")
R> tdpredfit <- ctFit(datawide = ctExample2, ctmodelobj = tdpredmodel,
+   TDpredtype = 'impulse')
```

```
$TDPREDEFFECT              $discreteTDPREDEFFECT     $asymTDPREDEFFECT
            MoneyInt                    MoneyInt                    MoneyInt
LeisureTime    0.623       LeisureTime     0.395      LeisureTime        0
Happiness      0.430       Happiness       0.317      Happiness          0
```

The matrices reported from `summary(tdpredfit)` will now include those related to the time dependent predictor, and the parameters section will include all the additional free parameters estimated, including many variance and covariance parameters, and the effect parameters TDpred_LeisureTime_MoneyInt and TDpred_Happiness_MoneyInt. Looking at the various TDPREDEFFECT matrices, TDPREDEFFECT shows us the initial impact, discreteTDPREDEFFECT shows the impact remaining after 1 unit of time, and asymTDPREDEFFECT shows the effect remaining as time continues to infinity. Of course, in the case of an impulse predictor in a stable model, the asymptotic effect is always 0. From the matrices, we can see that the monetary intervention relates directly to subsequent increases in leisure time, with also an additional direct effect on happiness. Standardised estimates are not provided because we assume no model for the variance of time dependent predictors. To test the longer term changes introduced via the monetary intervention, we can instead estimate the time dependent predictor as a level type predictor by specifying `TDpredtype = 'level'` with `ctFit`. We also need to dummy code the time dependent predictor somewhat differently, specifying a 1 for every measurement occasion after the intervention, rather than the single occasion of the intervention.

```
R> data('ctExample2level')
R> tdpredmodel <- ctModel(n.manifest = 2, n.latent = 2, n.TDpred = 1,
+   Tpoints = 8, manifestNames = c('LeisureTime', 'Happiness'),
+   TDpredNames = 'MoneyInt', latentNames = c('LeisureTime', 'Happiness'),
+   LAMBDA = diag(2), TRAITVAR = "auto")
R> tdpredfit <- ctFit(datawide = ctExample2level, ctmodelobj = tdpredmodel,
+   confidenceintervals = 'TDPREDEFFECT', TDpredtype = 'level')
```

[6] While this form of dummy coding works well, **ctsem** adds a small amount of noise to any predictors that have 0 variance at any measurement occasions to facilitate estimation. Due to individually varying time intervals, parameter estimation in this example may take some time.

```
$TDPREDEFFECT              $discreteTDPREDEFFECT     $asymTDPREDEFFECT
             MoneyInt                   MoneyInt                  MoneyInt
LeisureTime -0.060545      LeisureTime  -0.0588      LeisureTime    -1.085
Happiness   -0.000598      Happiness    -0.0046      Happiness      -0.394

$CI
                                lbound  estimate  ubound note
TDeffect_LeisureTime_MoneyInt -0.102 -0.060545 -0.0204
TDeffect_Happiness_MoneyInt   -0.113 -0.000598  0.1090
```

From `summary(tdpredfit)`, our confidence intervals (`$CI`) suggest small or zero level change effects of our monetary intervention. In contrast to the impulse predictor form, with a level predictor the asymptotic matrix is likely the most interesting, with the asymTDPREDEFFECT matrix showing the estimated long term effect of a 1 unit change in the predictor. Presently, impulse and level predictors may not be combined, but this possibility will be included shortly.

7.3. $N = 1$ time series with multiple indicators

In the examples so far, we have dealt with multiple individuals with relatively few measurement occasions, and latent processes have been estimated by a single indicator. However, **ctsem** may also be used for the analysis of time series data for single subjects observed at many measurement occasions, as well as the estimation of latent factors estimated from multiple indicators. With single-subject data, a Kalman filter implementation is typically far quicker than the RAM approach we use for multiple subjects, however **ctsem** allows either to be used. To illustrate these features, we perform a dynamic factor analysis on a single individual, with three manifest indicators measured at 50 occasions. Because the model is fitted to a single individual, we cannot freely estimate both the latent variance and mean at the first measurement occasion, but we must fix the 1×1 `T0VAR` matrix to a reasonable value (or implement stationarity constraints as discussed in Section 4.3). The precise fixed value becomes irrelevant as the time series length increases (Durbin and Koopman 2012). Note that in this example the LAMBDA matrix specifies a loading of 1.00 for manifest Y1, while loadings for Y2 and Y3 are freely estimated. Similarly, the mean for Y1 is fixed to 0, with the others free (by default these would be fixed to 0, but this may be too restrictive for a factor model). These constraints serve to identify the measurement model without further constraining it. Note also that although **ctsem** uses the Kalman filter by default when a single subject is specified, this can be overridden by specifying the `objective = "mxRAM"` argument to `ctFit`, if one wishes to use the RAM implementation.

