# Simulation and Panel models

## A Critique of the Cross-Lagged Panel Model

Christopher Weber Stanley Feldman Adam Panish Bang Zheng

2024-12-09

### The Cross Lagged Panel Model

The CLPM is common in many social science applications, particularly in longitudinal, observational research designs. The design is straightforward, motivated by the difficulty in figuring out whether $X \rightarrow Y$, $Y \rightarrow X$. In political science, there are many examples (Federico and Gole De Zavala 2023; Craemer 2008; Claasen 2008; Hatemi, Crabtree and Smith 2019; Hetherington and Globeti 2002; Luttig 2021, Klandermans 2004; Lupu 2015; Goren and Chapp 2017; Layman and Carsey 2002; Putz 2002; Tilley, Neundor and Hobolt 2018; Sibley and Duckitt 2010, among many others), and among these examples many stem from political psychology. For instance.

| Reference | Description |
|---|---|
| Bakker, Lelkes and Malka (2021) | Politics and personality reciprocal effects |
| Brandt, Wetherell, and Henry (2015) | Income/SES predicts trust |
| Claibourn and Martin (2000) | Joining groups increases interpersonal trust |
| Campbell, Layman, Green and Sumaktoyo (2018) | Religion is endogenous to party |
| Carsey and Layman (2006) | Party attachment and ideology |
| Claasen (2008) | Party identification and campaign participation |
| Federico and Gole De Zavala (2023) | Collective narcissism and nationalism |
| Goren and Chapp (2017) | Culture war attitudes predict religion and partisan affiliation |
| Hetherington and Globetti (2002) | Trust predicts preferences for spending (among Whites) |
| Luttig (2021) | Authoritarianism is endogenous to party |
| Sibley and Duckitt (2010) | SDO, RWA |

However, many of the applied examples we identified in our literature review are from political psychology, in which the modeled variables are psychological characteristics that are relatively stable over time, such as authoritarianism and trust (Federico and Gole De Zavala 2023; Craemer 2008; Claasen 2008; Hatemi, Crabtree and Smith 2019; Hetherington and Globeti 2002; Luttig 2021, Klandermans 2004; Lupu 2015; Goren and Chapp 2017; Layman and Carsey 2002; Putz 2002; Tilley, Neundor and Hobolt 2018; Sibley and Duckitt 2010, among many others), and among these examples many stem from political psychology.

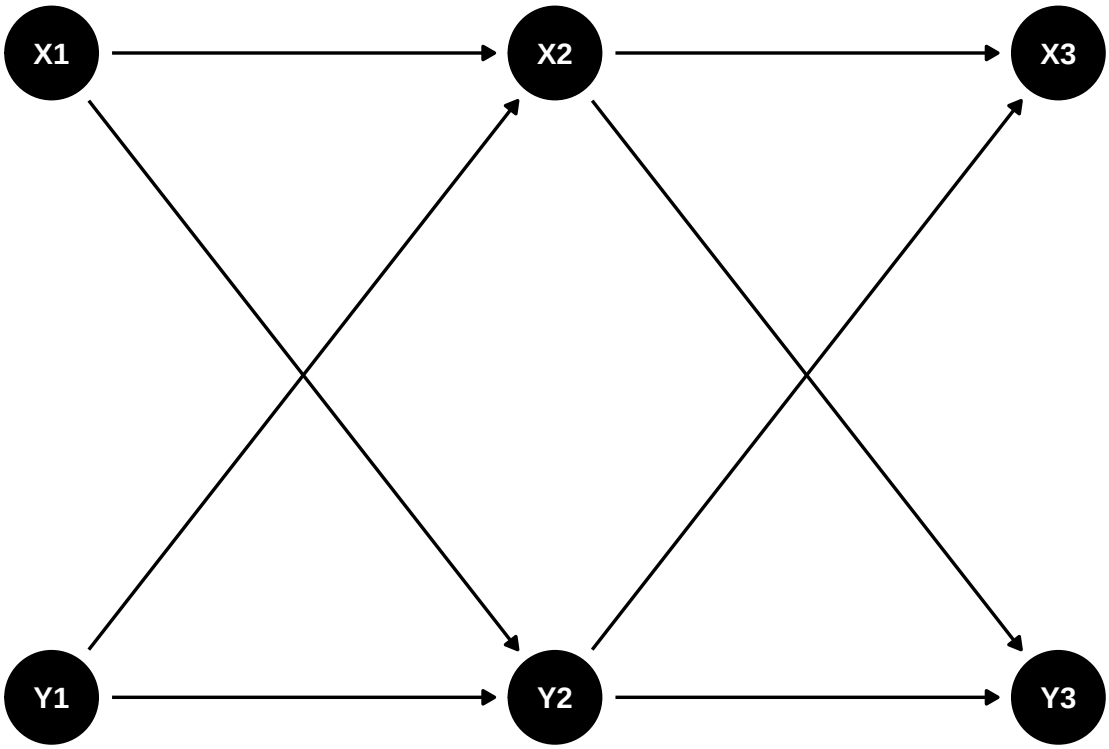
An emerging body of literature, particularly in psychology, has drawn attention to these issues, pointing to models that decompose the variance structure over time, into *stable, individual differences* , and *time-varying* differences. For instance, the **Random Intercept Crossed Lagged Regression** is an example of a model that decomposes the variance structure over time.

