Hypothesis Testing with Error Correction Models

Replication Log File

Patrick W. Kraft* Ellen M. Key† Matthew J. Lebo‡

February 8, 2021

R Code & Output

```
# ====================================================================== #
# Project: Hypothesis Testing with Error Correction Models
# Author: Patrick
# Date: 02/26/2019 (last update: 02/08/2021)
# Summary: Run simulations and create plots for paper and appendix
# ====================================================================== #

## Set save_plots = TRUE if plots should be saved as pdfs in current working directory
save_plots <- FALSE

## Set nsim = 10 to reduce runtime (nsim = 1000 required to replicate exact results)
nsim <- 1000


# Load packages ------------------------------------------------------------

library(car)
```

```
## Loading required package: carData
```

```
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following object is masked from 'package:car':
##
##     recode
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

*University of Wisconsin-Milwaukee
†Appalachian State University
‡University of Western Ontario

```
library(tidyr)
library(purrr)
```

```
##
## Attaching package: 'purrr'
```

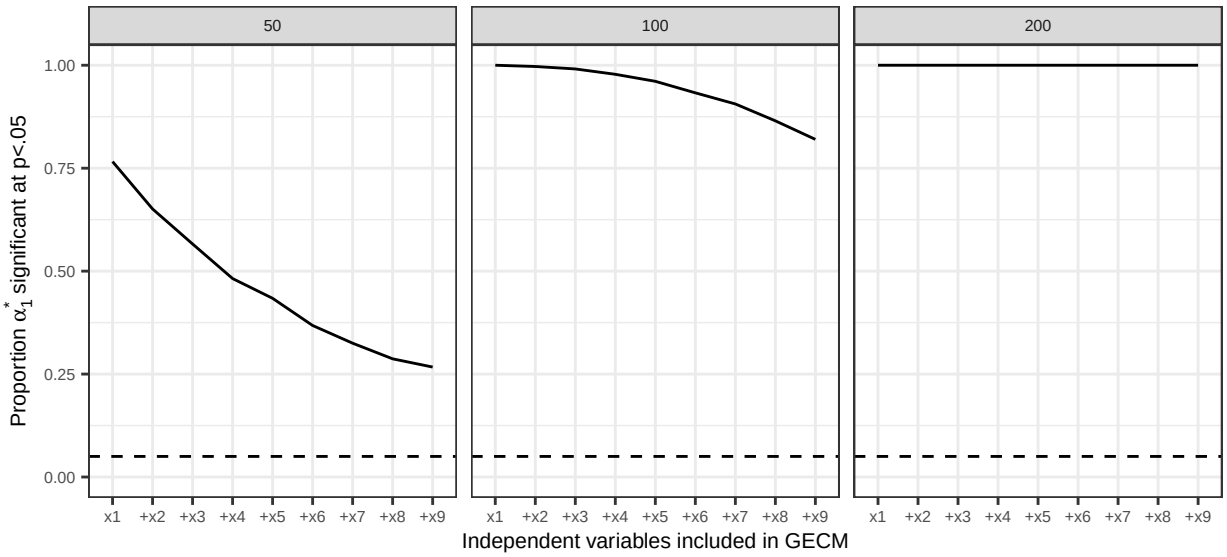
```
## The following object is masked from 'package:car':
##
##     some
```

```
library(broom)


# Custom functions ------------------------------------------------------

## compute Ericsson/MacKinnon critical values based on response surface estimates and t
mkFunc <- function(k, t){
  ## k: number of variables in ecm (includes dv!)
  ## t: length of time series