```
R> data('ctExample3')
R> model <- ctModel(n.latent = 1, n.manifest = 3, Tpoints = 100,
+   LAMBDA = matrix(c(1, 'lambda2', 'lambda3'), nrow = 3, ncol = 1),
+   MANIFESTMEANS = matrix(c(0, 'manifestmean2', 'manifestmean3'), nrow = 3,
+     ncol = 1), T0VAR = diag(1))
R> fit <- ctFit(data = ctExample3, ctmodelobj = model)
```

7.4. Multiple group continuous time models

In some cases, certain groups or individuals may exhibit different model parameters. We can investigate group or individual level differences by specifying a multiple group model using the `ctMultigroupFit` function. For this example, we will use the same model structure as in the single subject example from Section 7.3, but apply it to two groups of 10 individuals, whom we expect to exhibit differences in the loading of the third manifest variable. When using `ctMultigroupFit`, all parameters are free across groups by default. However, in addition to the standard model specification you may also specify either a *fixed model*, or a *free model*. A fixed model should be of the same structure as the base model, with any parameters you wish to constrain across groups set to the character string 'groupfixed'. The value for any other parameters is not important. Alternatively, one may specify a free model, where any parameters to freely estimate for each group are given the label 'groupfree', and all others will be constrained across groups. In this example, because we only want to examine group differences on one parameter, we specify a free model in which the loading parameter between manifest3 and our latent process eta1 is labelled 'groupfree' – this estimates distinct lambda3 parameters for each group, and constrains all other parameters across the two groups to equality. The group specific parameter estimates will appear in the resulting summary prefixed by the specified *grouping vector*. This is the final requirement for `ctMultigroupFit` and is simply a vector specifying a group label for each row of our data. In this case we have groups one and two, containing the first and the last 10 rows of data respectively, prefixed by the letter 'g' to denote group.

```
R> data('ctExample4')
R>
R> basemodel <- ctModel(n.latent = 1, n.manifest = 3, Tpoints = 20,
+   LAMBDA = matrix(c(1, 'lambda2', 'lambda3'), nrow = 3, ncol = 1),
+   MANIFESTMEANS = matrix(c(0, 'manifestmean2', 'manifestmean3'),
+     nrow = 3, ncol = 1))
R>
R> freemodel <- basemodel
R> freemodel$LAMBDA[3, 1] <- 'groupfree'
R> groups <- paste0('g', rep(1:2, each = 10), '_')
R>
R> multif <- ctMultigroupFit(datawide = ctExample4, groupings = groups,
+   ctmodelobj = basemodel, freemodel = freemodel)
```

```
g1_lambda3 g2_lambda3
     1.382      0.194
```

Looking at the estimated parameters from `summary`, we indeed see a difference between parameters g1_lambda3 (group 1) and g2_lambda3 (group 2), and could test this with the usual approaches discussed in Section 6.1. A point to note is that the multiple group and Kalman filter implementations can be easily combined by specifying a distinct group for each row of data. This can allow for an interesting mixture of individual and group level parameters.