Ideally, one estimates the CLPM using a covariance structure model, which allows for a direct estimation of the panel error structure over time. While we do this below, we start with a more simplistic approach, reducing the problem to two linear equations – one for $x$ and one for $y$

## The Model

The CLPM is easily represented in graphical format as the image below (excluding correlated errors, latent variables, and control variable parameters.

```
library(dagitty)
library(ggdag)
library(dplyr)
library(lavaan)
library(crossLag)
library(ggplot2)
library(tibble)
library(ggridges)
library(gganimate)
library(knitr)
library(latex2exp)
devtools::load_all() # Change when complete

g <- dagitty("dag {
    X1 -> Y2
    Y1 -> Y2
    X1 -> X2
    Y1 -> X2
    X2 -> Y3
    Y2 -> Y3
    X2 -> X3
    Y2 -> X3

}")
coordinates(g) <- list(
    x = c(U1 = 0.5, Y1 = 1, X1 = 1, Y2 = 2, X2 = 2, X3 = 3, Y3 = 3),
    y = c(U1 = 1.5, Y1 = 1, X1 = 2, Y2 = 1, X2 = 2, X3 = 2, Y3 = 1)
)
ggdag(g, text = TRUE) + theme_dag() +
  ggtitle("The Cross-Lagged Panel Model (CLPM")
```




## SEM

The model may be represented as a system of equations, where each equation includes an AR(1) parameter and a cross-lag parameter, representing the effects of the lagged realization of the independent variable.

$$y_{t,i} = a_0 + a_1 y_{t-1,i} + a_2 x_{t-1,i} + e_{1,i} x_{t,i} = b_0 + b_1 y_{t-1,i} + b_2 x_{t-1,i} + e_{2,i}$$

In this simple framework, there are no contemporaneous effects specified, e.g, $X_3$ does not predict $Y_3$, nor does $X_2$ predict $Y_2$ nor $X_1$ predict $Y_1$. Instead, effects unroll over time. Perhaps a change in $X$ at wave 2 leads to an observed change in $Y$ at wave 3. By including the AR(1) parameter – the lagged dependent variable – the belief is that one controls for the stability in the dependent variable.

It is perhaps no wonder that the framework also appears well suited to tease out competing causal processes. The design leverages the temporal ordering of measurements in a panel data set. If $X$ is measured at time $t-1$ and $t$, and $Y$ is measured at time $t-1$ and $t$, then one can estimate a model that predicts $Y_t$ from $X_{t-1}$ and $Y_{t-1}$.

The problem with this logic, however, is that the CLPM correlational model, not a causal model, and this type of is "causation" is often understood to mean something different than causal structures in an experimental settings. It is called *Granger Causality*: $X$ is said to **Granger cause** $Y$ if $X_{t-1}$ predicts $Y_t$ better than $Y_{t-1}$ predicts $X_t$. This is a statistically derived approach to causality, and it is not necessarily consistent with common approaches to causal inference, like the potential outcomes framework. For the same reason that observing a relationship between two variables in a cross section design does not establish causation. Because two variables unroll across time in a similar manner, this does **not** mean one causes the other. Through simulation we will demonstrate the following.