  ## response surface estimates from Ericsson/MacKinnon (2002)
  ## (5% level with constant and no trend (Table 3))
  tab3 <- tribble(
    ~k, ~theta_inf, ~theta_1, ~theta_2, ~theta_3,
    1,   -2.8617,   -2.81,    -3.2,      37,
    2,   -3.2145,   -3.21,    -2.0,      17,
    3,   -3.5057,   -3.27,     1.1,     -34,
    4,   -3.7592,   -2.92,    -3.7,       5,
    5,   -3.9856,   -2.50,    -1.7,     -35,
    6,   -4.1922,   -1.73,    -7.8,      -9,
    7,   -4.3831,   -0.90,   -12.2,       1,
    8,   -4.5608,    0.02,   -15.4,      -2,
    9,   -4.7287,    1.25,   -26.0,      42,
    10,   -4.8876,    2.46,   -31.7,      43,
    11,   -5.0394,    3.88,   -45.7,     117,
    12,   -5.1836,    5.33,   -55.9,     134
  )

  ## compute adjusted sampe size (assuming constant w/o trend)
  ## (c.f., p.295)
  t_adj <- t - (2*k-1) - 1

  ## compute critical value
  crit <- tab3$theta_inf[k] + tab3$theta_1[k]/t_adj +
    tab3$theta_2[k]/t_adj^2 + tab3$theta_3[k]/t_adj^3
  crit
}

## estimate multivariate GECM
gecmEst <- function(dframe){
  ## Extract variables (first column in dframe is dv!)
  # y: dependent variable (vector)
  # x: independent variables (matrix)
  x  <- as.matrix(dframe[,-1])
  y <- as.vector(dframe[,1])
```

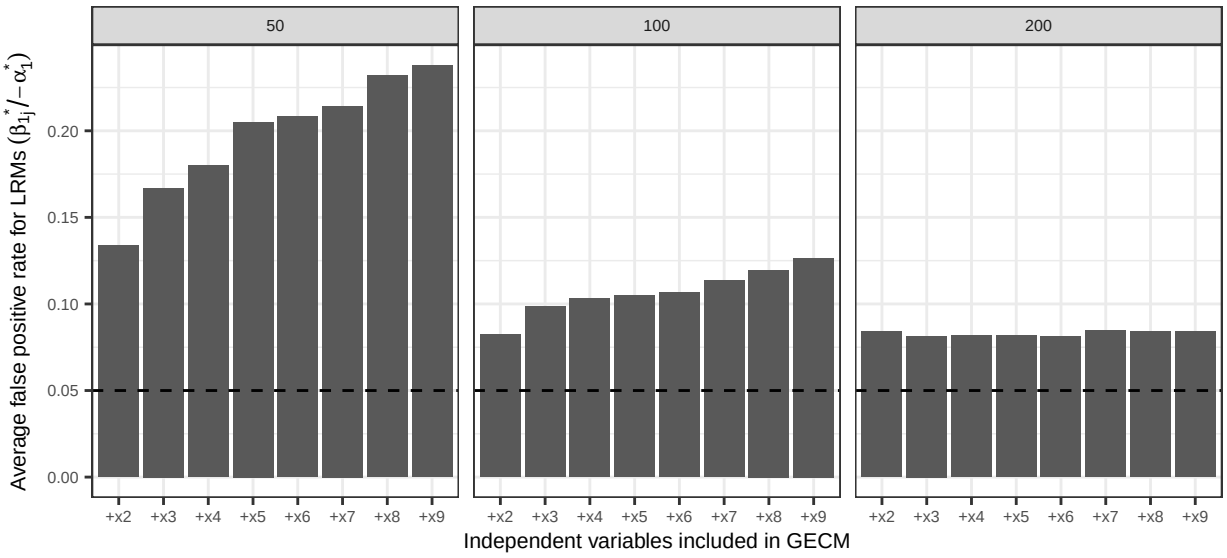
```r
  ## compute lagged and differenced series
  dy <- diff(y)
  ly <- y[-length(y)]
  dx <- diff(x)
  lx <- as.matrix(x[-nrow(x),])

  ## fix variable names
  colnames(dx) <- paste0("d",colnames(dx))
  if(ncol(dx)==1) colnames(dx) <- "dx1"
  colnames(lx) <- paste0("l",colnames(lx))
  if(ncol(lx)==1) colnames(lx) <- "lx1"