7.5. Dynamics on the diffusion process

In the models discussed so far, the latent error term was assumed to be independent over time. However, what about a situation where we have frequently measured variables which show very slow patterns of change, upwards or downwards trajectories that are maintained over many observations? In exactly the same way as the expected value of the *process* depends on prior values, in such a situation the expected value of the *innovation* would also be predictable based on prior values, rather than always 0. The difference between a standard autoregressive process and an autoregressive process with autoregression on the diffusion process is depicted in Figure 6. Such an autoregressive diffusion is theoretically plausible, as changes to the level

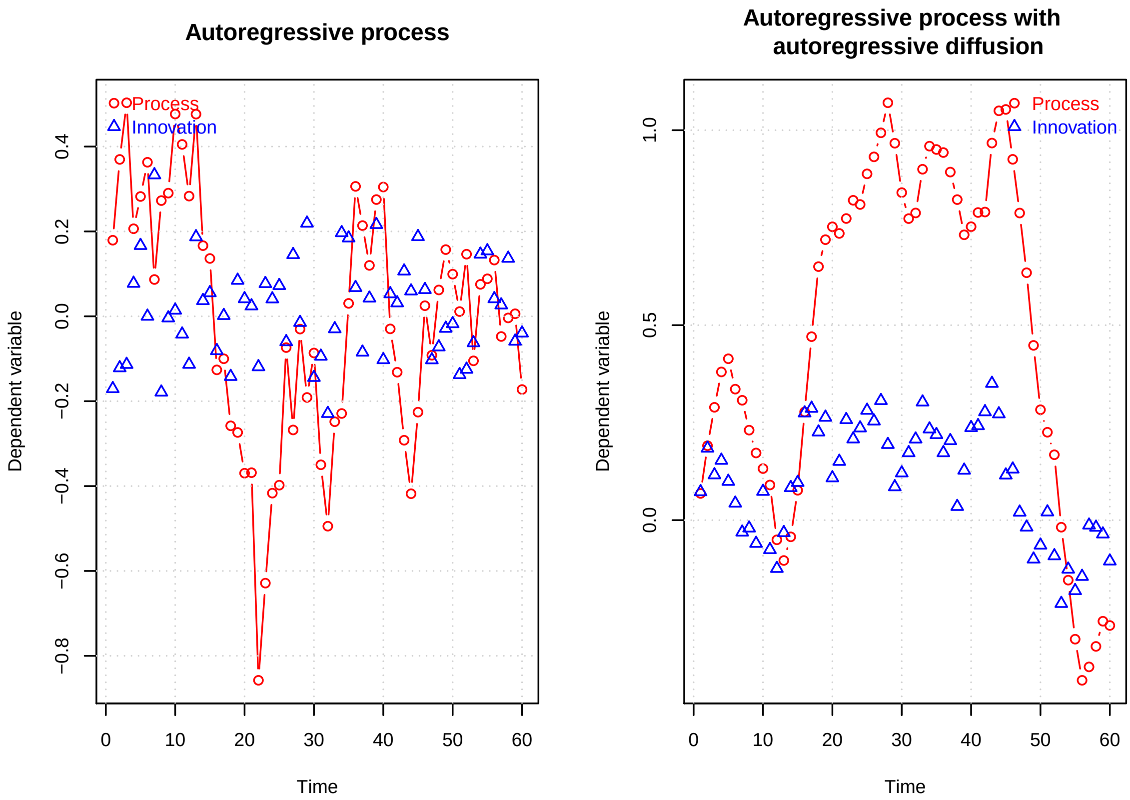

Figure 6: Difference between a process with only autoregression on the process itself (left) and process with an autoregression on the diffusion also (right) – note the smooth trends in the right process in comparison to the random direction changes in the left process, as well as some persistence in innovation values generated by the diffusion process (rescaled for ease of visual comparison).

of a process are not necessarily always random in direction, but may depend on contextual circumstances that have some persistence. Consider an individual's overall health over the course of 20 years, sampled every few months. If the individual changes exercise or eating habits, changes in health do not manifest instantly, rather we could expect either a slow increase or slow reduction, depending on whether the change of habits was positive or negative. Thus, for many measurements, the change in health from the previous measurement will likely be in the same direction as the change was one step earlier. Data on exercise or eating habits may be able to account for this, but we may not have such data. Failing to account for consistency in the diffusion will create misspecification (biased parameter estimates and confidence intervals) in the model. This is similar to the situation with ARMA modelling when the moving average process is not included or misspecified (see Granger and Newbold (1974) for the general ARMA case, and Hamaker, Dolan, and Molenaar (2003) for a treatment using SEM), as the autoregression on the diffusion here achieves a similar result – past innovations influence future innovations. With **ctsem**, this autoregressive diffusion can be included (and hence, tested) by endogenising the diffusion process – treating it as a new latent process without any manifest indicators. To do this for a single process, we specify the model as before, but with an additional latent process and some additional constraints (For more details on this form of specification, see Oud and Singer 2008; Delsing and Oud 2008).