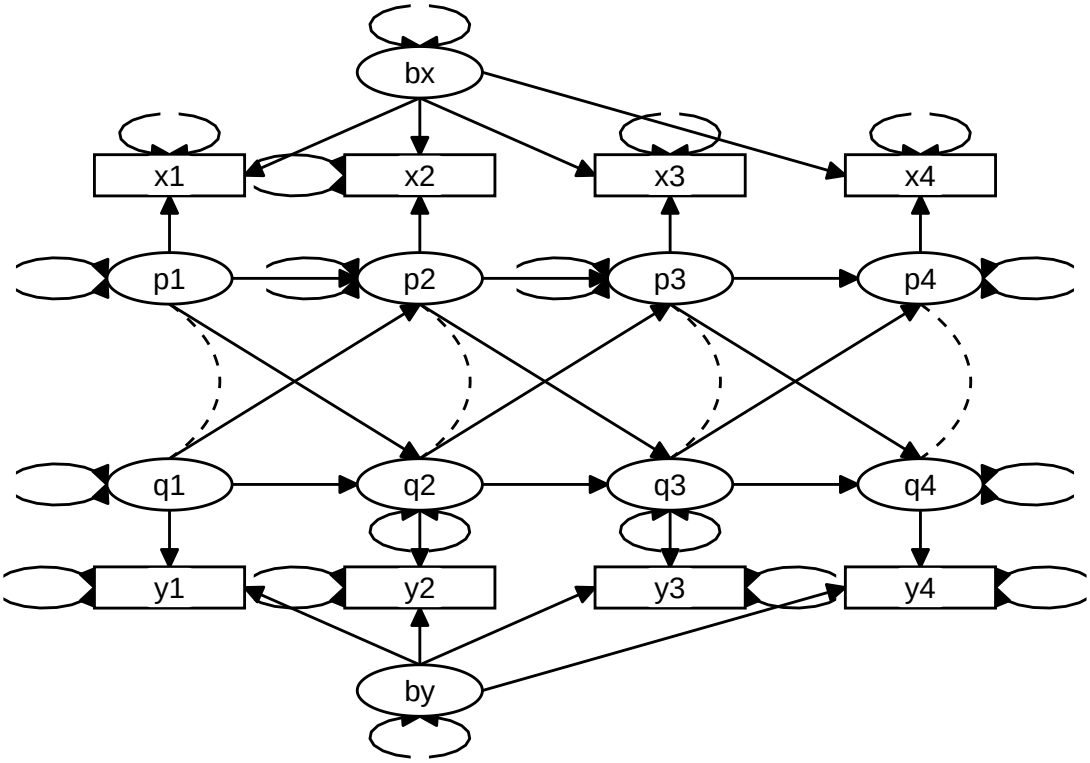
The CLPM is not a causal model, though even as a correlational model it presents several methodological limitations. Consider published research in political science – Table XX – the vast majority of CLPM research has been applied to two-, occasionally three- wave panel structures. This can be problematic, particularly if

the variables under consideration are stable over time. Somewhat ironically, the greater the autocorrelation in $y$, the the harder it is to accurately estimate the effect of $x_{t-1}$.

## Simulate

The Random-Intercept Cross Lagged Panel (RI-CLPM) is specified as:

```
library(tidySEM)
library(lavaan)
library(devtools)
library(dplyr)
devtools::load_all() # Change when complete

## This is the way to structure the plot
lavaan(model_syntax_clpm(waves = 4, model_type = "clpm"),
                      simulate_riclpm(waves = 4, stability.p = 0.5, stability.q = 0.5, sample.nobs = 150
```




In a CLPM, variables are typically measured at multiple time points, and the model includes paths that represent the stability of each variable over time, the *autoregressive* effects. Oftentimes, constructs are stable because they are stable, psychological or political characteristics. Characteristics like *authoritarianism*, *political ideology*, *partisanship* (Campbell, Converse, Miller and Stokes 1960), *personality traits*, and *political trust* are generally quite stable over time, with vary little fluctuation in scores from wave to wave.

The random intercept model is a mixed-effects model that includes lagged and cross lagged effects, as well as a *random intercept* that captures stable individual differences in the $x$ and $y$ variables. There is a *between subject* source of variation, and there is also a *within subject* source of variation. By specifying a random-intercept, the autoregressive and cross-lagged parameters represent the *within-subject* effects, and whether a change in $x$ predicts a change in $y$, **controlling** for the stability, relatively constant variation in $x$ and $y$.

## The Model

Let's start with a relatively simple demonstration. We'll simulate a three-wave panel dataset assuming the RI-CLPM is the data generating process. We'll then estimate a CLPM using linear regression, and two equations, one for $x$ and one for $y$.

$$y_{t,i} = a_0 + a_1 y_{t-1,i} + a_2 x_{t-1,i} + e_{1,i} x_{t,i} = a_0 + b_1 y_{t-1,i} + b_2 x_{t-1,i} + e_{2,i}$$

Let's simulate a 3 wave panel, 10 trials, data generated using the RI-CLPM, and no "residual" autoregressive effect of $x$ or $y$ in the equations. The rhs slope coefficients are set to zero in the DGP. Likewise, there is no correlation between random intercepts, the variance of the indicators are set to 1 and the variances of the random intercepts is set to 0.5. The sample size is fixed at 1000.

It's useful to use the Monte Carlo function to explore the characteristics of an OLS model in different, but often very plausible data generating situations.

The data were generated using a wrapper function, **monteCarlo_OLS**. Let's assume the autoregressive parameters are null; there is no lagged effect of $x$ or $y$ on contemporary $x$ or $y$. Because the DGP for the $x$ and $y$ variables are the same – they have the same functional form and stem from the parametrically identical distributions – we need not present the simulated effects for both equations.

The data were generated using this functional form:

$$y_{t,i} = a_0 + a_1 y_{t-1,i} + a_2 x_{t-1,i} + e_{1,i} x_{t,i} = a_0 + b_1 y_{t-1,i} + b_2 x_{t-1,i} + e_{2,i}$$

And we present the Monte Carlo estimates for this: $x_{t,i} = a_0 + b_1 y_{t-1,i} + b_2 x_{t-1,i} + e_{2,i}$ equation.