  ## combine variables in new data frame
  iv_ <- cbind(ly,dx,lx)

  ## estimate GECM
  m <- lm(dy ~ iv_)

  ## extract model results, remove intercept
  tab <- tidy(m) %>%
    filter(term != "(Intercept)")

  ## check for cointegration using MacKinnon values
  coint <- tibble(
    type = "Cointegration (alpha_1^*)",
    var = "y",
    coef = tab$estimate[tab$term=="iv_ly"],
    sig = tab$statistic[tab$term=="iv_ly"] < mkFunc(k = ncol(dframe), t = nrow(dframe))
  )

  ## long-run effects (beta_1^*/(-alpha_1^*))
  lr <- paste0("iv_lx",1:ncol(x),"/(-iv_ly)") %>%
    map_dfr(~deltaMethod(m, .)) %>%
    mutate(type = "Long run (beta_1^*/(-alpha_1^*))",
           var = paste0("x",1:ncol(x)),
           coef = NA,
           sig = abs(Estimate/SE)>qnorm(.975)) %>%
    select(type, var, sig)

  ## output
  out <- bind_rows(coint, lr) %>%
    mutate(k = ncol(dframe),
           n = nrow(dframe),
           coint_sig = coint$sig,
           comb_sig = case_when(
             var == "y" ~ coint_sig,
             var != "y" & coint_sig ~ sig)
    ) %>%
    select(k, n, type, var, coef, sig, coint_sig, comb_sig)
  return(out)
}

## simulate two cointegrated time series and additional unrelated regressors
```

```r
## estimate multiple GECMs that incrementally add unrelated regressors
cointSim <- function(nT = 50, arimalist = list(ar = c(0.6)), sd = 1){
  x1 <- cumsum(rnorm(nT))
  e1 <- arima.sim(n = nT, arimalist, sd=sd)
  y <- x1 + e1
  x2 <- cumsum(rnorm(nT))
  x3 <- cumsum(rnorm(nT))
  x4 <- cumsum(rnorm(nT))
  x5 <- cumsum(rnorm(nT))
  x6 <- cumsum(rnorm(nT))
  x7 <- cumsum(rnorm(nT))
  x8 <- cumsum(rnorm(nT))
  x9 <- cumsum(rnorm(nT))
  tmp <- data.frame(y, x1, x2, x3, x4, x5, x6, x7, x8, x9)

  ## estimate multiple GECMs
  out <- bind_rows(gecmEst(tmp[,1:2]),
                   gecmEst(tmp[,1:3]),
                   gecmEst(tmp[,1:4]),
                   gecmEst(tmp[,1:5]),
                   gecmEst(tmp[,1:6]),
                   gecmEst(tmp[,1:7]),
                   gecmEst(tmp[,1:8]),
                   gecmEst(tmp[,1:9]),
                   gecmEst(tmp[,1:10]))
  return(out)
}

## simulate true GECM DGP
gecmSim <- function(alpha0 = 1,
                    alpha1 = -.4,
                    beta0 = c(0.5, rep(0,8)),
                    beta1 = c(0.5, rep(0,8)),
                    nT = 50, sd = 1, endo = NULL){
  if(length(beta0) != length(beta1)) stop("beta0 and beta1 have to be of same length")
  eps <- rnorm(nT, sd = sd)
  k <- length(beta0)
  X <- replicate(k, cumsum(rnorm(nT)))
  dX <- rbind(rep(NA, k), apply(X, 2, diff))
  y <- c(rnorm(1), rep(NA, nT-1))
  dy <- rep(NA, nT)
  for(t in 2:nT){
    dy[t] <- alpha0 + alpha1*y[t-1] + sum(beta0*dX[t,]) + sum(beta1*X[t-1,]) + eps[t]
    y[t] <- y[t-1] + dy[t]
  }
  if(!is.null(endo)){
    for(i in 1:length(endo)){
      X[,endo[i]] <- X[,endo[i]] + eps + rnorm(nT)
    }
  }
  out <- data.frame(cbind(y, X))
  colnames(out) <- c("y", paste0("x",1:k))
  return(out)
```