```
R> testm <- ctModel(Tpoints = 10, n.latent = 2, n.manifest = 1,
+   LAMBDA = matrix(c(1, 0), nrow = 1, ncol = 2),
+   DIFFUSION = matrix(c(0, 0, 0, "diffusion"), 2),
+   DRIFT = matrix(c("processAR", 1, 0, "diffusionAR"), nrow = 2),
+   CINT = matrix(c("processCINT", 0), nrow = 2),
+   TRAITVAR = matrix(c("processTrait", 0, 0, 0), nrow = 2, ncol = 2))
```

In the above, we specify with LAMBDA that a single manifest variable measures only the first latent process. With DIFFUSION we specify that only the 2nd process, our unobserved diffusion process, experiences standard random innovations. With DRIFT, we specify that both latent processes have a freely estimated autoregression term, and that the diffusion process directly impacts the main process with a 1:1 relationship. This link means that the total of the 2nd process, with its random innovations and autoregression, goes into the first process. With CINT we specify that only the main process has a freely estimated continuous intercept, by fixing the 2nd parameter to 0 we specify that the *conditional expectation* of the diffusion process over long time windows is 0. With TRAITVAR we specify that there is only unobserved heterogeneity in the level of the main process, and that the average (over long time windows) conditional expectation of the innovation should be 0 for all individuals. These specified restrictions may be varied or relaxed further to develop yet more complex models. To avoid overfitting and spurious parameter estimates, careful thought as to the theoretical meaning of any extra parameters, as well as information criteria (AIC and BIC) and other model fit comparisons, should be applied.

Oscillating processes

Similar to the above autoregressive diffusion example, to model an oscillating process we also include an unmeasured latent process to represent the latent error process. Here however, we also free the drift parameter that allows the level of the main process to influence the expected value of the latent error. This allows for an oscillating model, in which the oscillations are not fixed in time, but rather depend on values of the processes. The following code loads the data and fits the oscillating model from Voelkle and Oud (2013), in which more specific details regarding oscillating models in continuous time may be found. In this case, we specify good starting values with the `inits` argument, and thus set `carefulFit = FALSE` (because we do not wish to generate start values by first estimating a form of the model, see Section 8). The solution is found without specifying start values, but may require more `retryattempts` specified with `ctFit`.

```
R> data('Oscillating')
R>
```

```
R> oscillatingm <- ctModel(n.latent = 2, n.manifest = 1, Tpoints = 11,
+   MANIFESTVAR = matrix(c(0), nrow = 1, ncol = 1),
+   LAMBDA = matrix(c(1, 0), nrow = 1, ncol = 2),
+   DRIFT = matrix(c(0, "crosseffect", 1, "autoeffect"), nrow = 2, ncol = 2),
+   CINT = matrix(0, ncol = 1, nrow = 2, ),
+   DIFFUSION = matrix(c(0, 0, 0, "diffusion"), nrow = 2, ncol = 2),
+   inits = matrix(c("crosseffect", -38, "autoeffect", -.5, "diffusion", 1,
+     "T0var11", 1, "T0var22", 38, "m2", .9), byrow = T, ncol = 2))
R>
R> oscillatingf <- ctFit(Oscillating, oscillatingm, carefulFit = FALSE)
```

8. Additional specification options and tips for model estimation

Given the complexity of parameter constraints, model estimation is sometimes more difficult than for standard structural equation models. By default, the fit functions of **ctsem** optimize using a two-step procedure, in which the first iteration penalises the likelihood[7] so as to maintain potentially problematic parameters close to 0, and then uses these estimates as starting values for the maximum likelihood estimation. This feature can be disabled by specifying `carefulFit = FALSE` to the fit function, though we have found this approach generally helps for obtaining global rather than local optimums in models with simple dynamics. In this section we list some further options to pursue in cases of estimation difficulty. As a general guideline we suggest starting with simpler, more constrained models and freeing parameters in a stepwise fashion. This could involve developing the measurement model separately, fixing off-diagonals on the DRIFT matrix to 0 (so only autoregressive parameters are included at first), or fixing the factor loading matrix prior to free estimation. If progression to additional complexity results in worse likelihood values, the optimization has resulted in a local rather than global optimum. In such a case, ensure that sensible starting values are included in the `inits` argument of `ctModel`. These may be included based on prior fits by referring to the earlier fit object in the `inits = ctGetInits(priorctfitobject)`, where `priorctfitobject` is the name of the original fit object. The following code is an example of how this would be applied to our first fitting example, from Section 6.