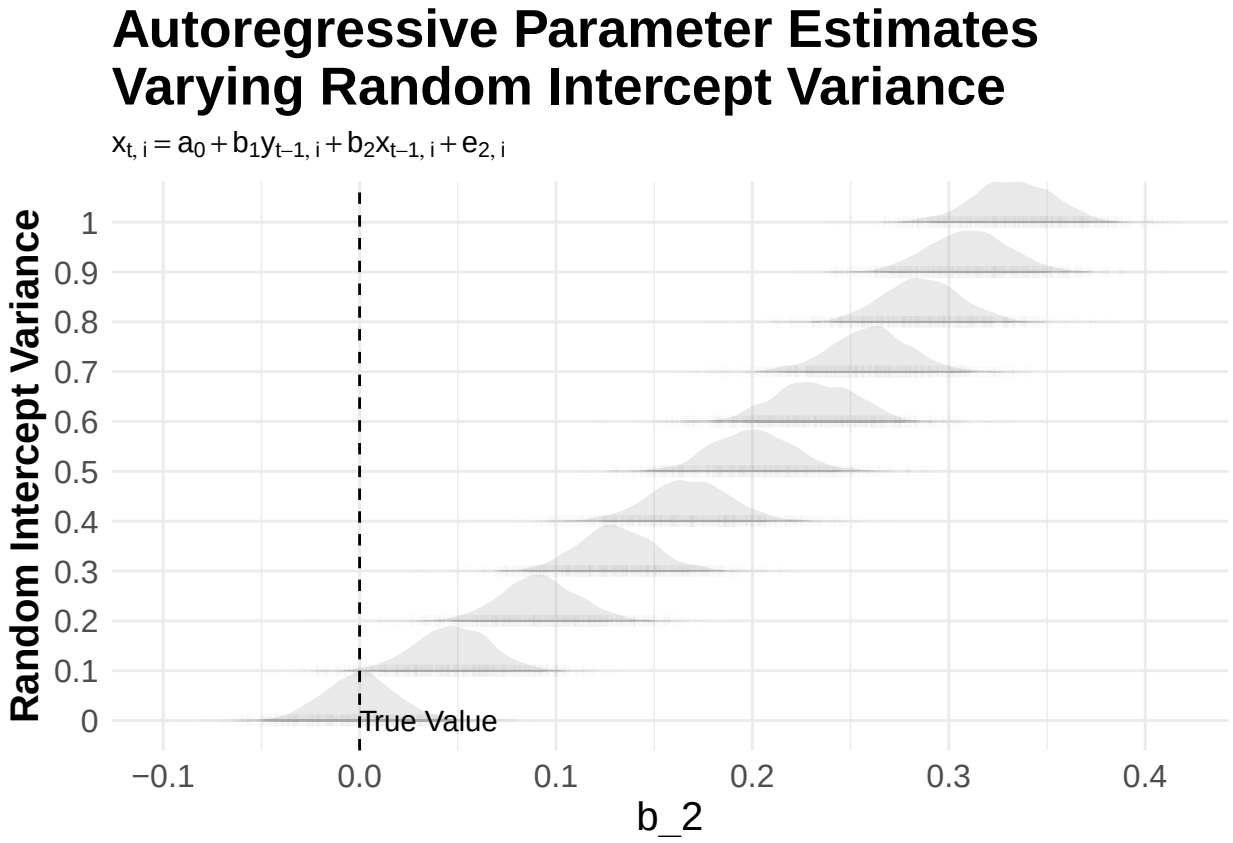
```
# Simulate data and estimate a seemingly unrelated regression
# Load the necessary libraries
monteCarlo_OLS(
   trials = 1,
   waves = 10,
   data_generation = "ri-clpm",
   within_person_stability_y = 0.0,
   within_person_stability_x = 0.0,
   cross_lag_x = 0.0,
   cross_lag_y = 0.0,
   variance.q = 0.1,
   variance.p = 0.1,
   variance.between.x = 1,
   variance.between.y = 1,
   cov.between = 0.5,
   cov.pq = 0.5,
   sample_size = 1000) -> results

var.x = seq(0, 1, by = .1)
var.y = seq(0, 1, by = .1)

for(x in var.x){
  for(y in var.y){
        results <- rbind(results,
                         monteCarlo_OLS(
                                        trials = 100,
                                        waves = 4,
                                        data_generation = "ri-clpm",
```

```
                                        within_person_stability_y = 0,
                                        within_person_stability_x = 0,
                                        cross_lag_x = 0.0,
                                        cross_lag_y = 0.0,
                                        variance.q = 0.1,
                                        variance.p = 0.1,
                                        variance.between.x = x,
                                        variance.between.y = y,
                                        cov.between = 0,
                                        cov.pq = 0,
                                        sample_size = 1000)
                    )
    }
  }
```

I varied the variance of the random intercept. Holding all else constant, as the random intercept variance increases, the estimates of the cross-lagged effects become more biased. This is because the random intercept captures the stable, individual differences in the $x$ and $y$ variables. If we ignore these stable differences, the autoregressive parameter will be biased – moving away from zero.