```
}

## wrapper to estimate multiple gecms with additional covariates
iterSim <- function(...){
  tmp <- gecmSim(...)
  out <- bind_rows(gecmEst(tmp[,1:2]),
                   gecmEst(tmp[,1:3]),
                   gecmEst(tmp[,1:4]),
                   gecmEst(tmp[,1:5]),
                   gecmEst(tmp[,1:6]),
                   gecmEst(tmp[,1:7]),
                   gecmEst(tmp[,1:8]),
                   gecmEst(tmp[,1:9]),
                   gecmEst(tmp[,1:10]))
  return(out)
}



set.seed(20190226)


# Main analysis and plots ------------------------------------------------

## run simulations
res <- rerun(nsim,
             bind_rows(cointSim(nT = 50),
                       cointSim(nT = 100),
                       cointSim(nT = 200))) %>%
  map_dfr(.,~mutate(., k = factor(k, labels = c("x1", "+x2", "+x3", "+x4", "+x5",
                                                "+x6", "+x7", "+x8", "+x9")),
                    var = recode_factor(var, y = "y", x1 = "x1",
                                        .default = "unrelated"))) %>%
  group_by(k, n, var, type) %>%
  summarize_all(mean, na.rm = T)

## Figure 1: The consequences of GECMs with unbalanced equations
res %>% ungroup() %>%
  filter(var == "y") %>%
  ggplot(aes(x=k, y=comb_sig)) +
  geom_hline(yintercept=.05, lty = "dashed") +
  geom_line(aes(group=var)) + facet_wrap(~n) +
  theme_bw() + xlab("Independent variables included in GECM") +
  ylab(expression(paste("Proportion ",alpha[1]^"*"," significant at p<.05"))) +
  theme(text = element_text(size=8)) + ylim(0,1)
```

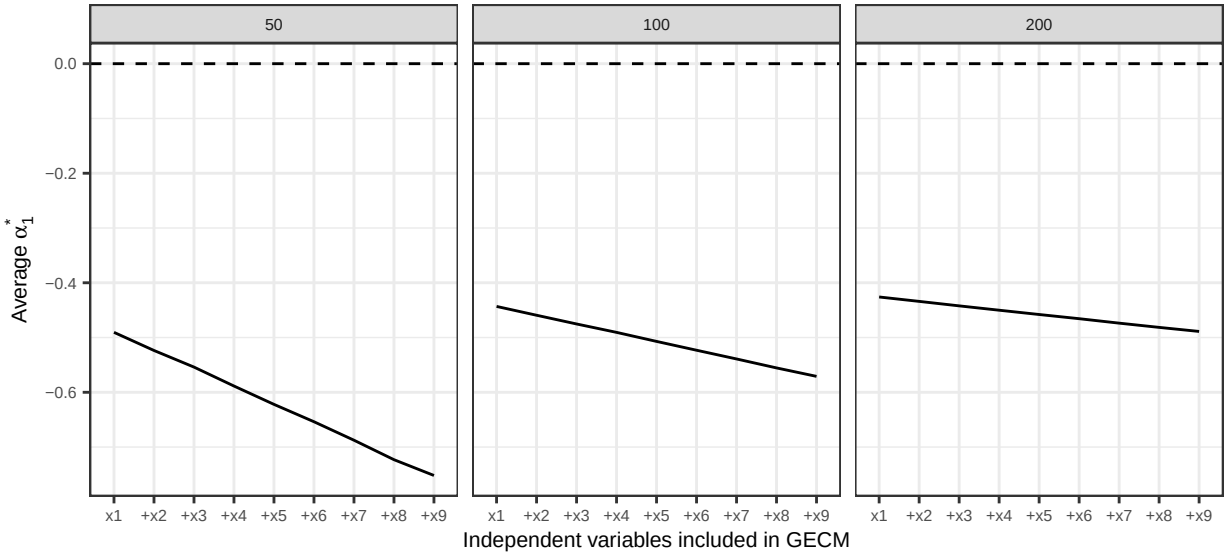



```
if(save_plots) ggsave("fig1.pdf", height = 3, width = 6.5)