```
R> newInits <- ctGetInits(example1fit)
R>
R> modelWithInits <- ctModel(n.latent = 2, n.manifest = 2, Tpoints = 6,
+   manifestNames = c('LeisureTime', 'Happiness'),
+   latentNames = c('LeisureTime', 'Happiness'),
+   LAMBDA = diag(2), inits = newInits)
R>
R> fitWithInits <- ctFit(data = ctExample1, ctmodelobj = modelWithInits)
```

If stepwise model building with inits based on simpler fits still fails to produce an improved solution, some of the following suggestions may be helpful. Centering the grand mean of the variables to 0 can help, as can standardising the variances, particularly in cases where

[7] The sum of squares of each parameter that is neither a loading nor mean related is multipled by 1 thousandth of the likelihood, then added to the likelihood

both a measurement model and dynamic model are estimated. Centering panel data at the individual level is not recommended, as this amounts to a crude form of estimation – better to estimate the unobserved heterogeneity directly. Parameters may be constrained within sensible, interpretable values with the `reasonable = TRUE` to the `ctFit` function. By sensible, we mean positive variances, and negative entries on the diagonals of the DRIFT matrix. These are imposed via the upper and lower bounds functions of **OpenMx** (Boker *et al.* 2011). In some cases this can improve both the speed of fitting and the estimate obtained, but in other cases fit is worsened, and in yet others the constrained parameters are simply estimated at the imposed limit, rather than finding new, more appropriate values. The constraints can also pose a problem for estimation of maximum likelihood confidence intervals – one could get around this if needed by estimating a model with the constraints and using the returned parameters as starting values for an unconstrained estimation. In models with oscillatory or unstable dynamics these constraints are *not* sensible. See Savalei and Kolenikov (2008) for some discussion on SEM estimation with constraints. One way to search for an improved solution is simply to try many times with varying starting values. This is automated by default using the `mxTryHard` function from **OpenMx**, however you may want to increase `retryattempts` with `ctFit`, or simply re-run `ctFit` many times, as it generates any unspecified starting values with some randomness. However, since both automated procedures begin within a similar range, for truly problematic cases one may consider adding more extensive randomness to the starting values manually. In situations with a limited number of time points, you may implement the stationarity assumption, so that parameters related to the first time point are no longer estimated, but constrained to the asymptotic effects, when the time interval $\Delta t \to \infty$. This can make optimization more straightforward, and may serve as a useful basis for determining starting values, or as a viable model in itself. For more discussion regarding stationarity conditions see Section 4.3. When time intervals vary for every individual, optimization can be quite slow. To quickly estimate very approximate versions of a specified continuous model, you may use the `meanintervals = TRUE` argument to `ctFit`, which will simply set every individual's time intervals to the mean of the interval across all individuals. In cases with lots of variability in intervals this may substantially speed up optimization, but will also waste important information and bias parameters. For this reason, this is only recommended as a very first step for further exploration.

9. Limitations and future directions

Currently, a number of assumptions are present in the specification of continuous time models implemented in **ctsem**. Although the processes are allowed to begin at different levels and variances, from then on a time-invariant model is assumed. Thus, **ctsem** cannot presently account for time-varying aspects of the processes, except in the form of observed impulses or level changes via time dependent predictors. Although as with many discrete models we could free various parameters across measurement occasions, their meaning would become unclear, as each measurement occasion (set of n.manifest columns in the wide format data) may contain observations from many different times. Instead, models which allow parameters to vary as a function of time could be incorporated in the future as per the time-varying specification in Oud and Jansen (2000). As we fit using full information maximum likelihood, standard assumptions regarding multivariate normality apply to any manifest variables. Generalisations of the measurement model to allow for non-normal indicators could be easily implemented using the regular **OpenMx** functionality however. The multi-group implementation has no regularisation applied to the estimation of parameters that are free across groups, hence variance of such parameters may be too large (for a discussion of fixed and random effect models see Robinson 1991). Exogenous predictors (both time dependent and independent) cannot currently be specified with a measurement model, thus are assumed to be measured without error. This is straightforward to implement (though will involve more assumptions than present), and may be included at some stage. Presently, effects are assumed to be transmitted near instantaneously, as we do not estimate a dead time between inputs and the effect of inputs. Thus, when there is a dead time between an input and its' effect, the model is misspecified. We believe this is unlikely to severely impact estimates unless the dead time is of a similar or greater order of magnitude as observation intervals, however such a parameter can be estimated and incorporated within the continuous time equations (Richard 2003). Although in such areas **ctsem** may benefit from expansion, **ctsem** as it stands allows for the straightforward specification of continuous time dynamic models for both panel and time series data, which may include a measurement model, exogenous predictors, multiple processes, unobserved heterogeneity, multiple groups, as well as easy to specify parameter constraints and more complex dynamic specifications.