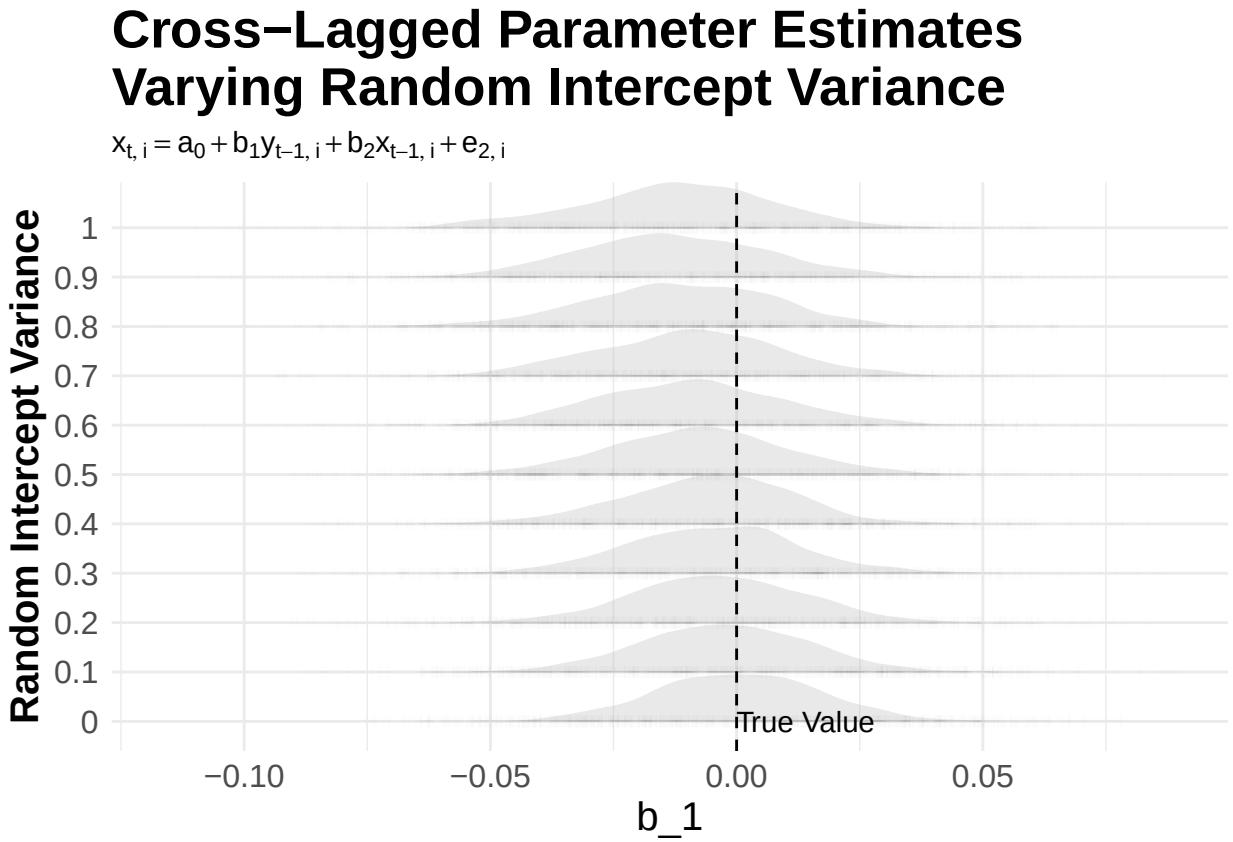
```
library(latex2exp)
results %>%
  filter(cov.between  != 0.5 ) %>%
  ggplot(aes(x = xlag_x, fill = "grey", y = as.factor(round(variance.between.x, 2)))) +
  geom_density_ridges2(
    alpha = 0.25,
    scale = 1,
    show.legend = FALSE,
    quantile_lines = TRUE,
    rel_min_height = 0.01,
    quantiles = 0.5,
    calc_ecdf = FALSE,
    jittered_points = TRUE,
    position = position_points_jitter(width = 0.05, height = 0),
    point_shape = 3,
    point_size = 1,
    point_color = "darkgrey",
    point_alpha = 0.01,
    fill = "darkgrey",
    color = NA  # Remove the outline

  ) +
  # scale_x_continuous("", limits = c(-0.25, 0.25)) +
  scale_y_discrete("Random Intercept Variance") +
  scale_x_continuous(TeX(" b_2 ")) +
  theme_minimal()  +
  # Draw a vertical line at 0 that is dashed, use abline
  geom_vline(xintercept = 0, linetype = "dashed", color = "black") +
  geom_hline(yintercept = 0, linetype = "dashed", color = "black") +
  annotate("text", x = 0, y = 1, label = "True Value ", hjust = 0) +
  labs(title = "Autoregressive Parameter Estimates\nVarying Random Intercept Variance", subtitle = TeX(
  # Style the plot
  theme(
    plot.title = element_text(size = 20, face = "bold"),
```

```
    axis.title = element_text(size = 15, face = "bold"),
    axis.text = element_text(size = 12),
    legend.title = element_text(size = 15, face = "bold"),