## Figure 2: The consequences of GECMs with unbalanced equations
res %>% ungroup() %>%
  filter(var == "unrelated") %>%
  mutate(type = gsub(" (","\n(", type, fixed =T)) %>%
  ggplot(aes(x=k, y=comb_sig)) +
  geom_bar(stat="identity") +
  theme_bw() + xlab("Independent variables included in GECM") +
  ylab(expression(paste("Average false positive rate for LRMs ",(beta[1]["j"]^"*"/-alpha[1]^"*")))) +
  geom_hline(yintercept=.05, lty = "dashed") + facet_wrap(~n) +
  theme(text = element_text(size=8))
```

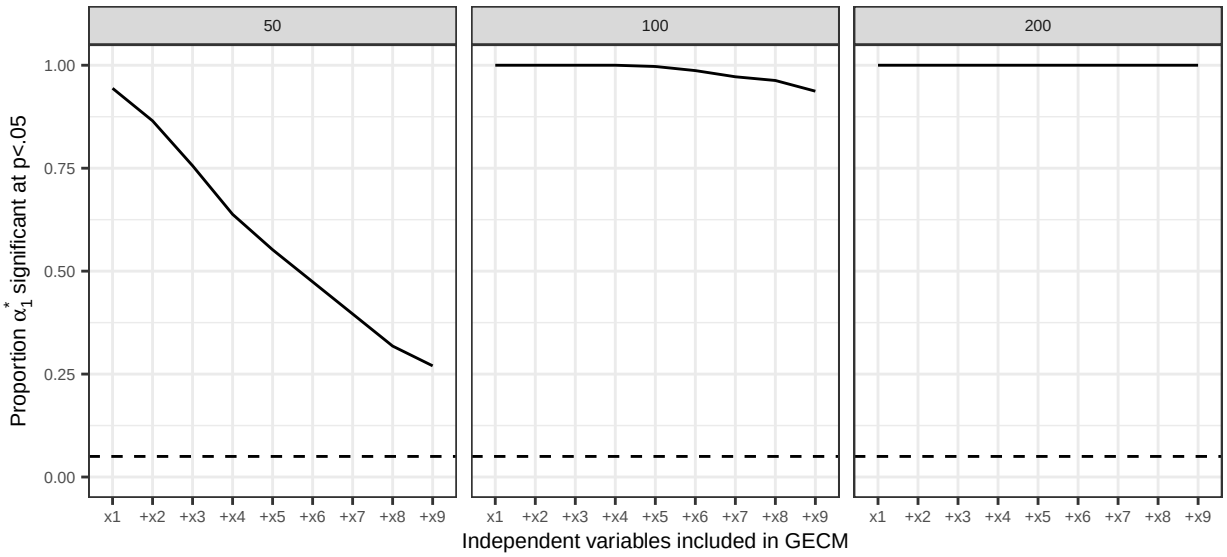



```
if(save_plots) ggsave("fig2.pdf", height = 3, width = 6.5)