10. Acknowledgements

We would like to thank the Intra Person Dynamics and the Formal Methods research groups at the Max Planck Institute for Human Development for assistance with this work, as well as Joshua Pritikin and the **OpenMx** development team for feedback and fast response to issues.

References

Arnold L (1974). *Stochastic Differential Equations: Theory and Applications*. John Wiley & Sons, New York.

Balestra P, Nerlove M (1966). "Pooling Cross Section and Time Series Data in the Estimation of a Dynamic Model: The Demand for Natural Gas." *Econometrica*, **34**(3), 585–612. ISSN 0012-9682. doi:10.2307/1909771. URL http://www.jstor.org/stable/1909771.

Boker S, Neale M, Maes H, Wilde M, Spiegel M, Brick T, Spies J, Estabrook R, Kenny S, Bates T, Mehta P, Fox J (2011). "**OpenMx**: An Open Source Extended Structural Equation Modeling Framework." *Psychometrika*, **76**(2), 306–317. ISSN 0033-3123, 1860-0980. doi:10.1007/s11336-010-9200-6. URL http://link.springer.com/article/10.1007/s11336-010-9200-6.

Boker SM, Neale MC, Maes HH, Wilde MJ, Spiegel M, Brick TR, Estabrook R, Bates TC, Mehta P, Oertzen Tv, Gore RJ, Hunter MD, Hackett DC, Karch J, Brandmaier AM, Pritikin JN, Zahery M, Kirkpatrick RM, OpenMx T (2014). "**OpenMx** 2.0 User Guide." URL http://openmx.psyc.virginia.edu/docs/OpenMx/latest/OpenMxUserGuide.pdf.

Bollen KA, Brand JE (2010). "A General Panel Model with Random and Fixed Effects: A

Structural Equations Approach." *Social Forces*, **89**(1), 1–34. ISSN 0037-7732, 1534-7605. doi:10.1353/sof.2010.0072. URL http://sf.oxfordjournals.org/content/89/1/1.

Canova F, Ciccarelli M, Ortega E (2012). "Do Institutional Changes Affect Business Cycles? Evidence from Europe." *Journal of Economic Dynamics and Control*, **36**(10), 1520–1533. ISSN 0165-1889. doi:10.1016/j.jedc.2012.03.017. URL http://www.sciencedirect.com/science/article/pii/S0165188912000942.

Chow SM, Ho MhR, Hamaker EL, Dolan CV (2010). "Equivalence and Differences Between Structural Equation Modeling and State-Space Modeling Techniques." *Structural Equation Modeling: A Multidisciplinary Journal*, **17**(2), 303–332. ISSN 1070-5511. doi:10.1080/10705511003661553.

Coleman JS (1964). "Introduction to Mathematical Sociology." *London Free Press Glencoe.* URL http://www.cabdirect.org/abstracts/19651802199.html.

Delsing MJMH, Oud JHL (2008). "Analyzing Reciprocal Relationships by Means of the Continuous-Time Autoregressive Latent Trajectory Model." *Statistica Neerlandica*, **62**(1), 58–82. ISSN 1467-9574. doi:10.1111/j.1467-9574.2007.00386.x. URL http://onlinelibrary.wiley.com/doi/10.1111/j.1467-9574.2007.00386.x/abstract.

Driver CC, Oud JHL, Voelkle MC (2015). "Continuous Time Structural Equation Modelling With R Package **ctsem**." *Submitted to Journal of Statistical Software*.

Durbin J, Koopman SJ (2012). *Time Series Analysis by State Space Methods: Second Edition*. Oxford University Press. ISBN 9780199641178.