    legend.text = element_text(size = 12)
  )
```




```
results %>%
  filter(cov.between  != 0.5 ) %>%
  ggplot(aes(x = xlag_y, fill = "grey", y = as.factor(round(variance.between.y, 2)))) +
  geom_density_ridges2(
    alpha = 0.25,
    scale = 1,
    show.legend = FALSE,
    quantile_lines = TRUE,
    rel_min_height = 0.01,
    quantiles = 0.5,
    calc_ecdf = FALSE,
    jittered_points = TRUE,
    position = position_points_jitter(width = 0.05, height = 0),
    point_shape = 3,
    point_size = 1,
    point_color = "darkgrey",
    point_alpha = 0.01,
    fill = "darkgrey",
    color = NA  # Remove the outline

  ) +
```

```
# scale_x_continuous("", limits = c(-0.25, 0.25)) +
scale_y_discrete("Random Intercept Variance") +
scale_x_continuous(TeX(" b_1 ")) +
theme_minimal()  +
# Draw a vertical line at 0 that is dashed, use abline
geom_vline(xintercept = 0, linetype = "dashed", color = "black") +
geom_hline(yintercept = 0, linetype = "dashed", color = "black") +
annotate("text", x = 0, y = 1, label = "True Value ", hjust = 0) +
labs(title = "Cross-Lagged Parameter Estimates\nVarying Random Intercept Variance", subtitle = TeX("$
# Style the plot
theme(
  plot.title = element_text(size = 20, face = "bold"),
  axis.title = element_text(size = 15, face = "bold"),
  axis.text = element_text(size = 12),
  legend.title = element_text(size = 15, face = "bold"),
  legend.text = element_text(size = 12)
)
```