## Appendix Figure 1: plot average alpha_1* across simulations
res %>% ungroup() %>%
  filter(var == "y") %>%
```

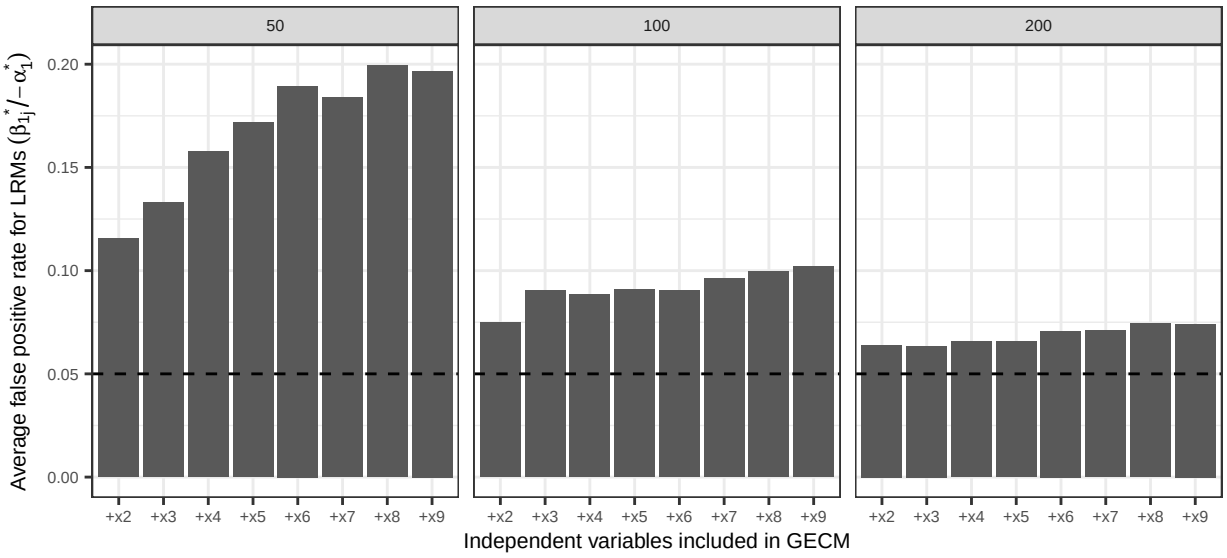
```
  ggplot(aes(x=k, y=coef)) +
  geom_hline(yintercept=0, lty = "dashed") +
  geom_line(aes(group=var)) + facet_wrap(~n) +
  theme_bw() + xlab("Independent variables included in GECM") +
  ylab(expression(paste("Average ",alpha[1]^"*"))) +
  theme(text = element_text(size=8))
```




```
if(save_plots) ggsave("app0.pdf", height = 3, width = 6.5)


# Appendix: alternative DGP ------------------------------------------------

set.seed(20190226)

## run simulations
res2 <- rerun(nsim,
              bind_rows(iterSim(nT = 50),
                        iterSim(nT = 100),
                        iterSim(nT = 200))) %>%
  map_dfr(.,~mutate(., k = factor(k, labels = c("x1", "+x2", "+x3", "+x4", "+x5",
                                                "+x6", "+x7", "+x8", "+x9")),
                    var = recode_factor(var, y = "y", x1 = "x1",
                                        .default = "unrelated"))) %>%
  group_by(k, n, var, type) %>%
  summarize_all(mean, na.rm = T)

## Appendix Figure 2: The consequences of GECMs with unbalanced equations (alternative DGP)
res2 %>% ungroup() %>%
  filter(var == "y") %>%
  ggplot(aes(x=k, y=comb_sig)) +
  geom_hline(yintercept=.05, lty = "dashed") +
  geom_line(aes(group=var)) + facet_wrap(~n) +
  theme_bw() + xlab("Independent variables included in GECM") +
  ylab(expression(paste("Proportion ",alpha[1]^"*"," significant at p<.05"))) +
  theme(text = element_text(size=8)) + ylim(0,1)
```

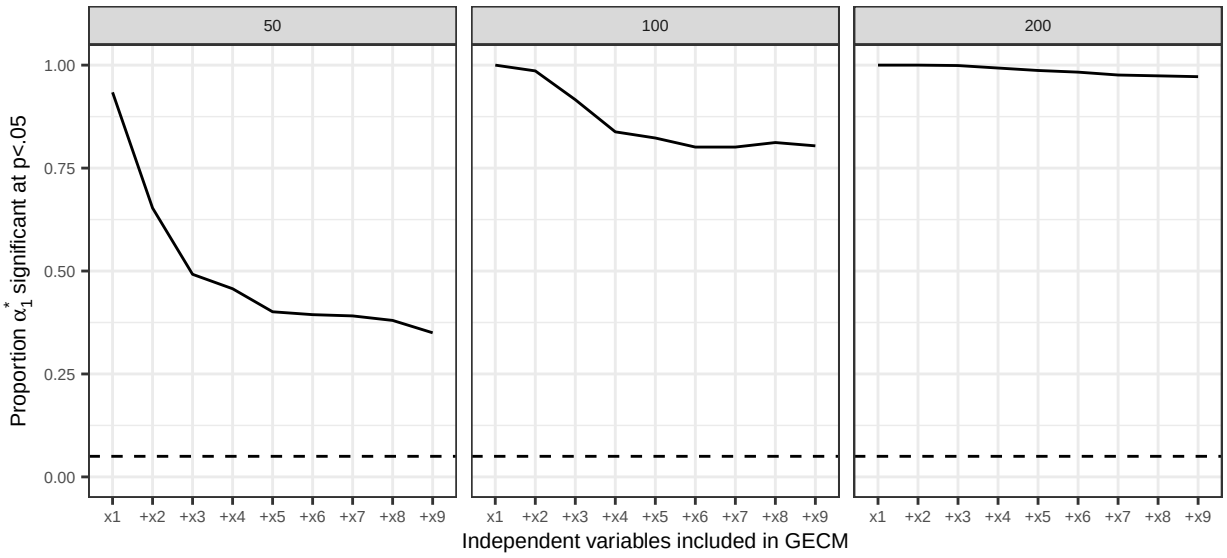