Ghisletta P, Lindenberger U (2005). "Exploring Structural Dynamics Within and Between Sensory and Intellectual Functioning in Old and Very Old Age: Longitudinal Evidence from the Berlin Aging Study." *Intelligence*, **33**(6), 555–587. ISSN 0160-2896. doi:10.1016/j.intell.2005.07.002. URL http://www.sciencedirect.com/science/article/pii/S0160289605000784.

Granger CWJ, Newbold P (1974). "Spurious Regressions in Econometrics." *Journal of Econometrics*, **2**(2), 111–120. ISSN 0304-4076. doi:10.1016/0304-4076(74)90034-7. URL http://www.sciencedirect.com/science/article/pii/0304407674900347.

Grijalva CG, Nuorti JP, Arbogast PG, Martin SW, Edwards KM, Griffin MR (2007). "Decline in Pneumonia Admissions After Routine Childhood Immunisation with Pneumococcal Conjugate Vaccine in the USA: A Time-Series Analysis." *The Lancet*, **369**(9568), 1179–1186. ISSN 0140-6736. doi:10.1016/S0140-6736(07)60564-9. URL http://www.sciencedirect.com/science/article/pii/S0140673607605649.

Halaby CN (2004). "Panel Models in Sociological Research: Theory into Practice." *Annual Review of Sociology*, **30**(1), 507–544. ISSN 0360-0572. doi:10.1146/annurev.soc.30.012703.110629. URL http://dx.doi.org/10.1146/annurev.soc.30.012703.110629.

Hamaker EL, Dolan CV, Molenaar PCM (2003). "ARMA-Based SEM When the Number of Time Points T Exceeds the Number of Cases N: Raw Data Maximum Likelihood." *Structural Equation Modeling: A Multidisciplinary Journal*, **10**(3), 352–379. ISSN 1070-5511. doi:10.1207/S15328007SEM1003_2. URL http://dx.doi.org/10.1207/S15328007SEM1003_2.

Hannan MT, Tuma NB (1979). "Methods for Temporal Analysis." *Annual Review of Sociology*, **5**(1), 303–328. ISSN 0360-0572. doi:10.1146/annurev.so.05.080179.001511. URL http://www.jstor.org/stable/2945957.

Helgason T, Tómasson H, Zoega T (2004). "Antidepressants and Public Health in Iceland. Time Series Analysis of National Data." *The British Journal of Psychiatry*, **184**(2), 157–162. ISSN 0007-1250, 1472-1465. doi:10.1192/bjp.184.2.157. 00118 PMID: 14754829, URL http://bjp.rcpsych.org/content/184/2/157.

Higham N (2009). "The Scaling and Squaring Method for the Matrix Exponential Revisited." *SIAM Review*, **51**(4), 747–764. ISSN 0036-1445. doi:10.1137/090768539. URL http://epubs.siam.org/doi/abs/10.1137/090768539.

Hunter MD (2014). *State Space Dynamic Mixture Modeling: Finding People With Similar Patterns Of Change*. Ph.D. thesis, University of Oklahoma, Norman, OK.

Keijsers L, Loeber R, Branje S, Meeus W (2011). "Bidirectional Links and Concurrent Development of Parent-Child Relationships and Boys' Offending Behavior." *Journal of Abnormal Psychology*, **120**(4), 878–889. ISSN 1939-1846(Electronic);0021-843X(Print). doi:10.1037/a0024588.

McArdle JJ, McDonald RP (1984). "Some Algebraic Properties of the Reticular Action Model for Moment Structures." *British Journal of Mathematical and Statistical Psychology*, **37**(2), 234–251. ISSN 2044-8317. doi:10.1111/j.2044-8317.1984.tb00802.x. URL http://onlinelibrary.wiley.com/doi/10.1111/j.2044-8317.1984.tb00802.x/full.

Mundlak Y (1978). "On the Pooling of Time Series and Cross Section Data." *Econometrica*, **46**(1), 69–85. ISSN 0012-9682. doi:10.2307/1913646. URL http://www.jstor.org/stable/1913646.

Neale MC, Hunter MD, Pritikin JN, Zahery M, Brick TR, Kirkpatrick RM, Estabrook R, Bates TC, Maes HH, Boker SM (2015). "**OpenMx** 2.0: Extended Structural Equation and Statistical Modeling." *Psychometrika*, pp. 1–15. ISSN 0033-3123, 1860-0980. doi:10.1007/s11336-014-9435-8. URL http://link.springer.com/article/10.1007/s11336-014-9435-8.