This is not entirely surprising. The non-hierarchical OLS model is not able to account for the stable, individual differences in the $x$ and $y$ variables. The total variance in the dependent variable is a consequence of variance between units, as well as the variance within units over time. However, the OLS model does not partition the variance in this way. Instead, the model we have estimated assumes a single source of error; the residuals for each respondent at each wave. As such, the autoregressive parameter, though it should be zero, is not zero, it is biased by the degree of variance in the random intercepts.

In this circumstance, the cross-lagged parameter should be - and is – close to zero. Even though the cross-lagged variable also has some degree of stability, it's stability is not systematically related to the stability in the dependent variable.

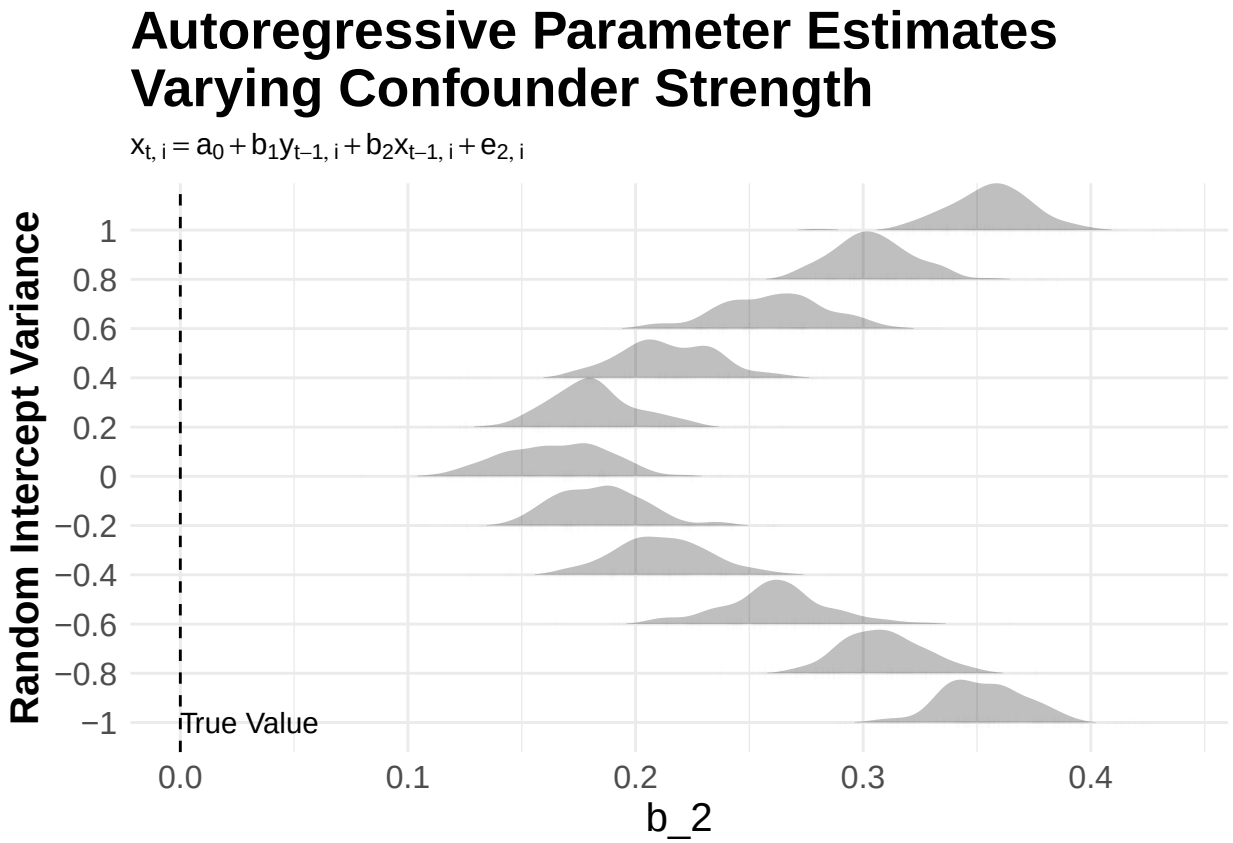
## A Confounder

```
monteCarlo_OLS(
   trials = 1,
   waves = 10,
   confound = 0.5,
   data_generation = "ri-clpms",
   within_person_stability_y = 0.0,
   within_person_stability_x = 0.0,
   cross_lag_x = 0.0,
   cross_lag_y = 0.0,
   variance.q = 0.1,
   variance.p = 0.1,
   variance.between.x = 1,
   variance.between.y = 1,
   cov.between = 0.5,
   cov.pq = 0.5,
   sample_size = 1000) -> results

confound = seq(-1, 1, by = .2)

for(c in confound){
        results <- rbind(results,
                         monteCarlo_OLS(
                                        trials = 100,
                                        waves = 4,
                                        data_generation = "ri-clpms",
                                        within_person_stability_y = 0,
                                        within_person_stability_x = 0,
                                        cross_lag_x = 0.0,
                                        cross_lag_y = 0.0,
                                        variance.q = 0.4,
                                        variance.p = 0.4,
                                        variance.between.x = 0.4,
                                        variance.between.y = 0.4,
                                        cov.between = 0,
                                        cov.pq = 0,
                                        confound = c,
                                        sample_size = 1000)
                    )
  }
```

### Autoregression Parameter Estimates

```
results %>%
  filter(cov.between  != 0.5 ) %>%
  ggplot(aes(x = xlag_x, fill = "grey", y = as.factor(round(confound, 2)))) +
  geom_density_ridges2(
    alpha = 0.25,
    scale = 1,
    show.legend = FALSE,
    quantile_lines = TRUE,
    rel_min_height = 0.01,
```