```
if(save_plots) ggsave("app1.pdf", height = 3, width = 6.5)

## Appendix Figure 3: The consequences of GECMs with unbalanced equations (alternative DGP)
res2 %>% ungroup() %>%
  filter(var == "unrelated") %>%
  mutate(type = gsub(" (","\n(", type, fixed =T)) %>%
  ggplot(aes(x=k, y=comb_sig)) +
  geom_bar(stat="identity") +
  theme_bw() + xlab("Independent variables included in GECM") +
  ylab(expression(paste("Average false positive rate for LRMs ",(beta[1]["j"]^"*"/-alpha[1]^"*")))) +
  geom_hline(yintercept=.05, lty = "dashed") + facet_wrap(~n) +
  theme(text = element_text(size=8))
```

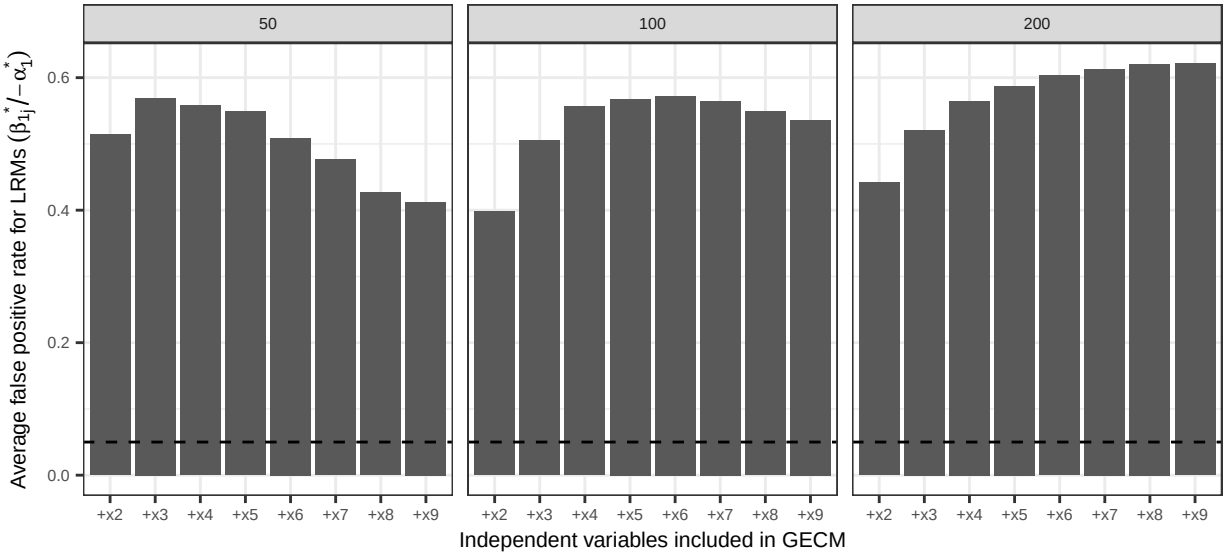