Oud JHL (2002). "Continuous Time Modeling of the Cross-Lagged Panel Design." *Kwantitatieve Methoden*, **69**(01), 1–26. URL http://members.chello.nl/j.oud7/Oud2002.pdf.

Oud JHL, Jansen RARG (2000). "Continuous Time State Space Modeling of Panel Data by Means of SEM." *Psychometrika*, **65**(2), 199–215. ISSN 0033-3123, 1860-0980. doi:10.1007/BF02294374. URL http://link.springer.com/article/10.1007/BF02294374.

Oud JHL, Singer H (2008). "Continuous Time Modeling of Panel Data: SEM Versus Filter Techniques." *Statistica Neerlandica*, **62**(1), 4–28. ISSN 1467-9574. doi:10.1111/j.1467-9574.2007.00376.x. URL http://onlinelibrary.wiley.com/doi/10.1111/j.1467-9574.2007.00376.x/abstract.

R Core Team (2014). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.

Richard JP (2003). "Time-Delay Systems: An Overview of Some Recent Advances and Open Problems." *Automatica*, **39**(10), 1667–1694. ISSN 0005-1098. doi:10.1016/S0005-1098(03)00167-5. URL http://www.sciencedirect.com/science/article/pii/S0005109803001675.

Rindskopf D (1984). "Using Phantom and Imaginary Latent Variables to Parameterize Constraints in Linear Structural Models." *Psychometrika*, **49**(1), 37–47. ISSN 0033-3123, 1860-0980. doi:10.1007/BF02294204. URL http://link.springer.com/article/10.1007/BF02294204.

Robinson GK (1991). "That BLUP is a Good Thing: The Estimation of Random Effects." *Statistical Science*, **6**(1), 15–32. ISSN 0883-4237, 2168-8745. doi:10.1214/ss/1177011926. URL http://projecteuclid.org/euclid.ss/1177011926.

Savalei V, Kolenikov S (2008). "Constrained Versus Unconstrained Estimation in Structural Equation Modeling." *Psychological Methods*, **13**(2), 150–170. ISSN 1939-1463(Electronic);1082-989X(Print). doi:10.1037/1082-989X.13.2.150.

Singer H (1998). "Continuous Panel Models with Time Dependent Parameters." *Journal of Mathematical Sociology*, **23**(2), 77–98. ISSN 0022250X. doi:10.1080/0022250X.1998.9990214. URL http://search.ebscohost.com/login.aspx?direct=true&db=sih&AN=4004337&site=ehost-live&scope=site.

Voelkle MC, Driver CC, Oud JHL (2015). "Accounting for Unobserved Heterogeneity in Panel Models with Unequal Sampling Intervals." *Manuscript in preparation*.

Voelkle MC, Oud JHL (2013). "Continuous Time Modelling with Individually Varying Time Intervals for Oscillating and Non-Oscillating Processes." *British Journal of Mathematical and Statistical Psychology*, **66**(1), 103–126. ISSN 2044-8317. doi:10.1111/j.2044-8317.2012.02043.x. URL http://onlinelibrary.wiley.com/doi/10.1111/j.2044-8317.2012.02043.x/abstract.

Voelkle MC, Oud JHL, Davidov E, Schmidt P (2012). "An SEM Approach to Continuous Time Modeling of Panel Data: Relating Authoritarianism and Anomia." *Psychological Methods*, **17**(2), 176–192. ISSN 1082-989X. doi:10.1037/a0027543.

von Oertzen T, Brandmaier AM, Tsang S (2015). "Structural Equation Modeling With Ωnyx." *Structural Equation Modeling: A Multidisciplinary Journal*, **22**(1), 148–161. ISSN 1070-5511. doi:10.1080/10705511.2014.935842. URL http://dx.doi.org/10.1080/10705511.2014.935842.

Affiliation:

Charles Driver
Center for Lifespan Psychology
Max Planck Institute for Human Development
Lentzeallee 94, 14195 Berlin
Telephone: +49 30 82406-367 E-mail: driver@mpib-berlin.mpg.de
URL: http://www.mpib-berlin.mpg.de/en/staff/charles-driver