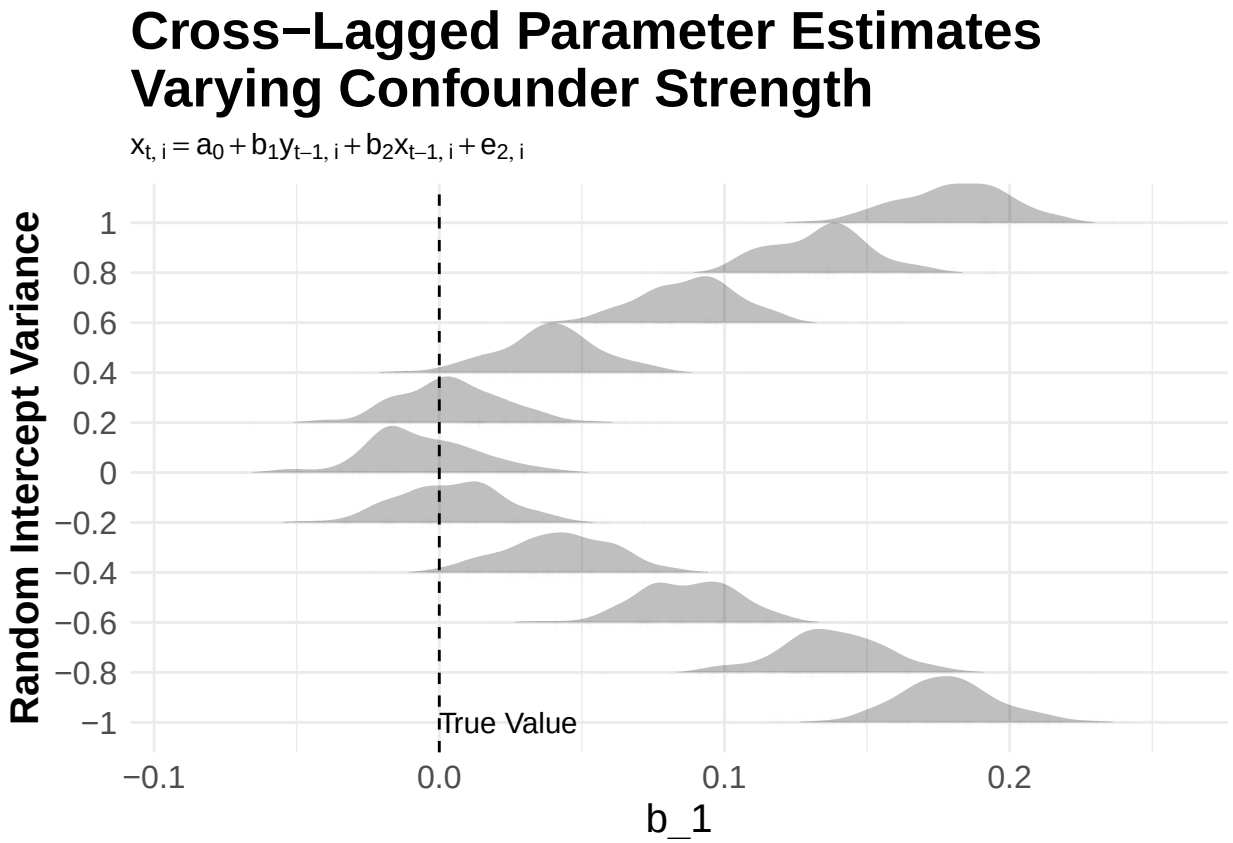
```
    quantiles = 0.5,
    calc_ecdf = FALSE,
    jittered_points = TRUE,
    position = position_points_jitter(width = 0.05, height = 0),
    point_shape = 3,
    point_size = 1,
    point_color = "darkgrey",
    point_alpha = 0.01,
    fill = "black",
    color = NA  # Remove the outline

  ) +
  # scale_x_continuous("", limits = c(-0.25, 0.25)) +
  scale_y_discrete("Random Intercept Variance") +
  scale_x_continuous(TeX(" b_2 ")) +
  theme_minimal()  +
  # Draw a vertical line at 0 that is dashed, use abline
  geom_vline(xintercept = 0, linetype = "dashed", color = "black") +
  geom_hline(yintercept = 0, linetype = "dashed", color = "black") +
  annotate("text", x = 0, y = 1, label = "True Value ", hjust = 0) +
  labs(title = "Autoregressive Parameter Estimates\nVarying Confounder Strength", subtitle = TeX("$x_{t
  # Style the plot
  theme(
    plot.title = element_text(size = 20, face = "bold"),
    axis.title = element_text(size = 15, face = "bold"),
    axis.text = element_text(size = 12),
    legend.title = element_text(size = 15, face = "bold"),
    legend.text = element_text(size = 12)
  )
```




**Cross Lagged Parameter Estimates**

```
results %>%
  filter(cov.between  != 0.5 ) %>%
  ggplot(aes(x = xlag_y, fill = "grey", y = as.factor(round(confound, 2)))) +
  geom_density_ridges2(
    alpha = 0.25,
    scale = 1,
    show.legend = FALSE,
    quantile_lines = TRUE,
    rel_min_height = 0.01,
    quantiles = 0.5,
    calc_ecdf = FALSE,
    jittered_points = TRUE,
    position = position_points_jitter(width = 0.05, height = 0),
    point_shape = 3,
    point_size = 1,
    point_color = "darkgrey",
    point_alpha = 0.01,
    fill = "black",
    color = NA  # Remove the outline

  ) +
  # scale_x_continuous("", limits = c(-0.25, 0.25)) +
  scale_y_discrete("Random Intercept Variance") +
  scale_x_continuous(TeX(" b_1 ")) +
```

```r
  theme_minimal()  +
  # Draw a vertical line at 0 that is dashed, use abline
  geom_vline(xintercept = 0, linetype = "dashed", color = "black") +
  geom_hline(yintercept = 0, linetype = "dashed", color = "black") +
  annotate("text", x = 0, y = 1, label = "True Value ", hjust = 0) +
  labs(title = "Cross-Lagged Parameter Estimates\nVarying Confounder Strength", subtitle = TeX("$x_{t, i
  # Style the plot
  theme(
    plot.title = element_text(size = 20, face = "bold"),
    axis.title = element_text(size = 15, face = "bold"),
    axis.text = element_text(size = 12),
    legend.title = element_text(size = 15, face = "bold"),
    legend.text = element_text(size = 12)
  )
```




Not surprisingly, the presence of a confounder biases **both** cross-lagged and autoregressive parameter estimates.