```
if(save_plots) ggsave("app2.pdf", height = 3, width = 6.5)


# Appendix: violating exogeneity assumption ------------------------------
```

```r
set.seed(20190226)

## run simulations
res3 <- rerun(nsim,
              bind_rows(iterSim(nT = 50, endo = 2:9),
                        iterSim(nT = 100, endo = 2:9),
                        iterSim(nT = 200, endo = 2:9))) %>%
  map_dfr(.,~mutate(., k = factor(k, labels = c("x1", "+x2", "+x3", "+x4", "+x5",
                                                "+x6", "+x7", "+x8", "+x9")),
                    var = recode_factor(var, y = "y", x1 = "x1",
                                        .default = "unrelated"))) %>%
  group_by(k, n, var, type) %>%
  summarize_all(mean, na.rm = T)

## Appendix Figure 4: The consequences of GECMs with unbalanced equations (violating exogeneity)
res3 %>% ungroup() %>%
  filter(var == "y") %>%
  ggplot(aes(x=k, y=comb_sig)) +
  geom_hline(yintercept=.05, lty = "dashed") +
  geom_line(aes(group=var)) + facet_wrap(~n) +
  theme_bw() + xlab("Independent variables included in GECM") +
  ylab(expression(paste("Proportion ",alpha[1]^"*"," significant at p<.05"))) +
  theme(text = element_text(size=8)) + ylim(0,1)
```




```r
if(save_plots) ggsave("app3.pdf", height = 3, width = 6.5)

## Appendix Figure 5: The consequences of GECMs with unbalanced equations (violating exogeneity)
res3 %>% ungroup() %>%
  filter(var == "unrelated") %>%
  mutate(type = gsub(" (","\n(", type, fixed =T)) %>%
  ggplot(aes(x=k, y=comb_sig)) +
  geom_bar(stat="identity") +
  theme_bw() + xlab("Independent variables included in GECM") +
  ylab(expression(paste("Average false positive rate for LRMs ",(beta[1]["j"]^"*"/-alpha[1]^"*")))) +
  geom_hline(yintercept=.05, lty = "dashed") + facet_wrap(~n) +
  theme(text = element_text(size=8))
```




```
if(save_plots) ggsave("app4.pdf", height = 3, width = 6.5)
```

Session Info

```
sessionInfo()
```

```
## R version 4.0.3 (2020-10-10)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 20.04.2 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/blas/libblas.so.3.9.0
## LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.9.0
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] broom_0.5.6   purrr_0.3.4   tidyr_1.1.0   dplyr_1.0.2   ggplot2_3.3.3
## [6] car_3.0-8     carData_3.0-4
##
## loaded via a namespace (and not attached):
##  [1] zip_2.0.4         Rcpp_1.0.6        pillar_1.4.6      compiler_4.0.3
##  [5] cellranger_1.1.0  forcats_0.5.0     tools_4.0.3       digest_0.6.25
##  [9] lattice_0.20-41   nlme_3.1-151      evaluate_0.14     lifecycle_0.2.0
## [13] tibble_3.0.3      gtable_0.3.0      pkgconfig_2.0.3   rlang_0.4.7
```

```
## [17] openxlsx_4.1.5    curl_4.3          yaml_2.2.1        haven_2.3.1
## [21] xfun_0.15         rio_0.5.16        withr_2.2.0       stringr_1.4.0
## [25] knitr_1.29        generics_0.0.2    vctrs_0.3.4       hms_0.5.3
## [29] tidyselect_1.1.0  grid_4.0.3        glue_1.4.2        data.table_1.12.8
## [33] R6_2.4.1          readxl_1.3.1      foreign_0.8-81    rmarkdown_2.6
## [37] farver_2.0.3      magrittr_1.5      backports_1.2.1   scales_1.1.1
## [41] ellipsis_0.3.1    htmltools_0.5.0   abind_1.4-5       colorspace_1.4-1
## [45] labeling_0.3      stringi_1.4.6     munsell_0.5.0     crayon_1.3.4
```